Implementation Guidance for
FIPS 140-2 and the Cryptographic Module
Validation Program

National Institute of Standards and Technology
Communications Security Establishment

Initial Release: March 28, 2003

Last Update: May 25, 2018

Table of Contents

Overview

This Implementation Guidance document is issued and maintained by the U.S. Government's National Institute of Standards and Technology (NIST) and the Communications Security Establishment Canada (CSE), which serve as the validation authorities of the Cryptographic Module Validation Program (CMVP) for their respective governments. The CMVP validates the test results of National Voluntary Laboratory Accreditation Program (NVLAP) accredited Cryptographic and Security Testing (CST) Laboratories which test cryptographic modules for conformance to Federal Information Processing Standard Publication (FIPS) 140-2, *Security Requirements for Cryptographic Modules*. The Cryptographic Algorithm Validation Program (CAVP) addresses the testing of *Approved Security Functions*, *Approved Random Number Generators* and *Approved Key Establishment Techniques* which are referenced in the annexes of FIPS 140-2.

This document is intended to provide programmatic guidance of the CMVP, and in particular, clarifications and guidance pertaining to the *Derived Test Requirements for FIPS PUB 140-2* (DTR)*, which is used by CST Laboratories to test for a cryptographic module's conformance to FIPS 140-2. Guidance presented in this document is based on responses issued by NIST and CSE to questions posed by the CST Labs, vendors, and other interested parties. *Information in this document is subject to change by NIST and CSE.*

Each section of this document corresponds with a requirements section of FIPS 140-2, with an additional first section containing general programmatic guidance that is not applicable to any particular requirements section. Within each section, the guidance is listed according to a subject phrase. For those subjects that may be applicable to multiple requirements areas, they are listed in the area that seems most appropriate. Under each subject there is a list, including the date of issue for that guidance, along relevant assertions, test requirements, and vendor requirements from the DTR. *(Note: For each subject, there may be additional test and vendor requirements which apply.)* Next, there is section containing a question or statement of a problem, along with a resolution and any additional comments with related information. This is the implementation guidance for the listed subject.

Cryptographic modules validation listings can be found at:

- Cryptographic Module Validation Lists

Cryptographic algorithm validation listings can be found at:

- Cryptographic Algorithm Validation Lists

General Issues

G.1 Request for Guidance from the CMVP and CAVP

Applicable Levels:	*All*
Original Publishing Date:	*02/25/1997*
Effective Date:	*02/25/1997*
Last Modified Date:	*08/07/2015*
Relevant Assertions:	*General*
Relevant Test Requirements:	
Relevant Vendor Requirements:	

Background

The Cryptographic Module Validation Program (CMVP) and the Cryptographic Algorithm Validation Program (CAVP) defines two types of questions: *Programmatic Questions* and *Test-specific Questions*. The CMVP and CAVP define two types of requests: *Informal Requests* and *Official Requests.*

Question/Problem

What is the difference between *Informal Requests* verses *Official Requests*? To whom should these questions be directed? If an official reply is requested for a question, is there a defined format for these types of requests?

Resolution

Programmatic Questions: These are questions pertaining to the general operation of the Cryptographic Module Validation Program or the Cryptographic Algorithm Validation Program. The CMVP and CAVP suggest reviewing the CMVP Management Manual, CMVP Frequently Asked Questions (FAQ), the CAVP Frequently Asked Questions (FAQ), CMVP Announcements and CMVP Notices posted on the CMVP and CAVP web sites first as the answer may be readily available. The information found on the CMVP web site provides the official position of the CMVP and CAVP.

Test-specific Questions: These are questions concerning specific test issues of the Cryptographic Module Validation Program or the Cryptographic Algorithm Validation Program. These issues may be technology related or related to areas of the standard that may appear to be open to interpretation.

General Guidance: Programmatic questions regarding the CMVP or the CAVP can be directed to either NIST or CSE by contacting the appropriate points of contact listed below. The complete list of NIST and CSE points of contacts **shall** be included on copy for all questions.

Vendors who are under contract with a CST laboratory for FIPS 140-2 or algorithm testing of a particular implementation(s) must contact the contracted CST laboratory for any questions concerning the test requirements and how they affect the testing of the implementation(s).

CST Laboratories must submit all *test-specific questions* in the RFG format described below. These questions must be submitted to all points of contact.

Federal agencies and departments, and vendors not under contract with a CST laboratory who have specific questions about a FIPS 140-2 test requirements or any aspect of the CMVP or CAVP should contact the appropriate NIST and CSE points of contact listed below.

Questions can either be submitted by e-mail, telephone, and facsimile or written (if electronic document, Microsoft Word document format is preferred).

Informal Request: Informal requests are considered as *ad hoc* questions aimed at clarifying issues about the FIPS 140-2 and other aspects of the CMVP and CAVP. Replies to informal requests by the CMVP are non-binding and subject to change. It is recommended that informal requests be submitted to all points of contact. Every attempt is made to reply to informal request with accurate, consistent, clear replies on a very timely basis.

Official Request: If an official response is requested, then an official request must be submitted to the CMVP and/or CAVP written in the Request for Guidance (RFG) format described below. An official response requires internal review by both NIST and CSE, as well as with others as necessary, and may require follow-up questions from the CMVP and/or CAVP. Therefore, such requests, while time sensitive, may not be immediate.

Request for Guidance Format: Questions submitted in this format will result in an official response from the CMVP and CAVP that will state current policy or interpretations. This format provides the CMVP and CAVP a clear understanding of the question. An RFG **shall** have the following items:

1. Clear indication of whether the RFG is **PROPRIETARY** or **NON-PROPRIETARY**,

2. A descriptive title,

3. Applicable statement(s) from FIPS 140-2,

4. Applicable assertion(s) from the FIPS 140-2 DTR,

5. Applicable required test procedure(s) from the FIPS 140-2 DTR,

6. Applicable statements from FIPS 140-2 Implementation Guidance,

7. Applicable statements from algorithmic standards,

8. Background information if applicable, including any previous CMVP or CAVP official rulings or guidance,

9. A concise statement of the problem, followed by a clear and unambiguous question regarding the problem, and

10. A suggested statement of the resolution that is being sought.

All questions should be presented in writing. The provided information should include a brief non-proprietary description of the implementation and the FIPS 140-2 target security level. All of this will enable a more efficient and timely resolution of FIPS 140-2 related questions by the CMVP and CAVP. The statement of resolution **shall** be stated in a manner which the CMVP and CAVP can either answer "YES" or "NO". The CMVP may optionally provide rationale if the answer is not in line with the suggested statement of resolution.

When appropriate, the CMVP and CAVP will derive general guidance from the problem and response, and add that guidance to this document. Note that general questions may still be submitted, but these questions should be identified as not being associated with a particular validation effort.

Preferably, questions should be non-proprietary, as their response will be distributed to ALL CST laboratories. Distribution may be restricted on a case-by-case basis.

NIST and CSE Points of Contact:

- **National Institute of Standards and Technology – CMVP**

    CMVP@nist.gov

- **National Institute of Standards and Technology (NIST) – CAVP**

    CAVPask@nist.gov

- **Communications Security Establishment Canada (CSE) – CMVP**
    CMVP@cse-cst.gc.ca

G.2 Completion of a test report: Information that must be provided to NIST and CSE

Applicable Levels:	*All*
Original Publishing Date:	*02/25/1997*
Effective Date:	*02/25/1997*
Last Modified Date:	*05/10/2016*
Relevant Assertions:	*General*
Relevant Test Requirements:	
Relevant Vendor Requirements:	

Question/Problem

What information should be submitted to NIST and CSE upon completion of the CST laboratory conformance testing in order for NIST and CSE to perform a validation review? Are there any other additional requirements during report COORDINATION?

Resolution

The following test report information **shall** be provided to both NIST and CSE by the CST laboratory upon report submission. The ZIP file and files within the ZIP file **shall** follow all programmatic naming conventions.[1] and be submitted to the CMVP using the specified encryption methods.

1. **Non-proprietary Security Policy** <pdf>

    a. Reference *FIPS 140-2 Appendix C*, *FIPS 140-2 DTR Appendix C* and the CMVP Implementation Guidance for requirements.
    b. The non-proprietary security policy **shall** not be marked as proprietary or copyright without a statement allowing copying or distribution.

2. **CRYPTIK v9.0c (or higher) Reports**

    The validation report submission **shall** be output from the NIST provided CRYPTIK tool.

    a. **Signature page** <insert PDF of signed signature page>

    b. **General Vendor/Module Information** < PDF>

    c. **Full Report with Assessments <** PDF**>**

    d. **Certificate** <DOC> or <DOCX>

        1. The RTF output from CRYPTIK **shall** be renamed to a DOC or DOCX file.
        2. **Shall** include PIV Card Application certificate number reference as applicable.

    e. **Vendor Text file** <TXT>
       Export the validation data and include the _vendor.txt file.

3. **Physical Security Test Report** <pdf – *mandatory* at FIPS 140-2 Section 4.5 Physical Security Levels 2, 3 and 4>

    The laboratory's physical testing report with photos, drawing, etc. as applicable.

---

[1] *CMVP Convention for E-mail Correspondence*

The physical security test evidence **shall** be traceable to the DTR by specifying the appropriate TE for each test described in the physical security test report.

4. **Revalidation Change Summary** <PDF – if applicable>

Reference IG G.8 for requirements.

5. **Entropy Report** <PDF> as required

The entropy report **shall** follow the guidelines in IG 7.15.

Note: Separate billing information is no longer required as it is part of the CRYPTIK _vendor.txt output.

The PDF files **shall** not be locked. All PDF submission documents (except Security Policy) **shall** be *merged* into a single PDF document in the following order: Signed Signature Page; General Vendor / Module Information; Executive Overview with Section Summaries or Re-Validation Report with Assessments; Full Report with Assessments; Physical Test Report as applicable; and Other as applicable.

The submission documents **shall** be ZIP'ed into a single file, encrypted (using the CMVP designated application) and sent to the following NIST and CSE points of contact:

- **NIST:** CMVP@nist.gov
- **CSE:** CMVP@cse-cst.gc.ca

Once the electronic report submission document is received by the CMVP it will be placed in the report queue in order received. Those reports marked to be listed, will appear in the weekly published Modules-In-Process listing posted on the CMVP web site. The listing and the definition of the five stages of the Modules-In-Process listing is found at: http://csrc.nist.gov/groups/STM/cmvp/inprocess.html

During the COORDINATION phase the CST laboratory will address each CMVP comment and update any applicable files as necessary in addition to providing a response and additional clarification as necessary in the CMVP comments document. The laboratory will re-submit the report in its entirety as above (i.e. full report submission) including the updated CMVP comments file.

6. **CMVP Comments** <DOC> or <DOCX>

Additional Comments

The naming convention for the submitted ZIP file, e-mail subject line, and files within the ZIP file is provided to the CST Labs in a separate document *CMVP Convention for E-mail Correspondence.* Contact cmvp@nist.gov and cmvp@cse-cst.gc.ca for the latest version of this document. The CRYPTIK *File I/O and EMAIL* function will generate the proper e-mail subject line name depending on the transaction.

An initial or preliminary review will be performed to ensure that the guidelines outlined in the *CMVP Convention for E-mail Correspondence* document have been followed and that required signatures have been included. During the initial review, the submission will not be checked for technical completeness. The report information in the _vendor.txt file will be imported to the CMVP Tracking DataBase and billing information, if applicable, will be sent to NIST billing. The weekly Modules-In-Process listing will be generated based on this provided information.

G.3 Partial Validations and Not Applicable Areas of FIPS 140-2

Applicable Levels:	*All*
Original Publishing Date:	*02/25/1997*
Effective Date:	*02/25/1997*
Last Modified Date:	*01/07/2014*
Relevant Assertions:	*General*
Relevant Test Requirements:	
Relevant Vendor Requirements:	

Question/Problem

Can a cryptographic module be validated only for selected areas of Section 4 of FIPS 140-2? Which areas of Section 4 of FIPS 140-2 can be marked *Not Applicable*?

Resolution

NIST and CSE will not issue a validation certificate unless the cryptographic module meets at least the Security Level 1 requirements for each area in Section 4 of FIPS 140-2 that cannot be designated as *Not Applicable* according to the following:

- **Section 4.5**, Physical Security may be designated as *Not Applicable* if the cryptographic module is a software-only module and thus has no physical protection mechanisms;

- **Section 4.6**, Operational Environment may be designated as *Not Applicable* depending on the module implementation (e.g. if the operational environment for the cryptographic module is a limited or non-modifiable operational environment); and

- **Section 4.11**, Mitigation of Other Attacks is *Applicable* if the module has been *purposely* designed, built and publicly documented to mitigate one or more specific attacks (RE: IG 11.1). Otherwise this section may be designated as *Not Applicable*.

The CST laboratory **shall** provide in the validation test report the rationale for marking sections as *Not Applicable*.

Additional Comments

If a section is *Not Applicable*, it will be identified as N/A on the module validation certificate entry. If Section 4.6 is N/A, depending on the module implementation, configuration information may still be required on the module validation certificate (e.g. a *firmware* module must provide the tested configuration).

G.4 Design and testing of cryptographic modules

Applicable Levels:	*All*
Original Publishing Date:	*11/12/1997*
Effective Date:	*11/12/1997*
Last Modified Date:	*01/07/2014*
Relevant Assertions:	*General*
Relevant Test Requirements:	
Relevant Vendor Requirements:	

Question/Problem

What activities may CST laboratories perform, regarding the design and testing of cryptographic modules?

Resolution

The following information is supplemental to the guidance provided by NVLAP, and further defines the separation of the design, consulting, and testing roles of the laboratories. CMVP policy in this area is as follows:

1. A CST Laboratory *may not* perform validation testing on a module for which the laboratory has:

   a. designed any part of the module,

   b. developed original documentation for any part of the module,

   c. built, coded or implemented any part of the module, or

   d. any ownership or vested interest in the module.

2. Provided that a CST Laboratory has met the above requirements, the laboratory *may* perform validation testing on modules produced by a company when:

   a. the laboratory has no ownership in the company,

   b. the laboratory has a completely separate management from the company, and

   c. business between the CST Laboratory and the company is performed under contractual agreements, as done with other clients.

3. A CST Laboratory may perform consulting services to provide clarification of FIPS 140-2, the Derived Test Requirements, and other associated documents at any time during the life cycle of the module.

Additional Comments

Item 3 in the Resolution references "other associated documents". Included in this reference are:

- Documents developed by the CMVP for the Cryptographic Module testing program (e.g., *CMVP and FIPS 140-2 Implementation Guidance*, *CMVP FAQs*, *CMVP Management Manual*, NVLAP Handbook 150-17:2012, *Cryptographic Module Testing*).

Also, see IG G.9, regarding FSM and Security Policy consolidation and formatting.

G.5 Maintaining validation compliance of software or firmware cryptographic modules

Applicable Levels:	*All*
Original Publishing Date:	*11/21/1997*
Effective Date:	*11/21/1997*
Last Modified Date:	*11/20/2015*
Relevant Assertions:	*General*
Relevant Test Requirements:	
Relevant Vendor Requirements:	

Question/Problem

For a validated software or firmware cryptographic module, how may such a module be implemented so that compliance with the validation is maintained?

Resolution

The tested/validated module version, operational environment upon which it was tested, and the originating vendor are stated on the validation certificate. The certificate serves as the benchmark for the module-compliant configuration.

This guidance addresses two separate scenarios: actions a vendor can affirm or change to maintain a module's validation and actions a user can affirm to maintain a module's validation.

This guidance is *not applicable* for validated modules when FIPS 140-2 Section 4.5 Physical Security has been validated at Levels 2 or higher. Therefore, this guidance is only applicable at Level 1 for *firmware* or *hybrid* modules.

Vendor

1. A vendor may perform post-validation recompilations of a software or firmware module and affirm the modules continued validation compliance provided the following is maintained:

   a) Software modules that do not require any source code modifications (e.g., changes, additions, or deletions of code) to be recompiled and ported to another operational environment must:

      i) For **Level 1 Operational Environment**, a software cryptographic module will remain compliant with the FIPS 140-2 validation when operating on any general-purpose computer (GPC) provided that the GPC uses the specified single user operating system/mode specified on the validation certificate, or another compatible single user operating system, and

      ii) For **Level 2 Operational Environment**, a software cryptographic module will remain compliant with the FIPS 140-2 validation when operating on any GPC provided that the GPC incorporates the specified CC evaluated EAL2 (or equivalent) operating system/mode/operational settings or another compatible CC evaluated EAL2 (or equivalent) operating system with like mode and operational settings.

   b) Firmware modules (i.e. Operational Environment is *not applicable)* that do not require any source code modifications (e.g., changes, additions, or deletions of code) to be recompiled and its identified unchanged tested operating system (i.e. same version or revision number) may be ported together from one GPC or platform to another GPC or platform while maintaining the module's validation.

   c) Hybrid modules (i.e. Operational Environment may or may not be applicable depending if the controlling component is software or firmware) may be ported together from one GPC or platform to another GPC or operating platform while maintaining the module's validation provided that they do not require any of the following:

> i) software or firmware source code modifications (e.g., changes, additions, or deletions of code) to be recompiled and its identified unchanged tested operating system (i.e. same version or revision number);
>
> ii) hardware components utilized by the controlling software or firmware is not modified (e.g. changes, additions, or deletions).

The CMVP allows vendor porting and re-compilation of a validated software, firmware or hybrid cryptographic module from the operational environment specified on the validation certificate to an operational environment which was not included as part of the validation testing as long as the porting rules are followed. Vendors may affirm that the module works correctly in the new operational environment. However, the CMVP makes no statement as to the correct operation of the module or the security strengths of the generated keys when so ported if the specific operational environment is not listed on the validation certificate.

The vendor **shall** work with a CST laboratory to update the security policy and submit to the CMVP under one of the available revalidation scenarios (see IG G.8). The update would affirm and include references to the new operational environment(s), GPC(s) or platform(s). The module's Security Policy **shall** include a statement that no claim can be made as to the correct operation of the module or the security strengths of the generated keys when ported to an operational environment which is not listed on the validation certificate.

2. Software or firmware modules that require non-security relevant source code modifications (e.g., changes, additions, or deletions of code) to be recompiled and ported to another hardware or operational environment must be reviewed by a CST laboratory and revalidated per IG G.8 (1) to ensure that the module does not contain any operational environment-specific or hardware environment-specific code dependencies.

3. If the new operational environment and/or platform is requested to be updated on the validation certificate, the CST laboratory **shall** follow the requirements for non-security relevant changes in IG G.8 (1) and in addition, perform the regression test suite of operational tests included in **IG G.8** Table G.8.1. Underlying algorithm validations must meet requirements specified in IG 1.4.

   Upon re-testing and validation, the CMVP provides the same assurance as the original operational environment(s) as to the correct operation of the module when ported to the newly listed OS(s) and/or operational environment(s) which would be added to the modules validation web entry.

The vendor must meet all applicable requirements in FIPS 140-2 Section 4.10.

This policy only addresses the operational environment under which a software, firmware or hybrid module executes and does not affect requirements of the other sections of FIPS 140-2. A module must meet all requirements of the level stated.

IG 1.3 describes the difference in terminology between a *software* and a *firmware* module.

IG 1.9 describes the attributes and definition of a hybrid module.

User

A user may not modify a validated module. Any user modifications invalidate a modules validation. [1]

A user may perform post-validation porting of a module and affirm the modules continued validation compliance provided the following is maintained:

1. For **Level 1 Operational Environment**, a software, firmware or hybrid cryptographic module will remain compliant with the FIPS 140-2 validation when operating on any general purpose computer (GPC) or

---

[1] A user may post-validation recompile a module if the unmodified source code is available and the module's Security Policy provides specific guidance on acceptable recompilation methods to be followed as a specific exception to this guidance. The methods in the Security Policy must be followed without modification to comply with this guidance.

platform provided that the GPC for the software module, or software controlling portion of the hybrid module, uses the specified single user operating system/mode specified on the validation certificate, or another compatible single user operating system, or that the GPC or platform for the firmware module or firmware controlling portion of the hybrid module, uses the specified operating system on the validation certificate, and

2. For **Level 2 Operational Environment**, a software cryptographic module will remain compliant with the FIPS 140-2 validation when operating on any GPC provided that the GPC incorporates the specified CC evaluated EAL2 (or equivalent) operating system/mode/operational settings or another compatible CC evaluated EAL2 (or equivalent) operating system with like mode and operational settings.

The CMVP allows user porting of a validated software, firmware or hybrid cryptographic module to an operational environment which was not included as part of the validation testing. The user may affirm that the module works correctly in the new operational environment as long as the porting rules are followed. However, the CMVP makes no statement as to the correct operation of the module or the security strengths of the generated keys when ported and executed in an operational environment not listed on the validation certificate.

Additional Comments

Users include third party integrators or any entity that is not the originating vendor as specified on the validation certificate.

G.6 Modules with both a FIPS mode and a non-FIPS mode

Applicable Levels:	*All*
Original Publishing Date:	*03/11/1998*
Effective Date:	*03/11/1998*
Last Modified Date:	*07/15/2011*
Relevant Assertions:	*General*
Relevant Test Requirements:	
Relevant Vendor Requirements:	

Question/Problem

How can a module be defined, when it includes both FIPS-approved and non-FIPS approved security methods?

Resolution

A module that contains both FIPS-approved and non-FIPS approved security methods **shall** have at least one "FIPS mode of operation" - which *only* allows for the operation of FIPS-approved security methods. This means that when a module is in the "FIPS mode", a non-FIPS approved method **shall** not be used in lieu of a FIPS-approved method (For example, if a module contains both MD5 and SHA-1, then when hashing is required in the FIPS mode, SHA-1 **shall** be used.). The operator must be made aware of which services are FIPS 140-2 compliant.

The FIPS 140-2 validation certificate will identify the cryptographic module's "FIPS mode" of operation.

For modules that support both FIPS approved and non-approved modes of operation, the certificate **shall** only list what is used in the approved mode of operation (i.e. all approved and allowed algorithms implemented within the module) while the Security Policy **shall** list what is used in both approved and non-approved modes (i.e. all the approved, allowed, and non-approved algorithms implemented within the module).

The selection of "FIPS mode" does not have to be restricted to any particular operator of the module. However, each operator of the module must be able to determine whether or not the "FIPS mode" is selected.

There is no requirement that the selection of a "FIPS mode" be permanent.

G.7 Relationships Among Vendors, Laboratories, and NIST/CSE

Applicable Levels:	*All*
Original Publishing Date:	*04/14/1998*
Effective Date:	*04/14/1998*
Last Modified Date:	*08/07/2015*
Relevant Assertions:	*General*
Relevant Test Requirements:	
Relevant Vendor Requirements:	

Question/Problem

What is the Cryptographic Module Validation Program policy regarding the relationships among vendors, testing laboratories, and NIST/CSE?

Resolution

The CST laboratories are accredited by NVLAP to perform cryptographic module validation testing to determine compliance with FIPS 140-2. NIST/CSE rely on the CST laboratories to use their extensive validation testing experience and expertise to make sound, correct, and independent decisions based on 140-2, the Derived Test Requirements, and Implementation Guidance. Once a vendor is under contract with a laboratory, NIST/CSE will only provide official guidance and clarification for the vendor's module through the point of contact at the laboratory.

In a situation where the vendor and laboratory are at an irresolvable impasse over a testing issue, the vendor may ask for clarification/resolution directly from NIST/CSE. The vendor should use the format required by Implementation Guidance IG G.1 and the point of contact at the laboratory **shall** be carbon copied. All correspondence from NIST/CSE to the vendor on the issue will be issued through the laboratory point of contact.

G.8 Revalidation Requirements

Applicable Levels:	*All*
Original Publishing Date:	*08/17/2001*
Effective Date:	*02/01/2017*
Last Modified Date:	*05/25/2018*
Relevant Assertions:	*General*
Relevant Test Requirements:	
Relevant Vendor Requirements:	

Question/Problem

What is the Cryptographic Module Validation Program (CMVP) policy regarding revalidation requirements and validation of a new cryptographic module that is significantly based on a previously validated module?

Resolution

An updated version of a previously validated cryptographic module can be considered for a *revalidation* rather than a *full validation* depending on the extent of the modifications from the previously validated version of the

module. (Note: the updated version may be, for example, a new version of an existing cryptographic module or a new model based on an existing model.)

There are seven possible **change** Scenarios (1, 1A, 1B, 2, 3, 4, 5):

Scenario 1:

Scenario 1 includes the following options:

1) Administrative updates (e.g. updating vendor contact information.)

2) Modifications are made to hardware, software or firmware components **that do not affect any FIPS 140-2 security relevant items**. The vendor is responsible for providing the applicable documentation to the CST laboratory, which identifies the modification(s). Documentation may include a previous validation report, design documentation, source code, source code difference evidence, etc.

3) Post validation, approved security relevant functions or services for which testing was not available at the time of validation, or security relevant functions or services that were not tested during the original validation, are now tested and are being submitted for inclusion as a FIPS approved function or service. The CST laboratory is responsible for identifying the documentation that is needed to determine whether a revalidation is sufficient and the vendor is responsible for submitting the requested documentation to the CST laboratory. Documentation may include a previous validation report and applicable CMVP rulings, design documentation, source code, etc.

Modules with certificates on the *Validated FIPS 140-1 and 140-2 Cryptographic Module List* may be submitted under any of the options listed.

Modules with certificates on the *CMVP Historical Validation List* may only be submitted under option 1. The CMVP will not accept options 2 and 3 for modules with certificates on the *CMVP Historical Validation List*.

For options 2 and 3, the CST laboratory **shall**:

- review the vendor-supplied documentation and identify any additional documentation requirements.

- determine additional testing as necessary to confirm that FIPS 140-2 security relevant items have not been affected by the modification.

- identify the assertions affected and **shall** perform the tests associated with those assertions by:

  o reviewing the COMPLETE list of assertions for the module embodiment and security level;

  o identifying from the previous validation report, the assertions that are newly tested;

  o identifying additional assertions that were previously tested but should now be re-tested; and

  o reviewing assertions where specific Implementation Guidance (IG) was provided at the time of the original validation to confirm that the IG is still applicable.

Upon successful review and applicable testing as required, the CST laboratory **shall** submit a signed explanatory letter that contains a description of the modification(s) and lists the affected TEs and their associated laboratory assessment.

When the certificate is being updated, the CST laboratory **shall** use the following format for listing the modifications to the certificate. Deletions **shall** be marked using strikethrough and additions **shall** be highlighted in yellow. This information **shall** be listed in the change letter.

For example:

Current Cert. #5000	Change Requested Cert. #5000

Hardware Version – 3.1	Hardware Versions 3.1, 3.2
Firmware Version – a.1, b.1	Firmware Versions – a.1, b.1, c.1
FIPS Approved Algorithms – AES (Cert. #1); DRBG (Cert. #1); DSA (Cert. #1); ECDSA (Cert. #1); HMAC (Cert. #1); ~~KBKDF (vendor affirmed);~~ RSA (Cert. #1); SHS (Cert. #1); Triple-DES (Cert. #1)	FIPS Approved Algorithms – AES (Cert. #1); DRBG (Cert. #1); DSA (Cert. #1); ECDSA (Cert. #1); HMAC (Cert. #1); KBKDF (Cert. #1); KTS (AES Cert. #1; key establishment methodology provides between 128 and 256 bits of encryption strength); RSA (Cert. #1); SHS (Cert. #1); Triple-DES (Cert. #1)
Other Algorithms – ~~AES (Cert #1, key wrapping, key establishment methodology provides between 128 and 256 bits of encryption strength),~~ DES	Other Algorithms - DES

When the module's documentation is being updated, the CST laboratory **shall** use the following format for listing the affected TEs and their associated laboratory assessment. This information **shall** be listed in the change letter.

For example:

TE or SP Section	Related Change
Module Information	The module name and firmware versions have been updated from version 05 to version 06.
TE.01.03.02 TE.01.08.01	Updated to reflect the updated firmware version, 06.
References	Updated security policy version number and added the vendor provided document listing the differences between the original validation and the revalidation.

The assessment **shall** include the analysis performed by the laboratory that confirms that no security relevant items were affected. The letter **shall** also indicate whether the modified cryptographic module replaces the previously validated module or adds to the latter. If new algorithm certificates were obtained, they **shall** be listed.

A new security policy **shall** be provided for posting if the modifications cause changes to it or updates the new services or functions that are now included in an approved mode of operation as a result of algorithm testing. If the security policy represents multiple versions of a validated module or multiple validated modules, the versioning information **shall** be updated in the security policy with text that clearly distinguishes each module instance with its unique versioning information and the differences between each module instance.

For a scenario 1 revalidation, the CST laboratory **shall** submit, at a minimum, an encrypted ZIP file containing the unsigned letter <pdf>, image of the signed letter <pdf> and the _vendor.txt file. If the security policy or validation certificate are updated, the CST laboratory **shall** include the updated security policy <pdf> and draft certificate <doc or docx or rtf>. The ZIP file and files within the ZIP file **shall** follow the *CMVP Convention for E-mail Correspondence* and submitted to the CMVP using the specified encryption methods.

The CST laboratory may combine multiple scenario 1 revalidations into 1 submission provided ALL of the changes are exactly the same for all certificates. If multiple security policies are updated, the submission **shall** include a security policy for each certificate included in the submission.

Upon a satisfactory review by the CMVP, the updated version or release information will be posted on the *Validated FIPS 140-1 and FIPS 140-2 Cryptographic Module List* web site entry associated with the original cryptographic module. A new certificate will not be issued. The sunset date for the certificate will not be changed.

Note: a scenario 1 submission *will not* be included on the CMVP MIP list.

Alternative Scenario 1A:

1. Alternative scenario 1A applies if there are no modifications to a module and the new module is a re-branding of an already validated Original Equipment Manufacturer (OEM) module. The CST laboratory **shall** check the OEM's approval for rebranding and determine that the re-branded module is identical to the OEM module. The test report submission **shall** include a letter requesting the validation of the re-branded module and indicate the applicable documentation changes (e.g. vendor name, address, POC information, versioning information, etc.).

2. Alternative scenario 1A applies if the module is a ported sub-chip cryptographic subsystem. Please see IG 1.20 for detailed porting guidance.

For options 1 and 2, only modules with certificates on the *Validated FIPS 140-1 and 140-2 Cryptographic Module List* may be used for scenario 1A modules. Modules with certificates on the *CMVP Historical Validation List* **shall not** be used for scenario 1A modules.

The CST laboratory **shall** use the following format for listing the information for the new certificate. This information **shall** be listed in the change letter.

For example:

Current Cert. #5000	New Certificate Information
Hardware Version – 3.0	Hardware Version – AA
Firmware Version – 8.3	Firmware Version – XZ
Product Link – www.productA.com	Product Link – www.productB.com
Vendor Name – Vendor A	Vendor Name – Vendor B

The laboratory **shall** provide an updated security policy which is technically identical to the originally validated security policy and describes the re-branded module.

For a scenario 1A revalidation, the CST laboratory **shall** submit, at a minimum, an encrypted ZIP file containing the unsigned letter <pdf>, image of the signed letter <pdf>, the _vendor.txt file, the security policy <pdf> and draft certificate <doc or docx or rtf>. The ZIP file and files within the ZIP file **shall** follow the *CMVP Convention for E-mail Correspondence* and submitted to the CMVP using the specified encryption methods.

NIST CR is applicable. A new validation certificate will be issued. The new validation certificate will inherit the sunset date of the original certificate.

Note: a scenario 1A submission *will not* be included on the CMVP MIP list.

Alternative Scenario 1B:

A CST laboratory has been contracted to perform a scenario 1 revalidation for a validated module for which the laboratory did not perform the testing on the module which is the basis of the scenario 1 revalidation.

a. The vendor **shall** provide the laboratory with the design documentation and implementation (including source code, HDL, etc.) of the base validated module and of the module that has been updated with the non-security relevant changes.

b. The laboratory **shall** determine that the provided base documentation and implementation is identical to the base validated module.

c. The laboratory **shall** examine each modification and confirm that the change is non-security relevant.

d. The laboratory **shall** determine that no other modifications, including unintentional, have been made that are not documented and verified to be non-security relevant.

Only modules with certificates on the *Validated FIPS 140-1 and 140-2 Cryptographic Module List* may be used for scenario 1B modules. Modules with certificates on the *CMVP Historical Validation List* **shall not** be used for scenario 1B modules.

The CST laboratory **shall** use the following format for listing the information for the new certificate. This information **shall** be listed in the change letter.

For example:

Current Cert. #5000	New Certificate Information
Firmware Version 3.1	Firmware Version 1.1
Operational Environments – Tested as meting Level 1 with Windows Server 2008 R2 on a Dell OptiPlex 755, SUSE Linux Enterprise 11 SP2 on a Dell OptiPlex 755, CentOS 6.3 on a GigaVUE TA1 (single user mode)	Operational Environments – Tested as meeting Level 1 with Windows Server 2008 R2 on a Dell OptiPlex 755, SUSE Linux Enterprise 11 SP2 on a Dell OptiPlex 755 (single user mode)
Module Name – Module A	Module Name – Module B

For a scenario 1B revalidation, the CST laboratory **shall** submit, at a minimum, an encrypted ZIP file containing the unsigned letter <pdf>, image of the signed letter <pdf>, the _vendor.txt file, the security policy <pdf> and draft certificate <doc or docx or rtf>. The ZIP file and files within the ZIP file **shall** follow the *CMVP Convention for E-mail Correspondence* and submitted to the CMVP using the specified encryption methods.

NIST CR is applicable. A new validation certificate will be issued with a reference to the new laboratory's NVLAP code. The new validation certificate will inherit the sunset date of the original certificate. The new entry will only reference the new version that reflects the non-security relevant change. The validation entry caveat will include the following text:

> *This validation entry is a non-security-relevant modification to Cert. #nnnn*

Note: a scenario 1B submission *will not* be included on the CMVP MIP list.

Scenario 2:

Scenario 2 is for extending the module's sunset date when a module has not changed. The module meets all of the latest standards, implementation guidance and algorithm testing in effect at the time the module revalidation package is submitted to the CMVP unless there is an implementation guidance transition that affects reports that have been submitted.

The laboratory **shall** confirm the module has not changed. If there are any changes to the module, it is a new module and must be submitted as a scenario 3 or 5.

Modules with certificates on both the *Validated FIPS 140-1 and 140-2 Cryptographic Module List* may be used for scenario 2, as well as modules with certificates on the *CMVP Historical Validation List*.

Upon successful review and applicable testing to confirm the module has not changed and meets the latest standards, implementation guidance and algorithm testing, the CST laboratory **shall** submit a signed explanatory letter that contains a rationale for extending the sunset date and a statement from the vendor that the module is still being supported by the vendor. It is permissible to include vendor contact updates as well as updates to the security policy, where these updates are added to meet documentation requirements in the latest implementation guidance. The security policy may also be updated to reflect **SP 800-131Arev1** transitions, where the changes are made in documentation only and no changes were made to the module. All changes to the security policy **shall** be listed in the signed explanatory letter.

For a scenario 2 revalidation, the CST laboratory **shall** submit, at a minimum, an encrypted ZIP file containing the unsigned letter <pdf>, image of the signed letter <pdf>, the _vendor.txt file, security policy <pdf> (even if the security policy has not changed), draft certificate <doc or docx or rtf> and the test report <pdf>. The ZIP file and files within the ZIP file **shall** follow the *CMVP Convention for E-mail Correspondence* and submitted to the CMVP using the specified encryption methods.

Additional documentation (e.g. entropy report) may be required if implementation guidance requiring the additional documentation has been published since the module's original validation.

Upon a satisfactory review by the CMVP, the security policy will be posted on the *Validated FIPS 140-1 and FIPS 140-2 Cryptographic Module List* web site and the sunset date will be extended 5 years from revalidation date.

Note: a scenario 2 submission *will not* be included on the CMVP MIP list.

Scenario 3:

Modifications are made to hardware, software or firmware components **that affect some of the FIPS 140-2 security relevant items**. An updated cryptographic module can be considered in this scenario if it is similar to the original module with only minor changes in the security policy and FSM, and less than 30% of the modules security relevant features[1].

The CST laboratory is responsible for identifying the documentation that is needed to determine whether a revalidation is sufficient and the vendor is responsible for submitting the requested documentation to the CST laboratory. Documentation may include a previous validation report and applicable CMVP rulings, design documentation, source code, etc.

Modules with certificates with Validation Status as *Active* or *Historical* are eligible for scenario 3 revalidation.

The CST laboratory **shall** identify the assertions affected by the modification and **shall** perform the tests associated with those assertions. This will require the CST laboratory to:

a. Review the COMPLETE list of assertions for the module embodiment and security level,

b. Identify, from the previous validation report, the assertions that have been affected by the modification,

c. Identify additional assertions that were NOT previously tested but should now be tested due to the modification, and

d. Review assertions where specific Implementation Guidance (IG) was provided to confirm that the IG is still applicable.

For example, a revision to a firmware component that added security functionality may require a change to assertions in Section 1.

In addition to the tests performed against the affected assertions, the CST laboratory **shall** also perform the regression test suite of operational tests included in Table G.8.1.

The CST laboratory **shall** use the following format for listing the affected TEs and their associated laboratory assessment. This information **shall** be listed in the beginning of the test report.

For example:

TE or SP Section	Assessment
General	The module's name has been changed from ModuleA to ModuleB
1. Cryptographic Module Specification	01.03.02 and 01.08.05 have been updated for clarification on how to bring the module in the approved mode of operation.
	01.06.02, 01.08.03, 01.08.04, 01.08.07, 01.08.10, 01.13.01, 01.14.01 have been updated to reference to the new security policy.
	01.06.03 has been updated to mention the new test platforms.

---

[1] For example, security relevant features may include addition/deletion/change of minor components and their composition, addition/deletion of ports and interfaces, addition/delete/modification of security functions, modification of the physical boundary and protection mechanisms. These changes may affect many TE's yet be considered a minor change (<30%), or affect few TE's yet be a gross change (>30%).

	01.08.01 has been updated to reference the updated operating environment.
	01.12.01 has been updated to mention the CAVS tool version used for CAVS testing, the new algorithm certificates.
	01.12.02 has been updated to clarify which non-FIPS approved algorithms are available to the user of the module.
	01.08.02 has been updated to mark some bullets as not applicable.
2. Cryptographic Modules Ports and Interfaces	02.01.01, 02.01.02, 02.01.03, 02.04.01, 02.09.01, 02.11.01, 02.12.01 have been updated to reference to the new security policy.
	02.06.01 has been updated to updated the testing approach.
3. Roles, Services, and Authentication	03.02.01, 03.11.01, 03.14.01 have been updated to reference to the new security policy.
	03.06.01, 03.06.02 have been updated to better reflect the services available to each role.
	03.02.01, 03.02.02 and 03.02.03 have been marked as not applicable.
4. Finite State Model	04.05.01 has been updated to add the state transitions.
	04.05.02 has been updated to clarify the differences between the crypto officer and user role.
5. Physical Security	No change
6. Operational Environment	06.04.01, 06.06.01 have been updated to reference to the new security policy.
	06.05.01 has been updated to clarify that the module does not support key generation.
	06.07.01 has been updated to reference to the new files comprising the module.
	06.08.02 has been updated to reference to the new module's file version and naming.
	06.05.01 has been updated to replace the DSA algorithm with RSA.
7. Cryptographic Key Management	07.01.01 has been updated to reference to the new security policy.
	07.02.01, 07.02.02 has been updated to clarify the RSA signature verification mechanism available by the module replacing the DSA algorithm.
	07.03.01 has been updated to clarify that the module does not support key generation.
	07.13.01, 07.13.02 have been updated to the address IG 7.15.
	07.23.01 has been updated to clarify that the SP800-90A DRBG implementation is automatically seeded by the module.
8. EMI/EMC	08.02.01 has been updated to mention the new test platforms FCC evidence.
9. Self-Tests	09.06.02 has been modified to mention a new testing approach.
	09.07.02 has been updated to add the transition from the operational state to the error state.
	09.09.02 and 09.22.07 have been updated to replace the term "kernel module" with the term "kernel loadable component".
	09.07.01, 09.18.01, 09.18.02, 09.18.03, 09.22.01, 09.22.02, 09.22.05, 09.22.06, 09.24.01, 09.35.01, 09.35.02, 09.35.03, 09.35.04 have been updated to replace the DSA signature verification with RSA.

	09.16.01 has been updated to update the last paragraph regarding the block chaining modes.
	09.16.02 has been updated to reflect the new KATs performed by the module.
	09.20.01 has been updated for a new source code review.
	09.22.03 has been updated to replace the DSA algorithm with RSA.
	09.35.05 has been updated to modify the kernel component that was tested.
	09.42.01 has been updated to remove ANSI CPRNG from the FIPS approved algorithms.
	09.43.01 has been updated to mention the DRBG which is the only approved RNG for the module.
10. Design Assurance	10.01.01, 10.02.01, 10.02.02, 10.02.03, 10.02.04 have been updated to remove CVS which has been fully replaced by GIT.
	10.03.02, 10.23.01 have been updated to reference to the new security policy document.
11. Mitigation of Attacks	No change

The CST laboratory must provide a summary of the changes and rationale of why this meets the <30% guideline. The CMVP upon review, may determine that the changes are >30% and **shall** be submitted as a full report. The CST laboratory **shall** document the test results in the associated assessments and all affected TEs **shall** be annotated as "re-tested." The CST laboratory **shall** submit a test report as specified in IG G.2 describing the modification and highlighting those assertions that have been modified and retested (selecting the re-tested option in CRYPTIK). Upon a satisfactory review by the CMVP, the updated version will be revalidated to FIPS 140-2.

NIST CR is applicable. For a scenario 3 revalidation, the CST laboratory **shall** submit, at a minimum, an encrypted ZIP file containing the _vendor.txt file, the security policy <pdf>, test report <pdf>, and draft certificate <doc or docx or rtf>. The ZIP file and files within the ZIP file **shall** follow the *CMVP Convention for E-mail Correspondence* and submitted to the CMVP using the specified encryption methods.

Upon a satisfactory review by the CMVP, the updated security policy and information will be posted on the *Validated FIPS 140-1 and FIPS 140-2 Cryptographic Module List*. A new certificate will be issued and will have a sunset date 5 years from the validation date.

Note: a scenario 3 submission *will* be included on the CMVP MIP list.

Alternative Scenario 3A:

A CST laboratory has been contracted to perform a revalidation for a module on which the vendor has made FIPS 140 security-relevant changes in response to one or more CVEs (Common Vulnerability and Exposure). For more information about CVEs please see https://cve.mitre.org/.

The purpose of the 3A revalidation scenario is to provide the vendor a means to quickly fix, test and revalidate a module that is subject to a *security-relevant CVE[1]*, while at the same time providing assurance that the module still meets the FIPS 140-2 standard. If a CVE does not require security relevant changes to address it, then the vendor may pursue a Scenario 1 revalidation.

To complete a Scenario 3A revalidation:

a. The CST laboratory **shall** determine that changes to the module are only to correct the vulnerability disclosed in the CVE.

b. The CST laboratory **shall** examine each modification and confirm that the change does not conflict with the requirements of FIPS 140-2.

c. The CST laboratory **shall** determine that no other modifications have been made.

d. The CST laboratory **shall** identify the assertions affected by the security-relevant modification and **shall** perform the tests associated with those assertions.

e. The vendor is not required to address IGs that have been published since the original validation.

f. If the fix to address the CVE is in the scope of an algorithm implementation, then this algorithm **shall** be CAVS tested again to obtain a new CAVP certificate with the new module version.

In addition to the tests performed against the affected assertions, the CST laboratory **shall** also perform the following regression suite of operational tests.

TE.01.03.02 - The tester shall invoke the Approved mode of operation using the vendor provided instructions found in the non-proprietary security policy.

TE.01.04.02 (levels 3 and 4) - The tester shall use the vendor provided instructions described in the non-proprietary security policy to obtain the Approved mode of operation indicator.

TE.02.06.02 - To the extent that the cryptographic module design and operating procedures allow, the tester shall cause the cryptographic module to enter each specified error state and verify that all data output via the data output interface is inhibited.

TE.02.06.04 - To the extent that the cryptographic module design and operating procedures allow, the tester shall command the module to perform the self-tests and verify that all data output via the data output interface is inhibited.

TE.04.05.08 - The tester shall exercise the cryptographic module, causing it to enter each of its major states of the Finite State Model.

TE.07.41.02 - The tester shall note which keys are present in the module and initiate the zeroize command.

TE.09.09.02 - The tester shall power-up the module and verify that the module performs the power-up self-tests without requiring any operator intervention.

Because the changes to address the CVEs are considered security relevant, the CST lab must submit an updated test report. The CST laboratory **shall** use the Scenario 3 table format for listing the affected TEs and their associated laboratory assessment. This information **shall** be listed in the beginning of the test report.

Modules with certificates on the *140-2 Cryptographic Module List* and on the *CMVP Historical Validation List* may be used for scenario 3A modules.

NIST CR[2] is not applicable. The laboratory **shall** submit a scenario 3A revalidation by using the 3SUB process and e-mail transmittal code, but **shall** clearly indicate in the letter that this is a revalidation in response to a CVE, and provide the relevant CVE number(s). The submitted package at a minimum **shall** consist of an encrypted ZIP file containing the unsigned letter <pdf>, image of the signed letter <pdf>, the _vendor.txt file, the updated security policy <pdf>, test report <pdf>, and draft certificate <doc or docx or rtf>. The ZIP file and files within the ZIP file **shall** follow the *CMVP Convention for E-mail Correspondence* and submitted to the CMVP using the specified encryption methods.

A new validation certificate will not be issued and the original sunset date will not be extended for modules on the active list. Because the change to the module is to address a security-relevant CVE, the previous version of the module is no longer considered validated and will be removed from the certificate; exceptions may be made if the vendor shows how the CVE can be mitigated by policies included in the Security Policy, while still adhering to the FIPS 140-2 standard.

Note: a scenario 3A submission *will not* be included on the CMVP MIP list.

[1] A *security-relevant CVE* is one that affects how the module meets the requirements of the FIPS 140-2 standard.

[2] Please note that ECR may still be applicable.

Scenario 4:

Modifications are made only **to the physical enclosure of the cryptographic module that provides its protection and involves no operational changes to the module**. The CST laboratory is responsible for ensuring that the change only affects the physical enclosure (integrity) and has no operational impact on the module. The CST laboratory **shall** fully test the physical security features of the new enclosure to ensure its compliance to the relevant requirements of the standard.

Only modules with certificates on the *Validated FIPS 140-1 and 140-2 Cryptographic Module List* may be submitted under scenario 4. Modules with certificates on the *CMVP Historical Validation List* will not be accepted.

The CST laboratory **shall** submit a letter to the CMVP that:

    a.   Describes the change (pictures may be required),

    b.   States that it is a security relevant change,

    c.   Provides sufficient information supporting that the physical only change has no operational impact,

    d.   Describes the tests performed by the laboratory that confirm that the modified enclosure still provides the same physical protection attributes as the previously validated module. For physical security levels 2, 3 and 4, the laboratory **shall** submit an updated Physical Security Test Report.

An example of such a change could be the plastic encapsulation of the Level 2 token which has been reformulated or colored. Therefore, the molding or cryptographic boundary has been modified. This change is security relevant as the encapsulation provides the opacity and tamper evidence requirements. But this can be handled as a letter only change with evidence that the new composition has the same physical security relevant attributes as the prior composition.

The CST laboratory **shall** include a new security policy for posting if the modifications cause changes to the areas addressed in FIPS 140-2 Appendix C. If the security policy represents multiple versions of a validated module or multiple validated modules, the versioning information **shall** be updated in the security policy with text that clearly distinguishes each module instance with its unique versioning information and the differences between each module instance.

The CST laboratory **shall** use the following format for listing the modifications to the certificate. Deletions **shall** be marked using strikethrough and additions **shall** be highlighted in yellow. This information **shall** be listed in the change letter.

For example:

Current Cert. #5000	New Certificate Information
Hardware Versions – AX12, AX13, ~~and~~ AX14 with FIPS kit AX00	Hardware Versions – AX12, AX13, AX14 and AX15 with FIPS kit AX00

The CST laboratory **shall** use the following format for listing the affected TEs and their associated laboratory assessment. This information **shall** be listed in the change letter.

For example:

TE or SP Section	Related Change
TE.01.08.02 TE.01.08.03 TE.01.08.12	New version of the hardware. Added to Bill of Materials.

| TE.02.09.01
TE.02.09.02 | Updated hardware version and power supply added. |
| TE.10.02.01
TE.10.02.02
TE.10.02.03
TE.10.02.04 | Updated version of configuration items. |

For a scenario 4 revalidation, the CST laboratory **shall** submit, at a minimum, an encrypted ZIP file containing the unsigned letter <pdf>, image of the signed letter <pdf>, the _vendor.txt file and physical security test report <pdf>. The ZIP file and files within the ZIP file **shall** follow the *CMVP Convention for E-mail* Correspondence and submitted to the CMVP using the specified encryption methods.

Upon a satisfactory review by the CMVP, the updated security policy and information will be posted on the *Validated FIPS 140-1 and FIPS 140-2 Cryptographic Module List* web site entry associated with the original cryptographic module. A new certificate will not be issued. The sunset date of the certificate will not be changed.

Note: a scenario 4 submission *will not* be included on the CMVP MIP list.

Scenario 5:

If modifications are made to hardware, software, or firmware components **that do not meet any of the above criteria**, then the cryptographic module **shall** be considered a new module and **shall** undergo a full validation testing by a CST laboratory. The CST laboratory **shall** submit a test report as specified in IG G.2. Scenario 5 is also applicable for a module that is eligible for scenario 3 but the original laboratory is not performing the revalidation. NIST CR is applicable. A new certificate will be issued.

Note: a scenario 5 submission *will* be included on the CMVP MIP list.

Additional Comments

A cryptographic module that is changed under change Scenarios 1, 1A, 1B and 4, must be listed on the *Validated FIPS 140-1 and 140-2 Cryptographic Modules List* at the time of submission and must meet ALL standards, implementation guidance and algorithm testing that were met at the time of original validation. A module does not need to meet requirements that were added since the time of original validation. Modules on the *CMVP Historical Validation List* are not eligible for revalidations under scenarios 1 (including 1A and 1B) or 4. The only exception is to allow vendor contact updates under scenario 1.

A cryptographic module that is changed under Scenarios 2, 3 and 5 above, must meet ALL standards, implementation guidance and algorithm testing in effect at the time the module report is submitted to the CMVP unless there is an implementation guidance transition that affects reports that have been submitted. The CST laboratory is responsible for requesting from the vendor all the documentation necessary to determine whether the cryptographic module meets the current standards and implementation guidance. This is particularly important for features/services of the cryptographic module that required a specific ruling from the CMVP.

For example, a cryptographic module may have been validated with an implementation of KBKDF prior to when KBKDF testing was available. If the same cryptographic module is later submitted for revalidation under scenarios 3 and 5, this KBKDF implementation to be used in an approved mode of operation **shall** be tested and validated against **SP 800-108**, and the cryptographic module must meet the applicable FIPS 140-2 requirements, e.g., self-tests.

If the overall Security Level of the cryptographic module is lowered, the module may be submitted as a 3SUB with full testing on the individual section(s) that is being lowered.

If the overall Security Level of the cryptographic module is raised or if the physical embodiment changes, e.g., from multi-chip standalone to multi-chip embedded, then the cryptographic module will be considered a new module and **shall** undergo full validation testing by a CST laboratory.

The sunset date for the module is determined based on the scenario.

- Scenario 1 – sunset date unchanged

- Scenarios 1A and 1B – sunset date is inherited from the original certificate

- Scenario 2 – sunset date is extended 5 years from the revalidation date

- Scenario 3 – new certificate issued; sunset date will be 5 years from the validation date

- Scenario 4 – sunset date unchanged

- Scenario 5 – new certificate issued; sunset date will be 5 years from the validation date

The NIST CR schedule is available on the CMVP web site.

Table G.8.1 – Regression Test Suite

AS	TE	Security Level			
		1	2	3	4
Regression Testing Table					
Section 1 - Cryptographic Module Specification					
AS.01.03	TE.01.03.02	x	x	x	x
Section 2 - Cryptographic Module Ports and Interfaces					
AS.02.06	TE.02.06.02	x	x	x	x
	TE.02.06.04	x	x	x	x
AS.02.13	TE.02.13.03	x	x	x	x
AS.02.14	TE.02.14.02	x	x	x	x
AS.02.16	TE.02.16.02			x	x
AS.02.17	TE.02.17.02			x	x
Section 3 - Roles, Services and Authentication					
AS.03.02	TE.03.02.02	x	x	x	x
	TE.03.02.03	x	x	x	x
AS.03.12	TE.03.12.03	x	x	x	x
AS.03.13	TE.03.13.02	x	x	x	x
AS.03.14	TE.03.14.02	x	x	x	x
AS.03.15	TE.03.15.02	x	x	x	x
AS.03.17	TE.03.17.02		x		
AS.03.18	TE.03.18.02		x		
AS.03.19	TE.03.19.02			x	x
	TE.03.19.03			x	x
AS.03.21	TE.03.21.02	x	x	x	x
AS.03.22	TE.03.22.02		x	x	x
AS.03.23	TE.03.23.02	x	x	x	x
Section 4 - Finite State Model					
AS.04.03	TE.04.03.01	x	x	x	x
AS.04.05	TE.04.05.08	x	x	x	x
Section 5 - Physical Security					
	NONE				
Section 6 - Operational Environment					

AS.06.05	TE.06.05.01	x			
AS.06.06	TE.06.06.01	x			
AS.06.07	TE.06.07.01	x	x	x	x
AS.06.08	TE.06.08.02	x	x	x	x
AS.06.11	TE.06.11.02		x	x	x
	TE.06.11.03		x	x	x
AS.06.12	TE.06.12.02		x	x	x
	TE.06.12.03		x	x	x
AS.06.13	TE.06.13.02		x	x	x
	TE.06.13.03		x	x	x
AS.06.14	TE.06.14.02		x	x	x
	TE.06.14.03		x	x	x
AS.06.15	TE.06.15.02		x	x	x
AS.06.16	TE.06.16.02		x	x	x
AS.06.17	TE.06.17.02		x	x	x
AS.06.22	TE.06.22.02			x	x
	TE.06.22.03			x	x
AS.06.24	TE.06.24.02			x	x
	TE.06.24.03			x	x
AS.06.25	TE.06.25.02			x	x
Section 7 - Cryptographic Key Management					
AS.07.01	TE.07.01.02	x	x	x	x
AS.07.02	TE.07.02.02	x	x	x	x
AS.07.15	TE.07.15.02	x	x	x	x
	TE.07.15.03	x	x	x	x
	TE.07.15.04	x	x	x	x
AS.07.25	TE.07.25.02	x	x	x	x
AS.07.27	TE.07.27.02	x	x	x	x
AS.07.28	TE.07.28.02	x	x	x	x
AS.07.29	TE.07.29.02	x	x	x	x
AS.07.31	TE.07.31.04			x	x
AS.07.39	TE.07.39.02	x	x	x	x
AS.07.41	TE.07.41.02	x	x	x	x
Section 8 - EMI / EMC					
	As Required				
Section 9 - Self Tests					
AS.09.04	TE.09.04.03	x	x	x	x
AS.09.05	TE.09.05.03	x	x	x	x
AS.09.09	TE.09.09.02	x	x	x	x
AS.09.10	TE.09.10.02	x	x	x	x
AS.09.12	TE.09.12.02	x	x	x	x
AS.09.22	TE.09.22.07	x	x	x	x
AS.09.35	TE.09.35.05	x	x	x	x
AS.09.40	TE.09.40.03	x	x	x	x
	TE.09.40.04	x	x	x	x
AS.09.45	TE.09.45.03	x	x	x	x

AS.09.46	TE.09.46.03	x	x	x	x
Section 10 - Design Assurance					
AS.10.03	TE.10.03.02	x	x	x	x
Section 11 - Mitigation of Other Attacks					
	NONE				
Appendix C - Cryptographic Module Security Policy					
	As Required				

G.9 FSM, Security Policy, User Guidance and Crypto Officer Guidance Documentation

Applicable Levels:	*All*
Original Publishing Date:	*05/29/2002*
Effective Date:	*05/29/2002*
Last Modified Date:	*08/01/2016*
Relevant Assertions:	*General*
Relevant Test Requirements:	
Relevant Vendor Requirements:	

Question/Problem

May a CST laboratory create original documentation specified in FIPS 140-2? The specific documents in question are the Finite State Model (FSM), Security Policy, User Guidance and Crypto Officer Guidance.

Resolution

<u>**FSM and Security Policy:**</u>

A CST laboratory may take existing vendor documentation for an existing cryptographic module (post-design and post-development) and consolidate or reformat the existing information (from multiple sources) into a set format. If this occurs, NIST and CSE **shall** be notified of this when the validation report is submitted. Additional details for the individual documents are provided below.

        FSM: The vendor-provided documentation must readily provide a finite set of states, a finite set of inputs, a finite set of outputs, a mapping from the sets of inputs and states into the set of states (i.e., state transitions), and a mapping from the sets of inputs and states onto the set of outputs (i.e., an output function).

        Security Policy: The vendor-provided documentation must readily provide a precise specification of the security rules under which a cryptographic module must operate, including the security rules derived from the requirements of FIPS 140-2 and the additional security rules imposed by the vendor.

In addition, a CST laboratory must be able to show a mapping from the consolidated or reformatted FSM and/or Security Policy back the original vendor source documentation. The mapping(s) must be maintained by the CST laboratory as part of the validation records.

Consolidating and reformatting are defined as follows:

- The original source documents were prepared by the vendor (or a subcontractor to the vendor) and submitted to the CST laboratory with the cryptographic module.

- The CST laboratory extracts applicable technical statements from the original source documentation to be used in the FSM and/or Security Policy. The technical statements may **only** be reformatted to improve readability of the FSM and/or Security Policy. The content of the technical statements must not be altered.

- The CST laboratory may develop transitional statements in the FSM and/or Security Policy to improve readability. These transitional statements **shall** be specified as developed by the CST laboratory in the mapping.

User Guidance and Crypto Officer Guidance:

A CST laboratory may create User Guidance, Crypto Officer Guidance and other non-design related documentation for an existing cryptographic module (post-design and post-development). If this occurs, NIST and CSE **shall** be notified of this when the validation report is submitted.

Additional Comments

Source code information is considered vendor-provided documentation and may be used in the FSM and/or Security Policy.

G.10 Physical Security Testing for Re-validation from FIPS 140-1 to FIPS 140-2

Applicable Levels:	*All*
Original Publishing Date:	*03/29/2004*
Effective Date:	*03/29/2004*
Last Modified Date:	*03/29/2004*
Relevant Assertions:	*General*
Relevant Test Requirements:	
Relevant Vendor Requirements:	

Background

FIPS 140-2 IG G.2 specifies that all report submissions must include a separate physical security test report section for Levels 2, 3 or 4.

Question/Problem

Questions have been asked regarding re-validation test reports where a previous separate physical security test report may not have existed or evidence such as images, etc. had not been provided with the original validation test report. What should the CST laboratory provide if the physical security requirements have not changed?

Resolution

If a previous *separate* physical security test report did not exist for the module undergoing re-validation testing and the physical security features of the module have not changed, the CST laboratory must compile the physical security test evidence that has been maintained from their records from the original tested module and

create and submit a new *separate* physical security test report. If the records no longer exist because they were generated outside the period of the CST laboratories record retention period specified in the quality manual, then re-testing **shall** be required to provide such evidence. It is not required that a CST laboratory perform re-testing simply to create new photographic images that may not have been saved or generated during the original testing

Additional Comments

If the CST laboratory was not the original testing laboratory and therefore does not have access to the previous test records, then the module **shall** be re-tested to be able to provide such evidence. Without the prior records, the new CST laboratory cannot make a determination that the physical security has or has not changed.

G.11 Testing using Emulators and Simulators

Applicable Levels:	*All*
Original Publishing Date:	*09/12/2005*
Effective Date:	*09/12/2005*
Last Modified Date:	*09/12/2005*
Relevant Assertions:	*General*
Relevant Test Requirements:	
Relevant Vendor Requirements:	

Background

Vendors of cryptographic modules use independent, accredited Cryptographic and Security Testing (CST) laboratories to have their modules tested for conformance to the requirements of FIPS 140-2. Organizations wishing to have testing performed would contract with the laboratories for the required services. The Derived Test Requirements (DTR) document describes the methods that will be used by accredited laboratories to test whether the cryptographic module conforms to the requirements of FIPS 140-2. It includes detailed procedures, inspections, documentation and code reviews, and operational and physical tests that the tester must follow, and the expected results that must be achieved for the cryptographic module to satisfy its conformance to the FIPS PUB 140-2 requirements. These detailed methods are intended to provide a high degree of objectivity during the testing process and to ensure consistency across the accredited testing laboratories.

Definitions:

An **emulator** attempts to "model" or "mimic" the behavior of a cryptographic module. The correctness of the emulators' behavior is dependent on the inputs to the emulator and how the emulator was designed. It is not guaranteed that the actual behavior of the cryptographic module is identical, as many other variables may not be modeled correctly or with certainty.

A **simulator** exercises the actual module source code (e.g., VHDL code) prior to physical entry into the module (e.g., an FPGA or custom ASIC). From a behavioral perspective, the behavior of the source code within the simulator may be logically identical when placed into the module or instantiated into logic gates. However, many other variables exist that may alter the actual behavior (e.g. path delays, transformation errors, noise, environmental, etc.). It is not guaranteed that the actual behavior of the cryptographic module is identical, as many other variables may not be identified with certainty.

Question/Problem

May a CST laboratory tester use module emulation and/or simulation methods to perform cryptographic module testing?

Resolution

There are three broad areas of focus during the testing of a cryptographic module: operational testing of the module at the defined boundary of the module, algorithm testing and operational fault induction error testing.

1. Operational Testing

   Emulation or simulation is prohibited for the operational testing of a cryptographic module. Actual testing of the cryptographic module must be performed utilizing the defined ports and interfaces and services that a module provides.

2. Operational Fault Induction

   An emulator or simulator may be utilized for fault induction to test a cryptographic module's transition to error states as a complement to the already allowed source code review. Rationale must be provided for the applicable TE why a method does not exist to induce the actual module into the error state for testing.

3. Algorithm Testing

   Algorithm testing utilizing the defined ports and interfaces and services that a module provides is the preferred method. This method most clearly meets the requirements of IG 1.4.

   If this preferred method is not possible where the module's defined set of ports and interfaces and services do not allow access to internal algorithmic engines, two alternative methods may be utilized:

   a. A module may be modified by the CST laboratory for testing purposes to allow access to the algorithmic engines (e.g. test jig, test API), or

   b. A module simulator may be utilized.

   When submitting the algorithm test results to the CAVP, the actual operational environment on which the testing was performed must be specified (e.g. including modified module identification or simulation environment). When submitting the module test report to the CMVP, **AS.01.12** must include rationale explaining why the algorithm testing was not conducted on the actual cryptographic module.

   An emulator may not be used for algorithm testing.

G.12 Post-Validation Inquiries

Applicable Levels:	*All*
Original Publishing Date:	*01/26/2007*
Effective Date:	*01/26/2007*
Last Modified Date:	*08/07/2015*
Relevant Assertions:	*General*
Relevant Test Requirements:	
Relevant Vendor Requirements:	

Background

FIPS 140-2 conformance testing that is performed by the accredited Cryptographic and Security Testing (CST) laboratories and validation of those test results by NIST and CSE provide a level of assurance that a module conforms to the requirements of FIPS 140-2 and other underlying standards.

Once a module is validated and posted on the NIST CMVP web site, many parties review and scrutinize the merits of the validation. These parties may be potential procurers of the module, competitors, academics or others.

If a party performing a post-validation review believes that a conformance requirement of FIPS 140-2 has not been met and was not determined during testing or subsequent validation review, the party may submit an inquiry to the CMVP for review.

Question/Problem

What is the procedure and process for submitting an inquiry for review and how is the review performed? If a review is determined to have merit, what actions may be taken regarding the module's validation status?

Resolution

An *Official Request* must be submitted to the CMVP in writing with signature following the guidelines in IG G.1. If the requestor represents an organization, the official request must be on the organization's letterhead. The assertions must be objective and not subjective. The module must be identified by reference to the validation certificate number(s). The specific technical details must be identified and the relationship to the specific FIPS 140-2 Derived Test Requirements assertions must be identified. The request must be non-proprietary and not prevent further distribution by the CMVP.

The CMVP will distribute the unmodified official request to the CSTL that performed the conformance testing of the identified module. The CSTL may choose to include participation of the vendor of the identified module during its determination of the merits of the inquiry. Once the CSTL has completed its review, it will provide to the CMVP a response with rationale on the technical validity regarding the merits of the official request. The CSTL will state its position whether its review of the official request regarding the module:

1. is without merit and the validation of the module is unchanged.
2. has merit and the validation of the module is affected. The CSTL will further state its recommendations regarding the impact to the validation.

The CMVP will review the CSTLs position and rationale supporting its conclusion.

If the CMVP concurs that the official request is without merit, no further action is taken.

If the CMVP concurs that the official request has merit, a security risk assessment will be performed regarding the non-conformance issue.

G.13 Instructions for Validation Information Formatting

Applicable Levels:	*All*
Original Publishing Date:	*06/28/2007*
Effective Date:	*06/28/2007*
Last Modified Date:	*03/27/2018*
Relevant Assertions:	*General*
Relevant Test Requirements:	
Relevant Vendor Requirements:	

Question/Problem

How are the various fields in a FIPS 140-2 validation provided to the CMVP for validation?

Resolution

The CST laboratory **shall** use the CMVP supplied CRYPTIK tool to document the module test information. The test report information is presented to the CMVP for review and validation as indicated in IG G.2.

These instructions describe how the information **shall** be formatted to appear on the NIST CMVP validation web page via entry into CRYPTIK.

Laboratory Information

1. **Lab Name** - the name of the CST laboratory. Please include any registration marks or special characters.[1]

2. **NVLAP code** [**nnnnnn-n**] - the code assigned by NVLAP to the CST laboratory

Vendor Information

1. **Vendor Name** - the name of the vendor (including Corp., Inc., Ltd., etc.) that developed the cryptographic module. Please include any registration marks or special characters[1].

   Examples: **AcmeSecurity, Inc.**

   Acmeproducts(R), Ltd.

   AcmeSecurity, Inc. and Acmeproducts(R), Ltd.

   The FIPS 140-1 and FIPS 140-2 Vendor Listing is an alphabetical list of vendors who have implemented validated cryptographic modules. It is desirable that the vendor name be consistent on validation certificates issued for modules from the same vendor. The listing can be found at: http://csrc.nist.gov/groups/STM/cmvp/documents/140-1/1401vend.htm

2. **Address** - the street, building, post office box, suite, etc. components of the vendor's address

3. **City** - the city of the vendor's address

4. **State / Prov** - the state or province of the vendor's address

5. **Postal Code** - the postal code of the vendor's address

6. **Country** - the country of the vendor's address

7. **Web Site** - generally the vendor's main URL. Do not include the prefix http://

8. **Product Link** – a URL that may be specific to the module or products which utilize the module. Do not include the prefix http:// or duplicate the Web Site URL.

9. **POC1** - the primary vendor point of contact which may include phone number, fax number and email

10. **POC2** - the secondary vendor point of contact which may include phone number, fax number and email

Module Information

1. **Module Name(s)** - the complete name of the cryptographic module. Do not include the version number with the name unless by vendor choice. The name of the cryptographic module **shall** be consistent with IG 1.1 and the name found in the security policy and test report. Please include any registration marks or special characters[2].

   Examples: **Crypto Acceleration Token**

   Secure Cryptographic ToolKit™

   Best Crypto©

   If the test report represents multiple modules, list all module names.

---

[1] The special symbols may not translate to the _vendor.txt properly. The special symbol may be indicated as follows: (R) for ®, (C) for ©, (TM) for ™, etc.

[2] The special symbols may not translate to the _vendor.txt properly. The special symbol may be indicated as follows: (R) for ®, (C) for ©, (TM) for ™, etc.

Examples: **Crypto Sensor AM-5000 and AM-5010**
Crypto 8000 PCI, Crypto 9000 PCI and Crypto Plus++ PCI

2. **Hardware, Software and Firmware Versioning** - the specific versioning information representative of each of the crypto modules elements. This number **shall** be of sufficient level such that updates/upgrades/changes **shall** be reflected in a new version. For example, version 4 may not be sufficient if the releases are numbered 4.0, 4.1, 4.2, etc. The version number may also include letters, for example, 4.0a, 4.0b, 4.0c, etc. This **shall** include the version numbers for each element; hardware, software, and firmware, if applicable. Each elements version number (e.g. hardware, firmware, software) **shall** be separated by a semi-colon. If a module does not include an element, leave the field blank; do not enter "NA". The version numbers **shall** be the same as the ones found in the security policy. For example, hardware version: 4.2; software version: 4.0a.

If possible, a hardware version of a module **shall** represent all of the components of the module, included (**AS.01.08**) or excluded (**AS.01.09**). If there are any additional components, included (**AS.01.08**) or excluded (**AS.01.09**), that are inside the module boundary but are not within the scope of the hardware version then the module certificate **shall** list these additional components separately in the hardware version field. Brackets **shall** be used to group hardware versions with their corresponding components. If the module is a collection of different hardware components, included (**AS.01.08**) or excluded (**AS.01.09**), and does not contain a hardware version, then the module certificate **shall** list all of the components of the module in the hardware version field without referencing any hardware version.

If there are multiple modules listed on the certificate, or if there are multiple part numbers with different versions of firmware for example, brackets **shall** be used to clearly indicate the pairings between the versioning information and/or the module names.

Examples: **(Hardware Version: 4.2; Software Version: 4.0a; Hardware)**
Hardware module with software embedded within it.

(Hardware Versions[1]: 5.2 and 5.3, Build 3; Firmware Version: 2.45; Hardware)
Two different hardware modules, each with the same embedded firmware. All of the components in these hardware modules must be considered: included (**AS.01.08**) or excluded (**AS.01.09**).

(Hardware Versions: 5.2 [1] and 5.3 [2], Build 3; Firmware Versions: 2.45 [1] and 2.50 [2]; Hardware)
Two different hardware modules each with the specified version of embedded firmware.

(Hardware Version: 88X8868; Software Version: 1.0; **Software-Hybrid)**
Software hybrid module referencing the hardware and disjoint software components.

(Hardware Version: BN45; Firmware version 1.0; Software Version 2.0; Software-Hybrid)
Software hybrid module referencing the hardware and disjoint software versions. The hardware component also has firmware embedded within it.

(Hardware Version: 88X8686; Firmware Version 1.4; Firmware-Hybrid)
Firmware hybrid module referencing both the hardware and disjoint firmware versions.

Note the use of the commas, semi-colons and colons.

(Hardware Version: [XYZ1, XYZ2, and XYZ3 with components 1234, 1235, 1236] and [ZYX1, ZYX2 and ZYX3 with components 1234, 5123, 6123]; Firmware Version: 1.0; Hardware)
Hardware module contains multiple hardware versions that have additional corresponding components that are included (**AS.01.08**) or excluded (**AS.01.09**).

(Hardware Version: P/N 5432, 7654, and 4321; Firmware Version: 1.0; Hardware)

---

[1] Version will be changed to plural during the posting by the CMVP

Hardware module that is a collection of hardware components that are included (**AS.01.08**) or excluded (**AS.01.09**) rather than a versioned hardware module.

3. **PIV Certificate [#nnnn]** - When a module implements a validated PIV application, the application validation certificate type and number **shall** be included. Additional information relating to PIV versioning can be found in IG 1.18.

4. **Certificate Caveat** - This caveat may be modified or expanded by the CMVP during the validation process. Cryptographic modules may not have a caveat if the module only has a single FIPS approved mode of operation.

Examples: <no caveat>
The module can only be installed and operated in an approved mode of operation (i.e. FIPS mode).

When operated in FIPS mode
The module can be installed or operated in either an approved or non-approved mode of operation.

The <tamper evident seals> and <security devices> installed as indicated in the security policy
Installation of the referenced components required for the module to operate in an approved mode of operation.

When operated in FIPS mode and initialized to overall level 2 per security policy
The module can be initialized to operate at different overall levels.

    Example: A module can be initialized to either support level 2 role-based authentication or initialized to support only level 3 identity-based authentication.

When operated in FIPS mode with module [module name] validated to FIPS 140-2 under Cert. #xxxx operating in FIPS mode
The module's validation is bound to another validated cryptographic module.

    Example: A software cryptographic module which requires services from another validated software cryptographic module operating in the same operational environment. Application services are available from either module.

This module contains the embedded module [module name] validated to FIPS 140-2 under Cert. #xxxx operating in FIPS mode
If the module incorporates an embedded validated cryptographic module.

    Example: A software cryptographic module which is compiled with a privately linked validated software cryptographic module operating in the same operational environment. Application services are only available from the module indicated on the certificate.

    Example: A hardware cryptographic module which has embedded within its physical boundary a validated cryptographic module.

This validation entry is a non-security-relevant modification to Cert. #nnnn
If the lab submits a revalidation under scenario 1B. Please refer to IG G.8.

When operated in FIPS mode and utilizing a Trusted Path as specified in the security policy.
If a module uses the mechanisms of a trusted path at level 3 or higher to input or output plaintext CSPs. Please refer to IG 2.1.

The module generates cryptographic keys whose strengths are modified by available entropy
Please refer to IG 7.14.

No assurance of the minimum strength of generated keys
Please refer to IG 7.14.

When entropy is externally loaded, no assurance of the minimum strength of generated keys
Please refer to IG 7.14.

The output of the DRBG may not be used to generate keys
If the module implements a DRBG where the module does not meet IG 7.14 and IG 7.15 requirements for the entropy source.

The protocol(s) <TLS, SSH, …> shall not be used when operated in FIPS mode
If the module implements a KDF from NIST SP 800-135rev1 and this KDF has not been validated by the CAVP. Please refer to IG D.11.

5. **Type -** the module type is one of the following: **Hardware, Firmware, Software**, **Software-Hybrid** or **Firmware-Hybrid**. If a module is hardware with embedded software and/or firmware, the modules type is simply labeled Hardware.

6. **Overall Level [n]** – the overall level of the crypto module. This value is the _lowest_ value of the individual levels.

7. **Section Level(s) [n]** - for each of the 11 areas, include the specific level. For FIPS 140-2, the Operating System security level, the physical security level and Mitigation of Other Attacks level may not be applicable and if so, **shall** be marked as N/A.

   If a module meets level 3 physical security and also has been tested for EFP and/or EFT, this **shall** be annotated on the certificate as: **Level 3** +**EFP** or +**EFT** or +**EFP/EFT**

   Note: If FIPS 140-2 Section 4.5 is level 3 with EFP/EPT, this is selected in CRYPTIK by selecting level 3 for FIPS 140-2 Section 4.5 and selection of the optional EFP/EFT button. CRYPTIK will then present the appropriate set of assessments. However, the generated *draft certificate* and *_vendor.txt* will not reflect the optional EFP/EFT annotation. Currently this must be added manually during validation posting.

8. **Operational Environment** - the specific operational environment(s) or configuration(s) that was employed during testing by the CST laboratory **shall** be specified for all module types. (e.g. software, firmware, hardware and hybrid). This **shall** match the information in the test report in **AS.01.08**. The operational environment includes both the operating system(s) and the tested platform(s).

   For a *software* cryptographic module at security level 1, the caveat "(single-user mode)" **shall** be included. For Java applets, the Java environment (JRE, JVM) version **shall** be specified for all security levels. For multiple operating environment entries, separate each with a semi-colon; do not use "and".

   Examples: **Microsoft Windows XP with SP2 running on a Dell Optiplex Model 4567;**

   Sun Solaris Version 2.6SE running on a Sun Ultra SPARC-1 workstation;

   Microsoft Windows XP with SP2 running on an HP Pavilon 4.5;

   HP-UX 11.23 running on an IBM RISC 6000RB2 (single-user mode)

   The following example for a *firmware* cryptographic module;

   Example: **BlackBerry® 7230 with BlackBerry OS® Versions 3.8, 4.0 and 4.1**

If the *firmware* module's physical security meets FIPS 140-2 Section 4.5 levels 2, 3 or 4, the hardware platform **shall** include applicable specific versioning information.

Example: **Crypto Unit (Hardware Version: 1.0) with Little OS® Version 3.7b**

The following example for a *software-hybrid* cryptographic module;

Example: **Debian GNU/Linux 4.0 (Linux kernel 2.6.17.13) running on 4402-A ViPr Desktop Terminal (single-user mode)**

The following example for a *firmware-hybrid* cryptographic module; the certificate **shall** specify the operating environment (hardware platform and operating system) that was used for testing.

Example: **BlackBerry 8700c with BlackBerry OS Version 4.2**

The operational environment includes both the operating system(s) and the tested platform(s). The operating system may also represent virtual environments. Virtual environments are run by computer software, firmware or hardware called a hypervisor. Native hypervisors run directly on the host computer. Hosted hypervisors run on a conventional operating system.

- For a Type 1 (or native) hypervisor, the OE listing **shall** include the platform, guest OS and hypervisor using the following format:

   Operational Environment: Tested as meeting Level 1 with *<Guest OS>* on *<hypervisor>* running on *<platform>*

   An example is "Windows XP on VMWare ESX 5 running on a Dell Optiplex 5460

- For a Type 2 (or hosted) hypervisor, the OE listing **shall** include the platform, guest OS, hypervisor and host OS using the following format:

   Operational Environment: Tested as meeting Level 1 with *<Guest OS>* on *<hypervisor>* on *<Host OS>* running on *<platform>*

   An example is "Windows 7 on Oracle VM VirtualBox on Oracle Solaris 11 running on a HP Model 20

If this field is not applicable, mark the field as N/A.

9. **FIPS Approved Algorithms** - the approved security functions included in the cryptographic module and utilized by the modules callable services or internal functions. The security function is listed and then the applicable algorithm Certificate number in parentheses. Do NOT include the modes or key lengths (e.g., ECB, CBC; 128 bits). All algorithm entries must be separated by semi-colons. The security functions **shall** be listed in alphabetical order using the official CAVP security function name.

   If a module contains within it or is bound to an already validated cryptographic module, all approved security functions that are used by the modules callable services and internal functions **shall** be annotated on the certificate (e.g. both those within the embedded/bound module and in addition to the embedding/binding module). Algorithms that are never called **shall not** be listed on the certificate. An algorithm that can only be called by a service that performs the self-tests also **shall not** be listed on the certificate; however, the module's security policy **shall** have an entry for the corresponding self-test and explain that this algorithm can only be executed when running a self-test.

   The algorithm **shall** meet all three (3) conditions to be listed as FIPS approved:

   1.   an approved security function as specified in FIPS 140-2 Annexes A, C or D and validated by the CAVP or vendor affirmed per CMVP implementation guidance;

   2.   meet all requirements of FIPS 140-2 (KAT, etc.); and

   3.   used in at least one FIPS approved cryptographic function or service for that cryptographic algorithm in a FIPS approved mode of operation.

   Examples: **AES (Cert. #1880);**

AES-CBC-CS[1] **(vendor affirmed);**

CKG[2] **(vendor affirmed);**

CVL[3] **(Cert. #4);**

DRBG[4] **(Cert. #12);**

DSA[5] **(Cert. #200);**

ECDSA[6] **(Cert. #100);**

HMAC[7] **(Cert. #23);**

KAS[8] **(Cert. #33);**

KAS[9] **(SP 800-56Arev2, vendor affirmed);**

KAS[10] **(SP 800-56Arev2 with CVL Certs. #24 and #32, vendor affirmed);**

KAS[11] **(SP 800-56B, vendor affirmed)**

KBKDF[12] **(Cert. #2);**

KTS[13] **(vendor affirmed);**

PBKDF[14] **(vendor affirmed);**

RSA[15] **(Cert. #133);**

RSA[16] **(SHA-3 Cert. #55, vendor affirmed);**

SHA-3[17] **(Cert. #55);**

SHS (Cert. #23);

Skipjack[18] **(Cert. #45);**

Triple-DES (Certs. #78 and #122);

Triple-DES MAC[19] **(Triple-DES Cert. #78, vendor affirmed)**

For multiple certificate entries, the term "Cert" **shall** be pluralized (i.e., Certs), an "and" **shall** be placed between the last two certificate numbers and there **shall** be a "#" in front of each number.

Examples: **Triple-DES (Certs. #118 and #133);**
              SHS (Certs. #103, #115 and #119)

If the module supports symmetric key wrapping, one of the following annotations **shall** be used, depending on the approved wrapping algorithm:

---

[1] SP 800-38A Addendum

[2] Cryptographic Key Generation; SP 800-133 and IG 7.8

[3] Component Validation List; CAVP CVL

[4] Deterministic Random Bit Generator; SP 800-90A

[5] FIPS 186-2 or FIPS 186-4

[6] FIPS 186-2 or FIPS 186-4

[7] Includes Truncated HMACs per IG A.8

[8] Key Agreement Scheme; SP 800-56A (original publication)

[9] Key Agreement Scheme; vendor affirmed to SP 800-56A revision 2

[10] Key Agreement Scheme; vendor affirmed to SP 800-56A revision 2

[11] Key Agreement Scheme; vendor affirmed to SP 800-56B

[12] Key Based Key Derivation Function; SP 800-108

[13] Key Transport Scheme; SP 800-56B

[14] Password Based Key Derivation Function; SP 800-132

[15] FIPS 186-2 or FIPS 186-4

[16] FIPS 186-4 and FIPS 202. RSA signatures with only the SHA-3 hash functions.

[17] FIPS 202

[18] Only decryption is approved for Skipjack

[19] **Shall** specify the underlying Triple-DES algorithm certificate number with the "vendor affirmed" caveat.

KTS (Triple-DES Cert. #50; key establishment methodology provides 112 bits of encryption strength) – an implementation has been tested for its compliance with three-key Triple-DES TKW and this mode of the Triple-DES is used for key wrapping. Triple-DES cert. #50 **shall** be listed separately on the approved line.

KTS (AES Cert. #100) – an implementation has been tested for its compliance with AES KW and/or AES KWP and this mode of AES is used for key wrapping. AES cert. #100 **shall** be listed separately on the approved line.

KTS (AES Cert. #200) - has been tested for its compliance with AES GCM (or any other authenticated encryption mode) and this mode of AES is used for key wrapping. AES cert. #200 **shall** be listed separately on the approved line.

KTS (AES Cert. #300) - has been tested for its compliance with both AES KW and AES GCM and each of these two modes of AES may be used for key wrapping. The AES cert. #300 **shall** be listed separately on the approved line. Each tested AES mode, KW and GCM (and any other) will be shown in the AES algorithm certificate. The security policy **shall** explain how each applicable mode of AES is used for key wrapping.

KTS (AES Cert. #700 and HMAC Cert. #200) - Example of CAVP testing of disjoint AES encryption and HMAC authentication with appropriate strength. AES cert. #700 and HMAC cert. #200 **shall** be listed separately on the approved line.

KTS (AES Cert. #750 and HMAC Cert. #250; key establishment methodology provides 192 bits of encryption strength) - Example of CAVP testing of disjoint AES encryption and HMAC authentication where an AES wrapping key may be of lower length than wrapped key. AES cert. #700 and HMAC cert. #250 **shall** be listed separately on the approved line.

KTS (AES Cert. #300 and HMAC Cert. #355; key establishment methodology provides 128 or 192 bits of encryption strength) – a combination of AES in any mode and message authentication using HMAC is used for key wrapping. There is a range of AES key lengths. AES cert. #300 and HMAC cert. #355 **shall** be listed separately on the approved line.

KTS (AES Cert. #400 and AES Cert. #10; key establishment methodology provides between 128 and 256 bits of encryption strength) - a combination of AES in any mode and message authentication using AES CMAC is used for key wrapping. AES certs. #10 and #400 **shall** be listed separately on the approved line.

NOTE 1: The AES or the Triple-DES algorithm certificate will provide information on the length of the wrapping key. To make a decision if this length is sufficient to avoid adding a strength caveat, one has to know the range of the possible lengths of the wrapped keys. **AS.07.19** requires that the wrapping key used in key transport be equal or of greater strength than the wrapped key. If the strength of the largest key that can be established by a cryptographic module is greater than the comparable strength of the implemented key establishment method, then the module certificate and security policy **shall** be annotated with, in addition to the other required caveats, the caveat "**(key establishment methodology provides xx bits of encryption strength)**"[1] for that key establishment method as allowed in IG 7.5 – *Strength of Key Establishment Methods*. No strength caveat is required if the wrapping key used in key transport be equal or of greater strength than the wrapped key. This applies to both approved KTS, or allowed key establishment methods (see section 10 of this IG G.13 for allowed key establishment methods). A similar caveat is used when a key is established using a key agreement protocol that might cause the resulting cryptographic strength of the key to be less than the key length in bits.

NOTE 2: The strength of an HMAC key and the size of the hash output are not reflected in the computation of the equivalent encryption strength.

---

[1] While this caveat only has a single encryption strength claimed, other examples included in this IG G.13 indicate that the strength caveat may have a range, depending on the key sizes used for the key establishment methodology.

10. **Allowed algorithms**[1] - cryptographic algorithms that are not approved but are allowed to be used in a FIPS approved mode of operation.

All allowed algorithms **shall** be identified in the security policy and listed on the validation certificate. Allowed algorithms **shall** be listed in alphabetical order on the certificate.

All non-FIPS approved and not allowed algorithms **shall** be listed in the security policy but NOT on the certificate. A non-FIPS approved implementation may exist for what appears to be an approved algorithm where a CAVP validation or the requirements of FIPS 140-2 (e.g. self-test) are not met. These non-FIPS approved implementations are considered non-approved and non-compliant and **shall** be described in the security policy as "*non-compliant*" so that it is clear the algorithm implementation **shall** not be used in an approved mode of operation.

---

[1] Through June 30, 2017, section 10 of this IG (Allowed algorithms) will be labelled *Other algorithms* on the certificate and will include allowed and non-approved algorithms. Starting July 1, 2017, section 10 of this IG (Allowed algorithms) will be labelled *Allowed algorithms* and will only include allowed algorithms. Starting July 1, 2017, non-approved and non-allowed algorithms **shall** only be listed in the security policy.

Examples: **AES[1] (Cert. #300, key unwrapping);**

                      Diffie-Hellman[2] (shared secret computation);

                      Diffie-Hellman[3] (key agreement);

                      Diffie-Hellman[4] (CVL Certs. #5 and #6, key agreement);

                      EC Diffie-Hellman[5] (key agreement);

                      EC Diffie-Hellman[6] (CVL Cert. #4 with SP 800-56C, vendor affirmed, key agreement);

                      MD5[7];

                      RSA[8] (key unwrapping);

                      RSA[9] (key wrapping);

                      RSA[10] (CVL Cert. #10, key wrapping);

                      Triple-DES[11] (key unwrapping);

For the non-FIPS approved key establishment schemes refer to IG's D.8 and D.9.

For algorithm implementations that have both approved and non-approved and not allowed (e.g. RSA) components, the approved component **shall** be listed on the FIPS approved line and the non-approved and not allowed component **shall** be listed only in the security policy. The security policy **shall** indicate all uses of the algorithm.

NOTE: Encryption strengths represented on a validation entry are based on algorithm key sizes in bits *only*. As indicated above the calculation of the encryption strength based on key size is performed per IG

---

[1] This is the allowed but non-SP-800-38F-compliant key unwrapping, where the key used in key transport is of equal or greater strength than the unwrapped key and therefore the strength caveat is not required.

[2] Only the untested shared secret computation primitive is implemented. This is allowed in an approved mode of operation.

[3] No claim of compliance with SP 800-56A DLC or KDF computation. allowed in an approved mode of operation.

[4] Composite of two disjoint tested components (DLC and KDF) which forms key agreement. Allowed in an approved mode of operation but the composite is not tested by the CAVP.

[5] No claim of compliance with SP 800-56A DLC or KDF computation. allowed in an approved mode of operation. **Shall** use the "EC Diffie-Hellman" annotation not the ECDH notation.

[6] Composite of two disjoint components (tested DLC and vendor-affirmed KDF) which forms key agreement. Allowed in an approved mode of operation but the composite is not tested by the CAVP.

[7] May be allowed in an approved mode of operation when used as part of an approved key transport scheme (e.g. SSL v3.1) where no security is provided by the algorithm.

[8] The module does not support RSA key wrapping but does employ RSA key unwrapping with no claim of compliance with any testable component of SP 800-56B. Allowed in an approved mode of operation.

[9] No claim of compliance with any testable component of SP 800-56B. Allowed in an approved mode of operation. If the module supports both RSA key wrapping and unwrapping in this way, or just key wrapping alone, the certificate **shall** only include a "key wrapping" entry without a separate "key unwrapping" entry.

[10] A component of SP 800-56B is tested but the composite is not tested by the CAVP. Allowed in an approved mode of operation.

[11] This is the allowed but non-SP-800-38F-compliant key unwrapping, where the key used in key transport is of equal or greater strength than the unwrapped key and therefore the strength caveat is not required.

7.5. The effective encryption strength may be less depending upon the amount of available entropy. See IG 7.14, IG 7.15 and this IG for additional guidance and applicable caveats.

In the following key establishment examples, the strength caveat *does* apply (i.e., the security strength of the key establishment scheme implemented by the module **can be** less than that of the agreed or wrapped key).

If the module supports, for a particular key establishment method, a single strength, then the caveat **shall** state the strength provided by the keys.

Examples: **Diffie-Hellman (key agreement; key establishment methodology provides 112 bits of encryption strength)**

RSA (key wrapping; key establishment methodology provides 112 bits of encryption strength)

RSA[1] (key unwrapping; key establishment methodology provides 112 bits of encryption strength)

EC Diffie-Hellman[2] (shared secret computation provides 192 bits of encryption strength)

If a module *only* implements two specific key sizes for Diffie-Hellman, then:

Diffie-Hellman (key agreement; key establishment methodology provides 112 or 128 bits of encryption strength)

If a module implements a key establishment scheme with several key sizes for Diffie-Hellman, RSA or EC Diffie-Hellman then only the range end points are indicated:

Diffie-Hellman (key agreement; key establishment methodology provides between 112 and 256 bits of encryption strength)

RSA (key wrapping; key establishment methodology provides between 130 and 180 bits of encryption strength)

EC Diffie-Hellman (key agreement; key establishment methodology provides between 112 and 256 bits of encryption strength)

If a module implements a key establishment scheme of several key sizes and also less than 112 bits of strength, then only the approved range end points are indicated.

Diffie-Hellman (key agreement; key establishment methodology provides between 112 and 256 bits of encryption strength)

If a module supports a key agreement algorithm such that the shared secret computation portion of the key agreement is tested for its compliance with **SP 800-56A** and issued a CVL certificate, then an example of the certificate annotation would be:

EC Diffie-Hellman (CVL Cert. #17, key agreement; key establishment methodology provides between 128 and 256 bits of encryption strength)

If, in addition, the module states compliance with another part of the key agreement protocol, then this also **shall** be caveated in the certificate. For example:

---

[1] The module does not support RSA key wrapping but does employ RSA key unwrapping with 2048-bit modulus.

[2] Do not claim "key agreement" when documenting a shared secret computation (not the full key-agreement scheme) using the FFC or ECC technology.

> **Diffie-Hellman (CVL Cert. #3 with SP 800-56C, vendor affirmed, key agreement; key establishment methodology provides between 112 and 150 bits of encryption strength)**
>
> **EC Diffie-Hellman (CVL Cert. #17 with CVL Cert. #6, key agreement; key establishment methodology provides between 112 and 192 bits of encryption strength)**

If the module supports only a portion of the key establishment scheme and this portion was tested for its compliance with its associated standard (i.e. **SP 800-56A**, **SP 800-56B**, **SP 800-135rev1**, etc.) and issued a CVL certificate, then the FIPS approved Algorithms line would include the CVL certificate but the *Allowed algorithms* line *would not* include the key establishment scheme, since the CVL certificate covers the implementation. For example, if the module only implements the shared secret computation of the Diffie-Hellman scheme, and this was CVL certified to comply with **SP 800-56A**, then the CVL certificate would be listed on the approved algorithms line but the Diffie-Hellman would *not* be listed in the Other Algorithm line.

If the module supports a key establishment scheme such that part of the scheme has a CVL certificate, but the CVL certificate does not cover all of the curves or key sizes that the scheme implements, then these would be split into separate entries on the certificate - one with the approved CVL reference, and the other without. For example:

> **EC Diffie-Hellman (CVL Cert. #842, key agreement; key establishment methodology provides 128 or 192 bits of encryption strength); EC Diffie-Hellman (key agreement; key establishment methodology provides 112 or 256 bits of encryption strength)**

If the module supports the key unwrapping algorithms that are not compliant with **SP 800-38F** then this **shall** be annotated in the certificate. For example:

> **AES (Cert. #300, key unwrapping; key establishment methodology provides 128 or 192 bits of encryption strength)**
>
> **Triple-DES (Cert. #114, key unwrapping; key establishment methodology provides 112 bits of encryption strength)**

If AES MAC is implemented for OTAR, it **shall** be specified as:

> **AES MAC (AES Cert. #2, vendor affirmed; P25 AES OTAR)**

All other uses of AES MAC are non-compliant and **shall** only be listed in the security policy (as non-compliant).

Note: In all cases, the CMVP report reviewer must ascertain the correctness of the added caveat(s) and the most accurate wording and the best interpretation to give to the Federal users.

If the Allowed algorithms field is not applicable, mark the field as N/A.

For non-FIPS approved algorithms that have names similar to approved security functions, they are considered non-approved and non-compliant and **shall** be listed in the security policy but NOT on the certificate. They **shall** be described as "non-compliant" in the security policy so that it is clear the algorithm implementation **shall not** be used in the approved mode of operation.

Non-deterministic random number generators used to seed a DRBG **shall** be called NDRNG and listed on the Allowed algorithms line.

11. **Embodiment Type** - the cryptographic module **shall** be specified as one of the three types: **Multi-chip Standalone**, **Multi-chip Embedded**, or **Single-chip**.

G.14 Validation of Transitioning Cryptographic Algorithms and Key Lengths

Applicable Levels:	*All*
Original Publishing Date:	*04/23/2012*
Effective Date:	*01/01/2011*
Last Modified Date:	*05/10/2017*
Relevant Assertions:	*AS.01.12*
Relevant Test Requirements:	*TE.01.12.01-02*
Relevant Vendor Requirements:	*VE.01.12.01-02*

Background

At the start of the 21st century, the National Institute of Standards and Technology (NIST) began the task of providing cryptographic key management guidance, which includes defining and implementing appropriate key management procedures, using algorithms that adequately protect sensitive information, and planning ahead for possible changes in the use of cryptography because of algorithm breaks or the availability of more powerful computing techniques. SP 800-57, Part 1 was the first document produced in this effort, and includes a general approach for transitioning from one algorithm or key length to another. SP 800-131Arev1 provides more specific guidance for transitions to the use of stronger cryptographic keys and more robust algorithms.

Question/Problem

How will the validation of the cryptographic algorithms and cryptographic modules be affected during the transition as specified in SP 800-131Arev1?

Resolution

1. Useful Terms

    1.1 **New Validations, Already Validated Implementations and Revalidations**

    The CAVP and CMVP, along with the accredited CST laboratories, have been in existence since 1995. Consequently, a large number of implementations have been tested and validated under these programs, and the number of new implementations that are validated continues to increase every year. The CMVP conducts revalidations of already-validated module implementations whenever changes are made to the module implementations or when new operational environments are added to an existing validation. These changes may require the validation of new implementations and/or the retesting of already-validated algorithm implementations.

    - *New Implementations* refers to the cryptographic algorithms or modules that have not been validated by the CAVP or CMVP, respectively.

        For algorithm implementations, new implementations are the algorithm implementations that are to be tested or are currently under test by an accredited CST laboratory for which the algorithm test results will be submitted to the CAVP.

        For cryptographic modules, new implementations refer to cryptographic modules that are either new modules or the revalidation of modules under IG G.8 Scenarios 3 and 5. These modules are either not yet tested, or are currently under test by an accredited CST laboratory for which the test report will be submitted to CMVP.

    - *Already-Validated Implementations* are algorithm or module implementations that have already been tested by a CST laboratory and validated by the CAVP or CMVP.

        Cryptographic module validations reference at least one approved algorithm implementation. These references are to algorithms that have been validated by the CAVP, algorithms for which standards may not have existed at the time of the CMVP validation, or algorithms for which CAVP validation testing was not available at the time of the module validation. Some algorithms

in NIST-Recommendations may appear on a CMVP validation certificate as "non-approved, but allowed for use in an approved mode of operation." In addition, the level of specificity found on a module validation-entry has changed over the life of the CMVP program, as standards and testing methods emerged.

1.2 Terms Used in SP 800-131A

The use-categorization terms "**acceptable**", "**deprecated**" and "**legacy use**" are used in **SP 800-131A** to address the use of cryptographic algorithms and key lengths. The term **"restricted"** was also used in **SP 800-131A** since this category existed at the time **SP 800-131A** was published. It is no longer used for any purpose.

- **Acceptable** is used to mean that the algorithm and key length is safe to use; no security risk is currently known.

- **Deprecated** means that the use of the algorithm and key length is allowed, but the user must accept some risk.

- **Legacy-use** means that the algorithm or key length may be used to process already-protected information (e.g., to decrypt ciphertext data or to verify a digital signature) that was protected using an algorithm or key length that has since been deprecated, restricted or disallowed for applying cryptographic protection.

An algorithm or key length is considered to be disallowed (i.e., no longer approved) for its purpose if it is not classified as acceptable, deprecated or allowed for legacy-use.

2. General Validation Strategy

The general validation strategy to be used by the CAVP and CMVP for new and already-validated implementations is the following:

- New implementations: When applied to cryptographic algorithms, the dates in the tables of **SP 800-131A** refer to the algorithm's validation date that is assigned by the CAVP.

  When applied to cryptographic modules, the dates in the tables refer to the dates of the CST laboratory's initial submission of a module test report to the CMVP for validation.

  Security policies for new module implementations are discussed in Section 5.

- Already-validated implementations: As resources permit, the CAVP and CMVP will review these implementations and their validations for compliance with the new security requirements as stated in **SP 800-131A** when a transition date occurs.

  The CAVP will review the algorithm validations to determine if a validated algorithm or a key length is disallowed in **SP 800-131A**. If a complete algorithm validation is disallowed, the CAVP will revoke the algorithm validation; references to these revoked validations will continue to be available for historical purposes. If only parts of a validation are disallowed (e.g., one of the validated key lengths is disallowed), the validation listing will be annotated to indicate the disallowed parts of the validation.

  The CMVP will review the list of module validations and take the appropriate actions, based on the module's provided algorithm validation references.

  o If an algorithm validation is revoked by the CAVP, the module's validation reference will be removed from the "FIPS Approved algorithms" line. If the algorithm could still be used in the non-approved mode, it **shall** be listed in the security policy.

  o References to revised algorithm validations will remain unchanged; i.e., if only part of the validation is disallowed by the CAVP, the certificate reference will not be revised.

  o References to other algorithms will be changed only if sufficient information was provided that would allow modification. The information provided at the time of module validation and presented on the validation-list entry may be insufficient to determine whether a module continues to satisfy all of the new security requirements or whether the module's validation

continues to be valid. Therefore, the CMVP will flag validations that have been partially revoked by the CAVP; this flag may be removed by the voluntary submission of an appropriately-updated security policy by the vendor that addresses the transition issue.

o   If all algorithm validations for a module are revoked, the module validation will be revoked, and the validation listing will be annotated to indicate the revocation. For historical purposes, the annotated entry in the validation listing will be retained.

o   It is the user's responsibility to determine that the algorithms and keys lengths utilized by their system are in compliance with the requirements of **SP 800-131A**. All questions regarding the implementation and/or use of any module located on the CMVP module validation lists should first be directed to the appropriate vendor point-of-contact (listed for each entry).

o   As appropriate, the CMVP will only modify the module validation entry information; however, the security policy provided with each module validation will not be modified except by vendor request. The CMVP encourages vendors to submit updated security policies with appropriate revisions. Updated security policies may be submitted directly to the CMVP; the updated policies will be placed on the CMVP web site, and the updated security policy and the validation listing for the associated module will be annotated to indicate the update.

o   Cryptographic modules revalidated under scenarios 1 and 4 of IG G.8 will be treated as already-validated implementations.

3. **Validation of Cryptographic Algorithms and Cryptographic Modules by Use Categorization**

SP 800-131A addresses the use of cryptographic algorithms and key lengths during given time periods, categorizing them as acceptable, deprecated, legacy-use and disallowed. These categorizations affect the validation of new implementations and the status of already-validated implementations.

Cryptographic algorithms and key lengths that are categorized as acceptable, deprecated or legacy-use **shall** be validated for Federal government use.

3.1   Acceptable

New algorithm validation submissions and new module validation submissions will be accepted by the CAVP or CMVP, respectively, through December 31st of the end-year indicated, if an end-year is provided, or with no date restriction if an end-year is not provided.

Already-validated algorithm or module implementations will remain valid during this period.

No additional requirements are placed on the cryptographic modules revalidated under scenarios 1, 2 and 4 of IG G.8.

3.2   Deprecated

In general, new algorithm or module validation submissions will be accepted for validation by the CAVP or CMVP, respectively, through December 31st of the end-year for the deprecation period.

Already-validated algorithm and module implementations will remain valid through December 31st of the end-year of the deprecation period.

3.3 Legacy-Use

The legacy-use categorization is intended to allow the processing of already-protected information – information for which the protection was originally applied using an algorithm or key length that was acceptable, or deprecated at the time of applying the protection, but is now disallowed for that purpose.

New algorithm and module validation submissions will be accepted for validation by the CAVP or CMVP, respectively, until disallowed.

Algorithm and module validations for already-validated implementations will remain valid for processing already-protected information only.

Example 1: After December 31, 2015, two-key Triple DES decryption can be validated, while two-key Triple DES encryption will not (see Section 4.5).

Example 2: After December 31, 2013, implementations that verify digital signatures that were generated using 1024 bit RSA keys can continue to be validated, even though the generation of signatures using this key length will no longer be validated.

3.4 **Disallowed Algorithms and Key Lengths**

New module validation submissions and submissions for the revalidation of modules containing only algorithms and key lengths that are disallowed for their purpose will not be accepted for validation by the CMVP; submissions containing one or more algorithms and/or key lengths categorized as acceptable, deprecated, or legacy-use will continue to be accepted for validation.

4. **The Remaining Use of FIPS 186-2**

Implementations of domain parameter generation, key pair generation and digital signature generation as specified in FIPS 186-2 are *no longer validated* by the CAVP or CMVP.

Cryptographic algorithm and module implementations that perform domain parameter validation, public key validation and digital signature verification may be tested by the CST labs for conformance to FIPS 186-2 (or parts of FIPS 186-2) and submitted for validation, subject to the following conditions.

- **CAVP**: The CAVP will accept test results from the CST labs of cryptographic algorithm implementations of FIPS 186-2 (or parts of FIPS 186-2) that contain testable key lengths permitted by FIPS 186-2 that are categorized as either acceptable or legacy-use as specified in SP 800-131Arev1.

- **CMVP**: New modules (3SUB and 5SUB submissions) and already-validated modules containing digital signature processes conforming to FIPS 186-2 that have algorithm validations issued by the CAVP may be validated or revalidated, as appropriate.

5. **Documentation Requirements for CMVP Validations**

Module security policies submitted for new validations and (optional) updated security policies provided for already-validated implementations **shall** either include or make a reference to the transition tables that will be available at the CMVP Web site (http://csrc.nist.gov/groups/STM/cmvp/). The data in the tables will inform users of the risks associated with using a particular algorithm and a given key length.

This documentation requirement applies to all new validation submissions made three months after the publication of this IG. This requirement also applies to revalidation submissions, Scenarios 3 and 5 of IG G.8.

G.15 moved to W.2

G.16 Requesting an Invoice Before Submitting a Report

Applicable Levels:	*All*
Original Publishing Date:	*05/10/2016*
Effective Date:	*05/10/2016*
Last Modified Date:	
Relevant Assertions:	
Relevant Test Requirements:	
Relevant Vendor Requirements:	

Background

NIST Cost Recovery (CR) is currently levied on all 1A, 1B, 3 and 5 submissions. Currently, the CR process is initiated upon receipt of the report submission and typically adds an average of 60 days to the validation process.

Question/Problem

Can the CR process be initiated before the report submission?

Resolution

The following requirements **shall** be met in order to initiate the CR process before the report submission.

- The lab sends an IUTA indicating the correct number of modules, overall security level and submission type. The IUTA can be submitted without requesting that the module be placed on the Implementation Under Test (IUT) list. The IUTA must be successfully processed by the NIST CMVP automated system. (This includes 1A and 1B submission types.) When the submission is successfully processed, the lab will receive an automated response, "*Thank you for your submission*".

- At any time after the lab receives the automated response to the IUTA, the lab has the option to send an IUTB to initiate the CR process before submitting the report. When the IUTB is successfully processed, the lab will receive an automated response, "*Thank you for your request. The cost recovery process for this submission has been initiated*." Changes to the overall security level and submission type will not be accepted.

  - If the lab sends an IUTB for a 1SUB, it is assumed that it is a 1A or 1B and CR applies.
  - If the lab sends an IUTB and then needs to cancel the invoice, the lab must send an IUTC. When the IUTC is successfully processed, the lab will receive the automated response, "*Your request has been received and will be processed. If there are any issues in cancelling the invoice, you will be notified.*"
    - Only unpaid invoices can be cancelled.
  - No files are required for an IUTB or IUTC. Only a properly formatted subject line is required.

- When the cost recovery process starts, no changes to the Security Level or Submission Type will be accepted.

- When the invoice is paid, there are no refunds regardless of when the CR process is initiated.

- If a report has not been received by 90 days after the IUTB was accepted, the module will be moved to On Hold and removed from the IUT list. The module can be automatically removed from On Hold and placed on the Modules In Process (MIP) list by sending the report.

If the lab chooses to not send an IUTB, the CR process will initiate upon receiving the report submission.

G.17 Remote Testing for Software Modules

Applicable Levels:	*1 and 2*
Original Publishing Date:	*08/07/2017*
Effective Date:	*08/07/2017*
Last Modified Date:	
Relevant Assertions:	*General*
Relevant Test Requirements:	
Relevant Vendor Requirements:	

Background

Section 4.1.2 of Cryptographic Module Validation Program Management Manual (http://csrc.nist.gov/groups/STM/cmvp/documents/CMVPMM.pdf, Last update 07 Mar 2017) states that the testing of the Cryptographic Module can be performed either by providing the cryptographic module to the laboratory or preparing it for testing at the vendor's facility. This testing requirement is clear for a hardware module which has self-contained operational environment and can only be physically located either in the laboratory or at the vendor's facility for testing. For a software cryptographic module that relies on an operating environment outside of the module's logical boundary, the CMVP Management Manual is unclear whether it is permissible for the testing to be performed by providing the compiled binary code as software cryptographic module to the laboratory but preparing its operating environment for testing at the vendor's facility.

Modern day networking enables the testing and deployment of software remotely on a General-Purpose Computer (GPC) that is either not necessary or even not possible to be physically accessible by the human operator. A vendor may have satellite development centers or remotely working developers who test their software on GPCs located elsewhere via the corporation private intranet. Laboratory personnel conducting testing at the vendor's facility may still end up utilizing an operating environment that the tester does not have physical access to and control over. Traveling to the vendor's facility and then performing the test on its remote operating environment not only costs time and money but also does not make a technical difference on the test results in comparison to performing the test on the same remote operating environment directly from the laboratory, as long as the network connection (e.g. VPN connection, SSH connection) between the local test console and the remote test operating environment provides the same level of security as testing onsite. The operational testing requirements of FIPS 140-2 should be able to use these technologies in a way that is practical and secure for all parties involved. This IG is intended to address the needs for testing a software module on a remote operating environment while obtaining the equivalent assurance as if the test were performed at the vendor's facility.

Question/Problem

Under what conditions can a software cryptographic module be tested on a remote operating environment?

Resolution

A software cryptographic module **shall** only be tested on a remote operating environment if the following conditions are met:
1. A software cryptographic module is provided by the vendor to the laboratory and its boundary and version is verified on screen against the Security Policy.
2. The network access to a remote test operating environment **shall** be authorized and controlled by the vendor. A 3rd party cloud system such as Amazon Web Services **shall** not be used. The tester must have control of the operating environment during testing. The lab's network must be connected to the vendor's network via a secure VPN connection or SSH connection. If a tester wishes to work offsite

per Lab Bulletin LB-96-2016 then the tester must connect to the lab's network before connecting to the vendor's network to test the module.

3. The operating environment information (e.g. operating system name and version, processor family, hardware platform model) as required by IG G.13 **shall** be obtained and verified against the operating environment information listed on the CAVP algorithm certificates for this module.

4. The tester must initialize, install, and start-up the module while connected to the remote operating environment.

5. If a test harness is used, it **shall** be reviewed or written by the lab. It **shall** be verified to have been maintained properly with no vendor manipulation prior to its execution. The test results on the remote operating environment **shall** be captured and transmitted back to lab without the risk of being modified. The tester **shall** verify the test harness runs properly on its operating environment. The tester must verify the integrity of the testing session as well as the completeness and accuracy of the test results.

6. The vendor may provide assistance to obtain evidence of test results such as printing out reports, taking screenshots or restarting the operating environment as a means to recover from the induced error state of the cryptographic module.

7. The remote testing **shall** cover the same set of FIPS 140-2 requirements including but not limited to the following list, as if the operating environment were local to the tester:
   a. The services listed in the module Security Policy can be invoked and verified by the tester.
   b. For a software module to be validated at Level 2 or 3 for FIPS 140-2 Section 4.3, the role-based or identity-based authentication **shall** be performed and verified by the tester.
   c. The failure of self-tests and the subsequent transition to an error state where module data output interfaces are inhibited can be observed and verified by the tester.
   d. The single-user requirements of AS.06.04 can be verified for Level 1 software module.
   e. Entropy can be effectively analyzed and an entropy report can be generated by the lab.

8. The test report **shall** document how the above conditions are met.

The vendor must provide a signed affirmation letter to the lab describing the remote testing process and access control mechanism that allows the lab to perform the test on the remote operating environment and protects the integrity of the test results. The lab **shall** provide a signed letter to the CMVP stating that the module had been tested remotely, affirming that the vendor provided their affirmation letter, stating what TEs were tested remotely, and explaining how the requirements of this IG were met during the remote testing.

Additional Comments

1. It is the responsibility of the tester to determine if a module is eligible to be tested remotely. If the tester cannot demonstrate a test requirement during remote testing, then the module **shall** not be fully tested remotely. If the tester wishes to test a subset of test requirements remotely, the remaining test requirements **shall** be tested onsite.

2. Rule #2 and Lab Bulletin LB-96-2016 are subject to change.

3. The tester must be able to confirm that the operating environment exactly matches the agreed upon test environment, including any virtual environments used. A Virtual Machine may not be used in lieu of an OS, unless the VM has been agreed to be part of the test environment and will be listed on the certificate.

Section 1 - Cryptographic Module Specification

1.1 Cryptographic Module Name

Applicable Levels:	*All*
Original Publishing Date:	*02/27/2004*
Effective Date:	*02/27/2004*
Last Modified Date:	*02/27/2004*
Relevant Assertions:	*AS.01.05, AS.01.08 and AS.01.09*
Relevant Test Requirements:	*TE01.08.03,04 and 05 and TE01.09.01 and 02*
Relevant Vendor Requirements:	*VE.01.08.03 and VE.01.09.01*

Question/Problem

How **shall** the name of a cryptographic module relate to the defined cryptographic boundary?

Resolution

The provided name of the cryptographic module (which will be on the validation certificate) **shall** be consistent with the defined cryptographic boundary as defined in the test report.

It is not acceptable to provide a module name that represents a module that has more components than the modules defined boundary. If it is desired to have a name that does represent a larger entity, then the cryptographic boundary must be consistent. All components residing within the cryptographic boundary must either be included (**AS.01.08**) or excluded (**AS.01.09**) in the test report.

Additional Comments

Example: The provided name of a cryptographic module is the *Crypto Card*. However, the defined cryptographic boundary in the test report is a small black encapsulated component placed in one corner of the card. The named card also has additional components that were not referenced (e.g. batteries, connectors). If the defined boundary in the test report specifies *ONLY* the black encapsulated component, it is clearly NOT the *Crypto Card*. A unique different name **shall** be provided to be consistent with the defined boundary. To represent the entire card, the boundary must be redefined and must include all the components and address them properly (include/exclude).

1.2 FIPS Approved Mode of Operation

Applicable Levels:	*All*
Original Publishing Date:	*03/15/2004*
Effective Date:	*03/15/2004*
Last Modified Date:	*05/02/2012*
Relevant Assertions:	*AS.01.02, AS.01.03 and AS.01.04*
Relevant Test Requirements:	*TE01.03.01-02 and TE01.04.01-12*
Relevant Vendor Requirements:	*VE.01.03.01-02 and VE.01.04.01-02*

Definition

Approved mode of operation: a mode of the cryptographic module that employs only approved security functions (not to be confused with a specific mode of an approved security function, e.g., AES CBC mode).

Question/Problem

Are there any operational requirements when switching between modes of operation, either from an approved mode of operation to a non-approved mode of operation, or vice versa?

Resolution

CSPs defined in an approved mode of operation **shall** not be accessed or shared while in a non-approved mode of operation. CSPs **shall** not be generated while in a non-approved mode.

Note: An approved DRBG may be used in a non-approved mode. However, the approved DRBGs seed or seed key **shall** not be accessed or shared in the non-approved mode.

Additional Comments

Preventing the access or sharing of CSPs mitigates the risk of untrusted handling of CSPs generated in an approved mode of operation.

Examples:
− a module may not generate keys in a non-approved mode of operation and then switch to an approved mode of operation and use the generated keys for approved services. The keys may have been generated using non-approved methods and their integrity and protection cannot be assured.
− a module may not electronically import keys in plaintext in a non-approved mode of operation and then switch to an approved mode of operation and use those keys for approved services.
− a module may not generate keys in an approved mode of operation and then switch to a non-approved mode of operation and use the generated keys for non-approved services. The integrity and the protection of the approved keys cannot be assured in the non-approved mode of operation.

1.3 Firmware Designation

Applicable Levels:	*All*
Original Publishing Date:	*04/28/2004*
Effective Date:	*04/28/2004*
Last Modified Date:	*06/12/2010*
Relevant Assertions:	*AS.01.01*
Relevant Test Requirements:	
Relevant Vendor Requirements:	

Background

Cryptographic module: the set of hardware, software, and/or firmware that implements approved security functions (including cryptographic algorithms and key generation) and is contained within the cryptographic boundary.

Firmware: the programs and data components of a cryptographic module that are stored in hardware (e.g., ROM, PROM, EPROM, EEPROM or FLASH) within the cryptographic boundary and cannot be dynamically written or modified during execution.

The *operational environment* of a cryptographic module refers to the management of the software, firmware, and/or hardware components required for the module to operate. The operational environment can be non-modifiable (e.g., firmware contained in ROM, or software contained in a computer with I/O devices disabled), or modifiable (e.g., firmware contained in RAM or software executed by a general-purpose computer).

A *limited operational environment* refers to a static non-modifiable virtual operational environment (e.g., JAVA virtual machine on a non-programmable PC card) with no underlying general purpose operating system upon which the operational environment uniquely resides.

If the operational environment is a limited operational environment, the operating system requirements in Section 4.6.1 do not apply.

Question/Problem

How **shall** a *software* cryptographic module running on a limited operational environment be designated as?

Resolution

If the Operational Environment is a limited operational environment, and is indicated as NA on the certificate, then the cryptographic module **shall** be designated as a ***firmware*** module.

Additional Comments

−   The reference tested OS must be indicated on the validation certificate for all software and firmware cryptographic modules. It will be referenced on the CMVP validation list web page as follows:

   o   If the Operational Environment is applicable: *-Operational Environment: Tested as meeting Level x with ...*

   o   If the Operational Environment is NA: *-Tested: ...*

−   For an overall Level 2, 3, or 4 module or where FIPS 140-2 Section 4.5 *Physical Security* is Level 2, 3 or 4, the reference hardware platform with appropriate specific versioning information used during operational testing **shall** also be listed. The certificate caveat **shall** minimally indicate: *When operated only on the specific platforms specified on the certificate*

   −   For JAVA applets, the tested JAVA environment (JRE, JVM) and operating system need to be specified for all Security Levels.

Per IG G.5, porting of software modules is only applicable to modules operating on a General-Purpose Computer (GPC) and when the Operational Environment is applicable. The module's validation will be maintained if no changes are made to underlying source code.

If the operational environment is not applicable, a firmware module at overall Level 1 (with FIPS 140-2 Section 4.5 *Physical Security* at Level 1) and its identified tested OS together may be ported from one platform to another platform while maintaining the module's validation (IG G.5). For firmware module's that are JAVA applets, the firmware module, its identified tested OS, and the tested JAVA environment (JRE, JVM) must be moved together when porting from one platform to another platform in order to maintain the module's validation.

For all other cases, the validation of the cryptographic module is not maintained if ported.

1.4 Binding of Cryptographic Algorithm Validation Certificates

Applicable Levels:	*All*
Original Publishing Date:	*01/21/2005*
Effective Date:	*01/21/2005*
Last Modified Date:	*07/15/2011*
Relevant Assertions:	*AS.01.12*
Relevant Test Requirements:	*TE01.12.01*
Relevant Vendor Requirements:	*VE.01.12.01*

Background

Cryptographic algorithm implementations are tested and validated under the Cryptographic Algorithm Validation Program (CAVP). The cryptographic algorithm validation certificate states the name and version number of the validated algorithm implementation, and the tested operational environment.

Cryptographic modules are tested and validated under the Cryptographic Module Validation Program (CMVP). The cryptographic module validation certificate states the name and version number of the validated cryptographic module, and the tested operational environment.

The validation certificate serves as a benchmark for the configuration and operational environment used during the validation testing.

Question/Problem

What are the configuration control and operational environment requirements for the cryptographic algorithm implementation(s) embedded within a cryptographic module when the latter is undergoing testing for compliance to FIPS 140-2?

Resolution

For a validated cryptographic algorithm implementation to be embedded within a software, firmware or hardware cryptographic module that undergoes testing for compliance to FIPS 140-2, the following requirements must be met:

1. the implementation of the validated cryptographic algorithm has not been modified upon integration into the cryptographic module undergoing testing; and

2. the operational environment under which the validated cryptographic algorithm implementation was tested by CAVS must be identical to the operational environment that the cryptographic module is being tested under by the CST laboratory.

Additional Comments

1. What are examples of an operational environment change?

   If an implementation has been tested on an X-bit processor (e.g. 32-bit, 64-bit), can a claim be made that the implementation also runs on different bit size processors?

   No. An example: An algorithm implementation was tested and validated on a 32-bit platform. This was used in a previous 32-bit version of a software module that was validated for conformance to FIPS 140-2. Now the software module is undergoing testing on a 64-bit platform. This software module cannot operate on a 32-bit platform without change. In this case the operational environments are not the same; therefore, the algorithm implementations must be re-tested on the 64-bit platform. Memory size, processor frequency, etc. are not relevant.

2. If an implementation has been tested on one processor, can a claim be made that the implementation also runs on a different processor when it is submitted for module testing?

   The answer to this question is dependent on the security assurance Level of the module validation and on whether or not the two processors are architecturally compatible or not.

   If the module is being validated as a Level 1 validation and the two processors are architecturally compatible platforms, the answer is Yes. For example, if a Level 1 software module is undergoing testing under Windows 2000 on a DellGatewayPro PC, but the algorithms were tested on Windows 2000 IBMHPClone PC, the algorithm validations do not need to be re-tested as both the DellGatewayPro and IBMHPClone PC's are considered General Purpose Computers (GPC).

   If the two processors are not architecturally compatible, then algorithm validation tests need to be rerun on both processors. For example, a firmware module is undergoing testing on a BlueLiteing processor running Handy OS v5.0. The underlying algorithm implementation was tested on a SlowJoe Processor running Handy OS v0.2. In cases such as this, the algorithm firmware implementations must be re-tested.

   If a Level 2 software module is undergoing testing under an evaluated operating system (OS) and specific platform identified by the evaluation and there is no extensibility provided, the underlying algorithm implementations must be tested under the exact same operational environment (platform and OS).

3. If an algorithm implementation has been tested on one operating system, can a claim be made that the implementation also runs on another operating system when it is considered for module testing?

   No, the algorithm implementation must have been tested on every operating system claimed by the software module at Level 1. The algorithm certificate may include other operating systems as well, but they are not relevant to the module under test. For example, if a Level 1 software module is undergoing testing under Windows 2000, Windows 98 and Linux, the underlying algorithm certificates must indicate at a minimum that the algorithms were tested under Windows 2000, Windows 98 and Linux.

   Another example: A vendor may re-use algorithm implementations between like operational environments. However, if the algorithm implementation testing was only performed on Windows 2000, and the algorithm implementation is to be re-used in a software module undergoing testing

under Windows XP, the algorithm implementations must be re-tested under Windows XP.

4. Who is responsible for finding out what operational environment (processor, operating system) the algorithm implementation is tested on if the testing is done by the vendor and not the CST Lab?

    If algorithm testing is not performed directly by the CST Lab (i.e., if test vectors are provided to the vendor), the CST Lab is responsible for asking the vendor to supply the operating environment (processor and/or operating system) on which they ran the algorithm implementation and with which they generated the RESPONSE files. It is the CST Labs' responsibility to verify that the results in the RESPONSE files were generated using the specified operating environment.

5. If an algorithm is implemented in HDL on a Field Programmable Gate Array (FPGA) device and there is no underlying "OS" implemented in the FPGA, can the algorithm implementation be classified as firmware and, when validated, ported as is to other FPGAs and still be considered validated?

    No. We do not validate HDL (which is equivalent to source code). The algorithm implementation would be validated in the FPGA as hardware.

    Once the FPGA device is validated, one could take the HDL on this FPGA and reuse it in creating a new FPGA. If this were done, the algorithm implementations would need to be validated on the new hardware because they would be considered as new hardware implementations.

6. Additional information regarding operational environment can be found in the CAVP FAQ **GEN.12**.

1.5 moved to A.1

1.6 moved to A.2

1.7 Multiple Approved Modes of Operation

Applicable Levels:	*All*
Original Publishing Date:	*09/12/2005*
Effective Date:	*09/12/2005*
Last Modified Date:	*05/02/2012*
Relevant Assertions:	*AS.01.03 and AS.01.04*
Relevant Test Requirements:	*TE01.03.01-02 and TE01.04.01-02*
Relevant Vendor Requirements:	*VE.01.03.01-02 and VE.01.04.01-02*

Background

FIPS 140-2 Section 4.1 does not preclude a vendor from implementing more than one approved mode of operation in a cryptographic module. An approved mode of operation (IG 1.2) employs the set of approved security functions which are associated with the set of services and CSPs implemented in the module. A module may be designed to employ multiple defined approved modes of operation, where each defined mode

employs a subset of the module's approved security functions, services and CSPs. An example of a module with multiple approved modes of operation is one where the module supports a primary mode that employs all of the approved security functions, services and CSPs of the module to personalize or setup the module, as well as a secondary mode which employs only a subset of approved security functions for normal operation and use.

Question/Problem

May a module implement more than one defined approved modes of operation, each employing a defined set or subset of the approved security functions? What are the requirements for a module to implement more than one approved modes of operation?

Resolution

A cryptographic module may be designed to support multiple approved modes of operation. For a cryptographic module to implement more than one approved modes of operation, the following **shall** apply:

- the security policy **shall** contain the following information describing each approved mode of operation implemented in the cryptographic module:

  - the definition of each approved mode of operation;
  - how each approved mode of operation is configured;
  - the services available in each approved mode of operation;
  - the algorithms used in each approved mode of operation;
  - the CSPs used in each approved mode of operation; and
  - the self-tests performed in each approved mode of operation;

- upon re-configuration from one approved mode of operation to another, the cryptographic module **shall** reinitialize and perform all power-up self-tests associated with the new approved mode of operation:

  - at a minimum, power-up self-tests **shall** be performed on the approved security functions used in the new selected approved mode of operation as specified in FIPS 140-2 Section 4.9 including **AS06.08** in FIPS 140-2 Section 4.6.1 (if applicable), and

  - power-up self-tests **shall** be performed in the new selected approved mode of operation regardless if it had been performed in a prior approved mode of operation.

To confirm the correct operation of the several defined approved modes of operation, the tester **shall**:

- verify the documentation describing each approved mode of operation;

- use the vendor provided instructions described in the non-proprietary security policy to invoke each approved mode of operation;

- verify that, for each approved mode of operation, only the security functions employed for that approved mode of operation are accessible and that security functions not implemented for that approved mode of operation are not; and

- verify that the requirements of **AS.01.03** and/or **AS.01.04** are met for each approved mode of operation.

Additional Comments

CSPs may be shared between multiple approved modes of operation

1.8 Moved to W.13

1.9 Definition and Requirements of a Hybrid Cryptographic Module

Applicable Levels:	*Level 1*
Original Publishing Date:	*03/10/2009*
Effective Date:	*03/10/2009*
Last Modified Date:	*03/19/2010*
Relevant Assertions:	*AS.01.01 and AS.01.08*
Relevant Test Requirements:	
Relevant Vendor Requirements:	

Background

Cryptographic module: the set of hardware, software, and/or firmware that implements approved security functions (including cryptographic algorithms and key generation) and is contained within the cryptographic boundary.

Software: the programs and data components within the cryptographic boundary, usually stored on erasable media (e.g., disk), that can be dynamically written and modified during execution.

Firmware: the programs and data components of a cryptographic module that are stored in hardware (e.g., ROM, PROM, EPROM, EEPROM or FLASH) within the cryptographic boundary and cannot be dynamically written or modified during execution.

Firmware Designation: IG 1.3:

Question/Problem

Define what a *hybrid* cryptographic module is and specify the requirements applicable to this module type?

Resolution

A *hybrid* cryptographic module is a special type of software or firmware cryptographic module that, as part of its composition, utilizes disjoint special purpose cryptographic hardware[1] components installed within the physical boundary of the GPC or operating environment. A hybrid cryptographic module implemented as disjoint hardware and software components is defined as a Software-Hybrid. A hybrid cryptographic module implemented as disjoint hardware and firmware components is defined as Firmware-Hybrid.

In addition to the requirements applicable to a software or firmware cryptographic module, the following requirements are also applicable to the additional cryptographic hardware of the *hybrid* cryptographic module:

- Cryptographic Module Specification: All the components of the *hybrid* cryptographic module must be fully specified by type, part numbers and version numbers;
  - o Manufacturer and model of the special purpose hardware component(s) and platform(s) on which testing was performed;
  - o Operating system(s) on which testing was performed; and
    - ▪ *If Software-Hybrid*: modifiable operating system

---

[1] e.g. cryptographic hardware accelerator cards, cryptographic hardware chip(s), etc.

- ▪ *If Firmware-Hybrid*: the limited or non-modifiable operating system

  o All additional special purpose hardware and firmware components as applicable

- Cryptographic Module Ports and Interfaces: By policy, all status and control ports and interfaces of the hybrid cryptographic module **shall** be directed through the software component logical interface if a software module (controlling component), and through the firmware interface if a firmware module (controlling component);

- Roles, Services and Authentication: All the services provided by the composite of the *hybrid* cryptographic module must be specified;

- Physical Security: FIPS 140-2 Section 5 – *Physical Security* is applicable for a *hybrid* module since a hardware component is specified as part of the hybrid composite.

- Cryptographic Key Management: Key exchanged within the boundary of the GPC or operating platform and between two or more components of the *hybrid* cryptographic module may be transferred in plaintext;

- Self-Tests: Self-tests requirements are applicable to all components of the *hybrid* cryptographic module;

  o A strong integrity test **shall** be performed on the software component,

  o A firmware integrity test (**AS.09.22**) **shall** be performed on any applicable special purpose firmware component, and

  o All other applicable power-up or conditional tests are applicable to all components as required.

- Security Policy: The security policy must specify all the components of the *hybrid* cryptographic module by type, part numbers and version numbers. The security policy must contain a picture of the hardware components of the module. The security policy must specify all the services and sub-services provided by each component of the *hybrid* cryptographic module.

- Operational Environment: FIPS 140-2 Section 6 – The operating system requirements may be applicable for a *hybrid* module.

  o If the module is a Software-Hybrid module; this section is applicable; or

  o If the module is a Firmware-Hybrid module; this section is not applicable.

IG G.13 provides information guidance on how to complete the FIPS certificate for a hybrid module.

Additional Comments

Hybrid cryptographic modules shall be only applicable at FIPS 140-2 Level 1.

The hybrid cryptographic module may be ported to other compatible environments per IG G.5.

Changes to *any* component of the *hybrid* cryptographic module require the re-validation of the complete module as per IG G.8 – *Revalidation Requirements*.

The hardware components and applicable firmware components of the *hybrid* module are considered an extension of the software or firmware module to perform or accelerate cryptographic operations. In a *hybrid* module, the hardware components can only exchange CSPs and control information with the controlling software or firmware component of the module.

1.10 moved to A.3

1.11 moved to [D.1]

1.12 moved to [C.1]

1.13 moved to [A.4]

1.14 moved to [A.5]

1.15 moved to [A.6]

1.16 Software Module

Applicable Levels:	*All*
Original Publishing Date:	*12/23/2010*
Effective Date:	
Last Modified Date:	*12/23/2010*
Relevant Assertions:	*AS.01.01, AS.01.06, AS.01.08, AS.01.09, AS.01.14, AS.06.01, AS.06.02, AS.09.22, AS.09.34, AS.09.35 and AS.14.02*
Relevant Test Requirements:	
Relevant Vendor Requirements:	

Background – FIPS 140-2

Cryptographic module: the set of hardware, software, and/or firmware that implements approved security functions (including cryptographic algorithms and key generation) and is contained within the cryptographic boundary.

Software: the programs and data components within the cryptographic boundary, usually stored on erasable media (e.g., disk), that can be dynamically written and modified during execution.

The *operational environment* of a cryptographic module refers to the management of the software, firmware, and/or hardware components required for the module to operate. The operational environment can be non-modifiable (e.g., firmware contained in ROM, or software contained in a computer with I/O devices disabled), or modifiable (e.g., firmware contained in RAM or software executed by a general-purpose computer).

A *modifiable operational environment* refers to an operating environment that *may* be reconfigured to add/delete/modify functionality, and/or *may* include general purpose operating system capabilities (e.g., use of a computer O/S, configurable smart card O/S, or programmable firmware). Operating systems are considered

to be modifiable operational environments if software/firmware components can be modified by the operator and/or the operator can load and execute software or firmware (e.g., a word processor) that was not included as part of the validation of the module.

If the operational environment is a modifiable operational environment, the operating system requirements in FIPS 140-2 Section 4.6.1 **shall** apply.

FIPS 140-2 DTR – Software

AS.01.01: (Levels 1, 2, 3, and 4) The cryptographic module shall be a set of hardware, software, firmware, or some combination thereof that implements cryptographic functions or processes, including cryptographic algorithms and, optionally, key generation, and is contained within a defined cryptographic boundary.

AS.01.06: (Levels 1, 2, 3, and 4) If the cryptographic module consists of software or firmware components, the cryptographic boundary shall contain the processor(s) and other hardware components that store and protect the software and firmware components.

AS.01.08: (Levels 1, 2, 3, and 4) Documentation shall specify the hardware, software, and firmware components of the cryptographic module, specify the cryptographic boundary surrounding these components, and describe the physical configuration of the module.

AS.01.09: (Levels 1, 2, 3, and 4) Documentation shall specify any hardware, software, or firmware components of the cryptographic module that are excluded from the security requirements of this standard and explain the rationale for the exclusion.

AS.01.14: (Levels 1, 2, 3, and 4) Documentation shall specify the design of the hardware, software, and firmware components of the cryptographic module. High-level specification languages for software/firmware or schematics for hardware shall be used to document the design.

AS.06.01: (Levels 1, 2, 3, and 4) If the operational environment is a modifiable operational environment, the operating system requirements in Section 4.6.1 shall apply.

AS.06.02: (Levels 1, 2, 3, and 4) Documentation shall specify the operational environment for the cryptographic module, including, if applicable, the operating system employed by the module, and for Security Levels 2, 3, and 4, the Protection Profile and the CC assurance level.

AS.09.22: (Levels 1, 2, 3, and 4) A software/firmware integrity test using an error detection code (EDC) or Approved authentication technique (e.g., an Approved message authentication code or digital signature algorithm) shall be applied to all validated software and firmware components within the cryptographic module when the module is powered up.

AS.09.34: (Levels 1, 2, 3, and 4) If software or firmware components can be externally loaded into the cryptographic module, then the following software/firmware load tests shall be performed.

AS.09.35: (Levels 1, 2, 3, and 4) An Approved authentication technique (e.g., an Approved message authentication code, digital signature algorithm, or HMAC) shall be applied to all validated software and firmware components when the components are externally loaded into the cryptographic module.

**AS.14.02: (Levels 1, 2, 3, and 4) The cryptographic module security policy shall consist of:
a specification of the security rules, under which the cryptographic module shall operate, including the security rules derived from the requirements of the standard and the additional security rules imposed by the vendor.**

Question/Problem

How is a *software* cryptographic module defined?

Resolution

A *software* module is a cryptographic module implemented entirely in executable or linked code executing in a modifiable operational environment.

- The physical boundary of a software module is the platform which the software and operating system reside per **AS.01.01** and **AS.01.06**.

- The logical boundary of a software module is the defined set of software components that implement the cryptographic mechanisms. The logical boundary is wholly contained within the physical boundary.

- All components of the cryptographic module **shall** be defined per **AS.01.08** or excluded per **AS.01.09**.

- FIPS 140-2 Section 4.2 defines the physical ports and logical interface requirements. A software modules logical interface **shall** be defined. If applicable, physical ports that map to logical interfaces **shall** be defined.

- FIPS 140-2 Section 4.5 may be marked not applicable (NA) for a software module.

- The power-up approved integrity test **shall** be performed over the defined software image(s) within the cryptographic module logical boundary (RE: **AS.01.01** and **AS.01.06**) per **AS.06.08**.

- The loading of software within the defined logical boundary **shall** meet **AS.09.34-35** and guidance in IG 9.7.

1.17 Firmware Module

Applicable Levels:	*All*
Original Publishing Date:	*12/23/2010*
Effective Date:	
Last Modified Date:	*12/23/2010*
Relevant Assertions:	*AS.01.01, AS.01.06, AS.01.08, AS.01.09, AS.01.14, AS.05.01, AS.06.01, AS.06.02, AS.09.22, AS.09.34, AS.09.35 and AS.14.02*
Relevant Test Requirements:	
Relevant Vendor Requirements:	

Background – FIPS 140-2

Cryptographic module: the set of hardware, software, and/or firmware that implements approved security functions (including cryptographic algorithms and key generation) and is contained within the cryptographic boundary.

Firmware: the programs and data components of a cryptographic module that are stored in hardware (e.g., ROM, PROM, EPROM, EEPROM or FLASH) within the cryptographic boundary and cannot be dynamically written or modified during execution.

The *operational environment* of a cryptographic module refers to the management of the software, firmware, and/or hardware components required for the module to operate. The operational environment can be non-

modifiable (e.g., firmware contained in ROM, or software contained in a computer with I/O devices disabled), or modifiable (e.g., firmware contained in RAM or software executed by a general-purpose computer).

A *limited operational environment* refers to a static non-modifiable virtual operational environment (e.g., JAVA virtual machine on a non-programmable PC card) with no underlying general purpose operating system upon which the operational environment uniquely resides.

If the operational environment is a limited operational environment, the operating system requirements in FIPS 140-2 Section 4.6.1 do not apply.

FIPS 140-2 DTR – Firmware

AS.01.01: (Levels 1, 2, 3, and 4) The cryptographic module shall be a set of hardware, software, firmware, or some combination thereof that implements cryptographic functions or processes, including cryptographic algorithms and, optionally, key generation, and is contained within a defined cryptographic boundary.

AS.01.06: (Levels 1, 2, 3, and 4) If the cryptographic module consists of software or firmware components, the cryptographic boundary shall contain the processor(s) and other hardware components that store and protect the software and firmware components.

AS.01.08: (Levels 1, 2, 3, and 4) Documentation shall specify the hardware, software, and firmware components of the cryptographic module, specify the cryptographic boundary surrounding these components, and describe the physical configuration of the module.

AS.01.09: (Levels 1, 2, 3, and 4) Documentation shall specify any hardware, software, or firmware components of the cryptographic module that are excluded from the security requirements of this standard and explain the rationale for the exclusion.

AS.01.14: (Levels 1, 2, 3, and 4) Documentation shall specify the design of the hardware, software, and firmware components of the cryptographic module. High-level specification languages for software/firmware or schematics for hardware shall be used to document the design.

AS.05.01: (Levels 1, 2, 3, and 4) The cryptographic module shall employ physical security mechanisms in order to restrict unauthorized physical access to the contents of the module and to deter unauthorized use or modification of the module (including substitution of the entire module) when installed.

AS.06.01: (Levels 1, 2, 3, and 4) If the operational environment is a modifiable operational environment, the operating system requirements in Section 4.6.1 shall apply.

AS.06.02: (Levels 1, 2, 3, and 4) Documentation shall specify the operational environment for the cryptographic module, including, if applicable, the operating system employed by the module, and for Security Levels 2, 3, and 4, the Protection Profile and the CC assurance level.

AS.09.22: (Levels 1, 2, 3, and 4) A software/firmware integrity test using an error detection code (EDC) or Approved authentication technique (e.g., an Approved message authentication code or digital signature algorithm) shall be applied to all validated software and firmware components within the cryptographic module when the module is powered up.

AS.09.34: (Levels 1, 2, 3, and 4) If software or firmware components can be externally loaded into the cryptographic module, then the following software/firmware load tests shall be performed.

AS.09.35: (Levels 1, 2, 3, and 4) An Approved authentication technique (e.g., an Approved message authentication code, digital signature algorithm, or HMAC) shall be applied to all validated software and firmware components when the components are externally loaded into the cryptographic module.

AS.14.02: (Levels 1, 2, 3, and 4) The cryptographic module security policy shall consist of: a specification of the security rules, under which the cryptographic module shall operate, including the security rules derived from the requirements of the standard and the additional security rules imposed by the vendor.

Question/Problem

How is a *firmware* cryptographic module defined?

Resolution

IG 1.3 defines the *firmware* module designation, referencing, versioning and porting guidance. Additional guidance:

- The physical boundary of a firmware module is the platform which the firmware and operating system reside per **AS.01.01** and **AS.01.06**.

- The logical boundary of a firmware module is the defined set of firmware components that implement the cryptographic mechanisms. The logical boundary is wholly contained within the physical boundary.

- All components of the cryptographic module shall be defined per **AS.01.06**, **AS.01.08** or excluded per **AS.01.09**.

- FIPS 140-2 Section 4.2 defines the physical ports and logical interface requirements. A firmware module's logical interface shall be defined. If applicable, physical ports that map to logical interfaces shall be defined.

- FIPS 140-2 Section 4.5 is applicable for a firmware module.

- For **Level 1** the firmware module shall prevent access by other processes to plaintext private and secret keys, CSPs, and intermediate key generation values during the time the firmware module is executing/operational. Processes that are spawned by the firmware module are owned by the module and are not owned by external processes/operators. Non-cryptographic processes shall not interrupt the firmware module during execution. The firmware shall be installed in a form that protects the software and firmware source and executable code from unauthorized disclosure and modification.

  Note: These requirements cannot be enforced by administrative documentation and procedures, but must be enforced by the firmware module itself.

  Required Vendor Information - Firmware Module (Level 1 only)

  VE.05.01.01: The vendor shall provide a description of the mechanism used to ensure that no other process can access private and secret keys, intermediate key generation values, and other CSPs, while the cryptographic process is in use.

  VE.05.01.02: The vendor shall provide a description of the mechanism used to ensure that no other process can interrupt the cryptographic module during execution.

  VE.05.01.03: The vendor shall provide a list of the cryptographic firmware that are stored on the cryptographic module and shall provide a description of the protection mechanisms used to prevent unauthorized disclosure and modification.

  Required Test Procedures – Firmware Module (Level 1 only)

  TE05.01.01: The tester shall perform cryptographic functions as described in the crypto officer and user guidance documentation. While the cryptographic functions are executing, the same or another tester shall attempt to access secret and private keys, intermediate key generation values, and other CSPs.

TE05.01.02: The tester **shall** perform cryptographic functions as described in the crypto officer and user guidance documentation. While the cryptographic functions are operating, the same or another tester **shall** attempt to execute another process.

TE05.01.03: The tester **shall** attempt to perform unauthorized accesses and unauthorized modifications to software and firmware source and executable code.

- The mechanisms that define, control and manage the non-modifiable or limited operational environment **shall** be identified per **AS.06.02** and are considered security relevant mechanisms.

- The power-up integrity test **shall** be performed over all non-excluded firmware image(s) defined within the cryptographic module boundary (RE: **AS.01.01** and **AS.01.06**) per **AS.09.22**.

- If the Section 4.5 physical security is Level 1, the loading of firmware within the defined logical boundary **shall** meet **AS.09.34-35** and guidance in IG 9.7.

- If the Section 4.5 physical security is Level 2, 3 or 4, the loading of firmware within the defined physical boundary **shall** meet **AS.09.34-35** and guidance in IG 9.7.

1.18 PIV Reference

Applicable Levels:	*All*
Original Publishing Date:	*04/23/2012*
Effective Date:	
Last Modified Date:	*04/23/2012*
Relevant Assertions:	*AS01.06 and AS01.08*
Relevant Test Requirements:	
Relevant Vendor Requirements:	

Background – FIPS 140-2

Cryptographic module: the set of hardware, software, and/or firmware that implements approved security functions (including cryptographic algorithms and key generation) and is contained within the cryptographic boundary.

A hardware cryptographic module may have an embedded PIV card application component that has been validated by the NPIVP. The PIV card application validation is a prerequisite to the module validation. For module validation, the PIV card application **shall** be tested on the module to be validated (i.e. same operational environment).

Question/Problem

How should a PIV card application component that is included as a component of a cryptographic module be referenced on the module validation entry?

Resolution

The cryptographic module validation entry **shall** provide reference to the PIV card application component(s) validation certificate number.

The PIV card application validation entry **shall** include the following information:

1. the name of the PIV card application component,

2. the name of the cryptographic module the PIV component was tested on, and
3. the complete versioning information of the module including the PIV component(s) (IG G.13).

The cryptographic module's versioning information **shall** include the complete versioning information of the module including the PIV component(s). Each PIV component(s) name **shall** be clearly distinguishable as a PIV component.

IG G.13 defines how the PIV Certificate number is referenced on a module validation.

The NPIVP validation entries can be found at:

http://csrc.nist.gov/groups/SNS/piv/npivp/validation_lists/PIVCardApplicationValidationList.htm

Additional Comments

If a PIV card application component will be used on different cryptographic module operating environments, the PIV card application **shall** be tested and validated by the NPIVP on each of the unique operating environments employed.

1.19 non-Approved Mode of Operation

Applicable Levels:	*All*
Original Publishing Date:	*04/23/2012*
Effective Date:	
Last Modified Date:	*06/20/2012*
Relevant Assertions:	*AS01.02, AS01.03 and AS01.04*
Relevant Test Requirements:	*TE01.03.01-02 and TE01.04.01-12*
Relevant Vendor Requirements:	*VE01.03.01-02 and VE01.04.01-02*

Background

Approved mode of operation: a mode of the cryptographic module that employs only approved security functions.

A cryptographic module **shall** implement at least one approved security function used in an approved mode of operation. Non-approved security functions may also be included for use in non-approved modes of operation. The operator **shall** be able to determine when an approved mode of operation is selected. For Security Levels 1 and 2, the cryptographic module security policy may specify when a cryptographic module is performing in an approved mode of operation. For Security Levels 3 and 4, a cryptographic module **shall** indicate when an approved mode of operation is selected.

Question/Problem

Are there any operational requirements when switching between an approved mode of operation to a non-approved mode of operation, or vice versa?

Resolution

A cryptographic module may be designed to support both an approved mode of operation (IG 1.2), multiple approved modes of operation (IG 1.7) and a non-approved mode of operation. For a cryptographic module to implement an approved mode of operation (one or more) and a non-approved mode of operation, all applicable requirements of FIPS 140-2 **shall** apply with specific attention to the following areas:

AS.01.03: The operator shall be able to determine when an Approved mode of operation is selected.

AS.01.04: For Security Levels 3 and 4, a cryptographic module shall indicate when an Approved mode of operation is selected.

AS01.12: Documentation shall list all security functions, both Approved and non-Approved, that are employed by the cryptographic module and shall specify all modes of operation, both Approved and non-Approved.

AS03.14: Documentation shall specify the services, operations, or functions provided by the cryptographic module, both Approved and non-Approved, and for each service provided by the module, the service inputs, corresponding service outputs, and the authorized role(s) in which the service can be performed.

AS04.02: The cryptographic module shall include the following operational and error states: *User states.* **States in which authorized users obtain security services, perform cryptographic operations, or perform other Approved or non-Approved functions.**

AS14.07: The security policy shall specify; all roles and services provided by the cryptographic module.

IG 1.2: Generation and sharing of CSPs.

IG 1.7: Multiple approved Modes of Operation; if applicable.

IG 9.5: Module Initialization during Power-Up.

IG 14.1: Level of Detail when Reporting Cryptographic Services

In summary, the security policy **shall** contain the following information:

- instructions for the operator to determine when the module is in an approved or non-approved mode of operation;

- instructions for the operator for the configuration to an approved or non-approved mode of operation;

  o is the module configured during initialization to operate only in an approved or non-approved mode of operation when in the operational state, or

  o when in the operational state can the module alternate service by service between approved and non-approved modes of operation

- list all security functions employed by the module in both approved and non-approved modes of operation; and

- list all roles and services, operations or functions provided by the cryptographic module in both approved and non-approved modes of operation;

  o for non-approved service names that reference approved terms, references or functions, the caveat "(non-compliant)" **shall** be appended to the service name to alleviate misinterpretation of approved services; and

  o keys or other parameters associated with non-approved services do not need to be provided.

If the module is configured during power-up initialization to operate only in an approved or non-approved mode of operation;

- a power-on reset **shall** be performed to re-configure the module during initialization from a non-approved mode of operation to an approved mode of operation or vice versa; and

- the conditional self-tests in FIPS 140-2 Section 4.2 are not required when in a non-approved mode of operation with the following exception;

  o the module **shall** not allow the loading of software/firmware components as addressed in FIPS 140-2 Section 4.9.2 *Software/Firmware load test* (i.e. **AS.09.34**).

Additional Comments

This implementation guidance is a further clarification of the FIPS 140-2 clauses and of existing implementation guidance.

1.20 Sub-Chip Cryptographic Subsystems

Applicable Levels:	*All*
Original Publishing Date:	*03/02/2015*
Effective Date:	*09/15/2015*
Last Modified Date:	*02/06/2017*
Relevant Assertions:	
Relevant Test Requirements:	
Relevant Vendor Requirements:	

Background

Increased levels of integration in IC design, such as ASIC, FPGA or System-on-Chip (SoC), have been developed with heterogeneous computing characteristics. Heterogeneous computing may include multiple processors or functional engines, with isolated security subsystem designs that may be re-used in multiple configurations or generations of products.

Question/Problem

What is a *sub-chip cryptographic subsystem*, and what are the requirements for *initial* validation? Once validated, how can the sub-chip cryptographic subsystem be re-validated if modified? How can a non-modified sub-chip cryptographic subsystem be ported and reused on other single-chip implementations?

Resolution

The following terminology is used in the context of this IG:

HDL – Hardware Design Language; examples are Verilog and VHDL.
Security relevant – relevant to the requirements of FIPS 140-2.
Soft circuitry core – an uncompiled hardware subsystem of an ASIC, FPGA or SoC.
Hard circuitry core – a fixed or precompiled hardware subsystem of an ASIC, FPGA or SoC.

For a hardware module, the minimum defined physical boundary in FIPS 140-2 is a single-chip. For single-chip hardware modules a sub-chip cryptographic subsystem may be defined as the set of hard and/or soft circuitry cores and associated firmware which represents a sub-chip cryptographic subsystem boundary of a single-chip hardware module. The sub-chip cryptographic subsystem is integrated on the single-chip which may contain other functional subsystems (e.g. processor(s), memory, I/O and internal bus controls, sensors, etc.) and associated firmware. Upon fabrication of the complete physical single-chip, the HDL will be transformed to a gate or physical circuitry representation which may or may not retain a definable internal sub-chip cryptographic subsystem boundary

1. **Initial validation or security relevant re-validations**
   (3SUB or 5SUB)

- The physical boundary **shall** be defined as the single-chip physical boundary;

  o FIPS 140-2 Section 4.5 requirements **shall** apply at the physical boundary

- FIPS 140-2 defines the Cryptographic boundary as an explicitly defined continuous perimeter that establishes the physical bounds of a cryptographic module, and contains all the hardware, software, and/or firmware components of a cryptographic module. According to IG 1.16, the physical boundary of a software module is the platform in which the software and operating system reside per AS.01.01 and AS.01.06. The logical boundary of a software module is the defined set of software components that implements the cryptographic mechanisms. The logical boundary is wholly contained within the physical boundary. Similarly, for the sub-chip cryptographic subsystem, the physical boundary is the single-chip physical boundary while its logical boundary (the sub-chip cryptographic subsystem boundary) is defined as the set of hard and/or soft circuitry cores and associated firmware that comprises the sub-chip cryptographic subsystem.

- If there is any associated firmware externally loaded into the sub-chip cryptographic subsystem, the associated firmware **shall** meet requirements of software/firmware load test (**AS09.29**, **AS09.34**, **AS09.35** and **AS09.36**).

- Except for externally loaded firmware, the associated firmware **shall** be stored and loaded inside the sub-chip cryptographic subsystem, and **shall** meet software/firmware integrity test (**AS09.13**, **AS09.22**, and **AS09.36**).

- The ports and interfaces (FIPS 140-2 Section 4.2) **shall** be defined at the sub-chip cryptographic subsystem boundary.

  o For operational testing purposes, access to the sub-chip cryptographic subsystem boundary ports **shall** be required and a mapping **shall** be provided. These may be mapped to physical I/O pins, internal test interfaces (e.g. Level Sensitive Scan Design (LSSD)) or the sub-chip boundary data and control ports. The tester **shall** demonstrate that the ports at sub-chip cryptographic subsystem boundary are accessible via the single-chip's other functional subsystems in a manner such that following four kinds of information are provably unmodifiable and under control of the test program:

  - Data input,
  - Data output,
  - Control input, and
  - Status output,

  even in the presence of intervening other functional subsystems.

  Note 1: Typically, the test program acting on behalf of the tester with direct access to the ports and interfaces defined at the sub-chip cryptographic subsystem boundary provides the required demonstration of port access.

  Note 2: In single-chip embodiments, there may be intervening functional subsystems (or intervening circuitry) other than the sub-chip cryptographic subsystem subject to testing. There is a security concern that such intervening subsystems might act maliciously (e.g. intercept, modify, and store CSPs, or attempt a replay attack and/or man-in-the-middle attack). The tester **shall** provide a rationale in the physical security test report explaining existing risks and mitigations. The CMVP may provide additional guidance in the future on how to analyze and document such potential security risks.

  Note 3: If applicable, **VE03.26.01** and **TE03.26.01** **shall** be considered at the level of the tested sub-chip cryptographic subsystem and potential differences between the

internal and external with respect to the subsystem boundary single chip clocks **shall** be accounted for properly.

- Depending on the level, the requirements for Cryptographic Key Establishment and Key Entry and Output (FIPS 140-2 Section 4.7.4) **shall** be applicable at the defined sub-chip cryptographic subsystem boundary.

  o If Key establishment and Key Entry and Output occur across the physical boundary of the single-chip embodiment, **AS07.29** and **AS07.30 shall** apply.

  o Transferring Keys/CSPs including the entropy input between a sub-chip cryptographic subsystem and an intervening functional subsystem for Levels 1 and 2 on the same single chip is considered as not having Key Establishment and Key Entry and Output crossing the boundary of the sub-chip module per **IG 7.7**. Nevertheless, the above Note 2 for the ports and interfaces is applicable for the transferring of Keys/CSPs as well. That is, the tester **shall** provide a rationale in the physical security test report explaining risks and mitigations to the malicious act by such intervening subsystems.

    ▪ Fore Level 3 and Level 4 modules, key establishment is ED / EE as stated in **IG 7.7**.

- Versioning information **shall** be provided for the

  o physical single-chip including any excluded functional subsystem firmware (**shall** be specified in the OE field of the validation),
  o the sub-chip cryptographic subsystem soft and hard circuitry cores, and
  o the associated firmware.

- Processor sub-functions outside the sub-chip cryptographic subsystem boundary but within the physical boundary such as a processor, memory macros, I/O controllers, etc. **may** be excluded under **AS01.09**. However, the data paths used to meet either **AS02.16** and **AS02.18** or **AS02.17** and **AS02.18 shall not** be excluded.

2. **Non-security relevant re-validations associated with changes within physical boundary** (1SUB and 4SUB): Existing **IG G.8** guidance is applicable.

3. **Sub-chip cryptographic subsystem porting**
   The sub-chip cryptographic subsystem may be ported to other single-chip implementations which may be different chip technologies, and/or different non-security relevant functional subsystems.

   A sub-chip cryptographic subsystem that was previously validated in a single-chip can be ported to other single-chip constructs as a 1SUB/4SUB submission to the CMVP. The following is applicable to validate this new single-chip module as a 1SUB/4SUB:

- The laboratory **shall** verify that there are no security relevant changes in the sub-chip cryptographic subsystem;

- If an entropy source is contained within the sub-chip cryptographic subsystem, a new entropy estimate **shall** be provided;

  Note 1: A new entropy estimate may not be required, if the entropy is collected outside the sub-chip cryptographic subsystem, depending on changes to the entropy source or the subsystem housing it. Please refer to **IG 7.14** and **IG 7.15** for details on entropy estimates and applicable caveats.

> **Note 2**: Single chip embodiments may implement a NDRNG or a DRBG linked to a dedicated entropy source (NDRNG) inside the physical boundary. Such cases may be implemented (a) inside the sub-chip cryptographic subsystem or (b) in two or more sub-chip cryptographic subsystems. The case (b) represents multiple disjoint sub-chip cryptographic subsystems (see 4 of this IG).

- Approved security functions **shall** be retested and validated by the CAVP if implemented in a soft circuitry core recompiled in a different part configuration.

  Note 3: If the original algorithm testing was performed as stated in **IG G.11** in a module simulator, and there is no change to the soft-core, no additional algorithm testing is required.

- Operational regression testing (Table G.8.1) **shall** be performed on the new sub-chip cryptographic subsystem after fabrication (transformation of the HDL to a gate or physical circuitry representation);

- FIPS 140-2 Section 4.2 **shall** be addressed for the new single-chip module for all Security Levels within this Section.

- FIPS 140-2 Section 4.5 **shall** be addressed for the new single-chip module at Security Level 1.

- FIPS 140-2 Section 4.8 **shall** be addressed for the new single-chip module for all Security Levels within this Section.

- FIPS 140-2 Sections 4.10.1 and 4.10.4 **shall** be addressed for the new single-chip module for all Security Levels within this Section.

- A new Security Policy **shall** be provided for the new single-chip module.

- A new validation certificate will be issued. Versioning information **shall** be provided for

  - the new physical single-chip
  - non-security relevant single-chip functional subsystem firmware if applicable,
  - the sub-chip cryptographic subsystem soft and hard circuitry cores (which are unchanged from the original validation), and
  - the associated firmware.

The testing laboratory **shall** submit a 1SUB/4SUB test report for the ported updated sub-chip cryptographic subsystem to the CMVP. NIST Cost Recovery fee for scenario 1A is applicable.

4. **Multiple disjoint sub-chip cryptographic subsystems:**

Disjoint sub-chip cryptographic subsystems may exist on a single-chip. Each **shall** be separately validated.

Transferring Keys/CSPs including the entropy input between two disjoint sub-chip cryptographic subsystems on the same single chip for Level 1 and Level 2 modules is considered not having Key Establishment and Key Entry and Output crossing their sub-chip cryptographic subsystem boundary per see **IG 7.7**.

- For Level 3 and Level 4 modules, key establishment is ED / EE as stated in **IG 7.7**.

Alternatively, plaintext CSPs may be shared directly between two disjoint sub-chip cryptographic subsystems via a Trusted Path (**IG 2.1**). In this scenario, the following porting rules **shall** apply:

a. If the two sub-chip modules that are connected by a Trusted Path are ported together, it is considered security relevant and the testing lab **shall** submit a 3SUB or a 5SUB.

b. If only one of the sub-chip modules that are connected by a Trusted Path is ported, then the testing lab **shall** verify that the trusted path is no longer functional and may submit a 1SUB/4SUB.

c. If only one of the sub-chip modules that are connected by a Trusted Path are ported and it is connected to a new sub-chip module, then it is considered security relevant and the testing lab **shall** submit a 3SUB or a 5SUB.

Additional Comments

This IG does not apply to single-chip implementations that do not contain sub-chip cryptographic subsystems, i.e. there is only one boundary which is the physical boundary.

If the sub-chip cryptographic subsystem enters an error state, the FIPS 140-2 requirements are applicable at the boundary of the sub-chip cryptographic subsystem; not at the boundary of the single-chip.

1.21 Processor Algorithm Accelerators (PAA) and Processor Algorithm Implementation (PAI)

Applicable Levels:	*All*
Original Publishing Date:	*03/02/2015*
Effective Date:	*03/02/2015*
Last Modified Date:	*08/07/2017*
Relevant Assertions:	
Relevant Test Requirements:	
Relevant Vendor Requirements:	

Background

As chip fabrication technology advances, additional real estate is becoming available for single-chip processor manufacturers to add acceleration functions to support complex cryptographic algorithms. When these functions are added, the CMVP, the CAVP and the Cryptographic Technology group at NIST will determine if the acceleration function is simply a mathematical construct and not the complete cryptographic algorithm as defined in the NIST standards.

If the function is deemed the complete cryptographic algorithm, then FIPS 140-2 defines the component to be security-specific hardware and complete documentation of the entire component, including HDL, **shall** be submitted to the testing laboratory when under test. This type of implementation is considered a Processor Algorithm Implementation (PAI) function. If the module has been designed to run with and without the security-specific hardware, the resolution below under Software/Firmware Module may apply.

If the function is deemed a mathematical construct and not the complete cryptographic algorithm as defined in the NIST standards, then FIPS 140-2 does not define the component to be security-specific hardware and complete documentation of the entire component, including HDL, is not required. This type of implementation is considered a Processor Algorithm Acceleration (PAA) function.

Question/Problem

What are the currently known processor chips that include Processor Algorithm Acceleration (PAA) and Processor Algorithm Implementation (PAI) functions to support complex cryptographic algorithms and how is it indicated on the validation certificate?

Resolution

If a cryptographic module is designed to utilize a processor chip that includes PAA and/or PAI, the part number or version of the processor chip **shall** be included in **TE.01.08.02**. A module that utilizes such processor hardware may or may not be defined as a hybrid module.

Software/Firmware-Hybrid Module: If the software or firmware component of the hybrid can only support a cryptographic algorithm by exclusively utilizing the PAA or PAI capability, then the module **shall** be defined as a Software/Firmware-Hybrid Module Embodiment ([IG 1.9](#)).

PAA

- **Module versioning** information **shall** include the part number or version of the processor chip.

- **Operational Environment:** Tested as meeting Level 1 with <OS> running on <platform> with PAA

PAI

- **Module versioning** information **shall** include the part number or version of the processor chip.

- **Operational Environment:** Tested as meeting Level 1 with <OS> running on <platform> with PAI

Software/Firmware Module: If the software or firmware component of the module can support a cryptographic algorithm natively or by utilizing the PAA or PAI capability if available, then the module **shall** be defined as a Software/Firmware module Embodiment, unless there are other reasons to designate the module as hybrid.

PAA

- **Algorithm certificates**; the accelerated algorithms **shall** be tested in both native execution and PAA execution.

- **Operational Environment:** Tested as meeting Level 1 with <OS> running on <platform> with PAA; <OS> running on <platform> without PAA

PAI

- **Algorithm certificates**; the algorithms **shall** be tested in both native execution and PAI execution.

- **Operational Environment:** Tested as meeting Level 1 with <OS> running on <platform> with PAI; <OS> running on <platform> without PAI

Known PAAs:

- Intel Processors – Xeon, Core i5, Core i7, Core M and Atom with Westmere, Sandy Bridge, Ivy Bridge, Haswell, Broadwell, Skylake, Kaby Lake micro-architectures: PAA = AES-NI
  - Accelerator sub-functions for AES implementations

- AMD Processors - Opteron, Athlon, Sempron, FX, and A series with Bulldozer, Piledriver, Steamroller, Jaguar, Puma micro-architectures: PAA = AES-NI
  - Accelerator sub-functions for AES implementations

- ARM Cortex A series, R series, Qualcomm Snapdragon, Apple A series processors, Samsung Exynos with ARMv7-A and ARMv8-A micro-architectures: PAA = NEON or Cryptography Extensions
  - Accelerator sub-functions for AES and SHA implementations

- IBM Power Processors 8, 9: PAA = Power ISA
  - Accelerator sub-functions for AES and SHA implementations

- Oracle: Oracle SPARC T series, M series: PAA = SPARC
  - Accelerator sub-functions for AES, DES, and SHA implementations

Known PAIs:

- IBM CP Assist for Cryptographic Functions (CPACF)
  - Full implementations of AES (ECB, CBC), SHA

Additional Comments

NOTE1: AES.2 in the CAVP FAQ gives requirements for both types of implementations.

NOTE2: Please reference IG 1.9 regarding hybrid definition and requirements.

NOTE3: The processor manufacturer may provide a device driver to support use of the processor algorithm accelerator. The device driver **shall** not provide any additional functionality to the PAA.

NOTE4: The implementation of complete algorithms, partial cryptographic modules, or full cryptographic modules as a component of a single-chip, or multiple of any of the above as components of a single-chip, is addressed in the Sub-Chip Cryptographic Subsystems IG.

NOTE 5: Please contact the CMVP to address new PAA or PAI implementations to make a determination whether they are full cryptographic functions or not.

NOTE 6: If the PAI security function appears on the list of known PAIs, its HDL is not required for validation of software modules using it.

1.22 Module Count Definition

Applicable Levels:	*All*
Original Publishing Date:	*11/15/2016*
Effective Date:	*11/15/2016*
Last Modified Date:	
Relevant Assertions:	
Relevant Test Requirements:	
Relevant Vendor Requirements:	

Background

The CMVP allows multiple modules to be validated on a single certificate. However, the separation of these modules in the report is not always clear.

Question/Problem

How does the vendor or lab determine what the module count is for a particular validation?

Resolution

Determining the module count for a validation depends on the type of report; that is, if it is Software, Hardware, Firmware, or a Hybrid.

Software:

- For a software module, its binary package(s) compiled from its source code is the Implementation Under Test (IUT). The same source code may result in different sets of binaries when it's compiled for the different target platforms. The module count **shall** be the number of distinct sets of binaries.

  Examples:

  o If a software module was validated on software version 1.0, and this source code package was compiled on three operating environments of the same family (e.g. *iOS 8.0 running on iPhone5*, *iOS 9.0 running on iPhone5, and iOS 9.1 running on iPhone5)* resulting in a single binary set, the module count is "1".

  o If a software module was validated on software version 1.0, and this source code package was compiled on two operating environments (e.g. *iOS 9.0 running on iPhone5* and *Android*

4.0 running on a Galaxy Nexus) resulting in two separate sets of binaries (each set forming the logical boundary of the module), the module count is "2".

o If a software module was validated on software version 1.0 and software version 2.0, and these source code packages were compiled on four operating environments (e.g. *iOS 9.0 running on iPhone5*, *iOS 9.1 running on iPhone5, Microsoft Windows Phone 8.1 running on Windows Phone 8.1,* and *Android 4.0 running on a Galaxy Nexus)*, where two of the environments are of the same family (*iOS 9.0 and iOS 9.1)* resulting in six separate sets of binaries (software versions 1.0 and 2.0 each map to three distinct sets of binaries), the module count is "6". In this case, a single iOS binary maps to both *iOS 9.0 and 9.1*, a single Microsoft Windows Phone binary maps to *Microsoft Windows Phone 8.1*, and a single Android binary maps to the *Android 4.0*, resulting in three distinct binaries for each software version (1.0 and 2.0), for a total of 6.

Hardware:

- For a hardware module report, the module count can be determined by the physical boundary of the module and understanding the components that are either tested individually and have their own boundary, or the boundary encompasses multiple components and these are tested collectively.

  o If the boundary of the module consists of one hardware component with other hardware components within it, with each having its own hardware version number listed in the certificate (such as tamper seals, service processing cards, switch fabric, core switch blades, control processor blade, power supplies, fan kits, filler panels, management modules, network modules), then the module count **shall** be the number of 'base' modules which support the components within it.

  Examples:

  - If a hardware module report contains a switch (Series 1500, P/N 1010) which can optionally support four additional network modules for uplink ports (P/Ns 10, 20, 30, 40), then the module count is "1" (the switch being the 'base' component).

  - If a hardware module report contains a router with three separately tested part numbers (Series 2000, P/Ns 10, 20, 30), and each router can be configured to use service processing card *A* (P/N 100) or service processing card *B* (P/N 101), along with tamper seal *TAMP1* (P/N 500), then the module count is "3" (the routers, each part number – 10, 20 and 30 - being a 'base' component).

  - If a hardware module report contains a series of four switches and two chassis-based switches (all running either the same firmware, or firmware with non-security relevant differences), and within the boundary of each of the chassis-based switches is a common control processor blade, four different core blades, fiber channel (FC) port blades, an optional extender blade, a power-supply and a tamper seal, then the module count is "6" (the switches being the 'base' component: four switches and two chassis-based switches).

  o If the report has several hardware modules that are individually tested and independent from one another, each having their own cryptographic boundary (flash drives, hard drives, single chips, multi-chips, etc.), but have slight hardware differences (shape, capacity storage, number or type of ports, etc.), then each of the independent hardware pieces **shall** contribute to the module count.

  Examples:

- ▪ If a hardware module report contains two hard drive series with five separately tested configurations [Series SSD1 (P/Ns 128, 256, 500) and SSD2 (P/Ns 1000, 2000)], each with their own cryptographic boundary, the module count is "5".

- ▪ If a hardware module report contains three switch series with eight separately tested configurations [Series 6000 (P/Ns 100, 101, 102), 7000 (P/Ns 200, 201) and 8000 (P/Ns 300, 301, 302)], each with their own cryptographic boundary, the module count is "8".

- o If the hardware module report contains multiple firmware versions tested (with non-security relevant differences) on the same hardware platform, then the module count **shall** reflect the number of hardware modules only, not the number of firmware versions that are running on it.

  - ▪ For example, if a hardware module includes two hard-drives (one being a 250GB drive and the other being a 500GB drive), and each of these drives map to four firmware versions (with non-security relevant differences), the module count is "2" to reflect the hardware platforms.

Firmware:

- For a firmware module, the firmware package itself **shall** be considered a separate module, regardless of the number of hardware platforms it was tested on.

  Examples:

  - o If a firmware package was validated as firmware version 1.0, and this package was tested on two hardware platforms (e.g. *hardwareX* version 1.0 and *hardwareY* version 2.0), the module count is "1".

  - o If a report includes firmware version 1.0 and firmware version 2.0, then the module count is "2", regardless of the number of hardware platforms these packages were tested on.

Hybrid:

- Since hybrid modules (firmware-hybrid or software-hybrid) are dependent on both the software/firmware and the hardware components, the module count **shall** be the total number of configurations that are possible that map to a single module boundary.

  Examples:

  - o If a firmware-hybrid includes hardware version 1.0 and firmware version 3.1, the module count is "1", since there is only a single combination of these two components.

  - o If a firmware-hybrid includes hardware versions 1.0, 1.1, and 1.2, and firmware versions 1.1 and 1.2, and each of the hardware version can map to either of the firmware versions, then the total combination is equal to "6" (3 hardware versions times 2 firmware versions).

1.23 Definition and Use of a non-Approved Security Function

Applicable Levels:	*All*
Original Publishing Date:	*8/07/2017*
Effective Date:	*8/07/2017*
Last Modified Date:	
Relevant Assertions:	

Relevant Test Requirements:	
Relevant Vendor Requirements:	

Background

FIPS 140-2 Glossary:

Approved: FIPS-Approved and/or NIST-recommended.

Approved mode of operation: a mode of the cryptographic module that employs only approved security functions (not to be confused with a specific mode of an approved security function, e.g., DES CBC mode).

Approved security function: for this standard, a security function (e.g., cryptographic algorithm, cryptographic key management technique, or authentication technique) that is either

> *a) specified in an approved standard,*
>
> *b) adopted in an approved standard and specified either in an appendix of the approved standard or in a document referenced by the approved standard, or*
>
> *c) specified in the list of approved security functions.*

FIPS 140-2 Section 3 Functional Security Objectives:

The security requirements specified in this standard relate to the secure design and implementation of a cryptographic module. The requirements are derived from the following high-level functional security objectives for a cryptographic module:

- *To employ and correctly implement the approved security functions for the protection of sensitive information.*

- *To protect a cryptographic module from unauthorized operation or use.*

- *To prevent the unauthorized disclosure of the contents of the cryptographic module, including plaintext cryptographic keys and CSPs.*

- *To prevent the unauthorized and undetected modification of the cryptographic module and cryptographic algorithms, including the unauthorized modification, substitution, insertion, and deletion of cryptographic keys and CSPs.*

- *To provide indications of the operational state of the cryptographic module.*

- *To ensure that the cryptographic module performs properly when operating in an approved mode of operation.*

- *To detect errors in the operation of the cryptographic module and to prevent the compromise of sensitive data and CSPs resulting from these errors.*

FIPS 140-2 Section 4.7 Cryptographic Key management:

Encrypted cryptographic keys and CSPs refer to keys and CSPs that are encrypted using an Approved algorithm or Approved security function. Cryptographic keys and CSPs encrypted using a non-Approved algorithm or proprietary algorithm or method are considered in plaintext form, within the scope of this standard.

IG 3.5 Documentation Requirements for Cryptographic Module Services:

FIPS 140-2 Section 2.1 Glossary of Terms does not provide a definition of service. However, the standard does give a few examples to illustrate the intended use of the term Service. Encryption, authentication, digital signature and key management are mentioned as examples of cryptographic services on page iv. Origin authentication, data integrity and signer non-repudiation are listed as the services provided by a digital signature. Show status and self-tests are mentioned in Section 4.3 as examples of services that do not affect the security of the module.

A service is any externally operator-invoked operation and/or function that can be performed by a cryptographic module.

Question/Problem

The term *non-approved security function* is not defined in the FIPS 140-2 Glossary of Terms, but is cited in multiple places in the standard, DTR and IG. It is central to the correct interpretation of IG 1.2 and IG 1.19. How is *non-approved security function* defined, and how is it interpreted in relation to IG 1.2 "FIPS Approved Mode of Operation" and IG 1.19 "Non-Approved Mode of Operation"?

Resolution

Definition of *non-approved security function*

FIPS 140-2 is concerned specifically with approved and non-approved security functions: the term *non-approved security function* must be defined relative to functions that claim security, rather than all functionality outside the set of *approved security functions*. The term *security* is not defined in the Glossary of Terms, but, within the scope of FIPS 140-2, is determined based on the Section 3 Functional Security Objectives, and the specific Section 4 Security Requirements derived from those objectives.

Security Function: A cryptographic algorithm, cryptographic key management technique, or authentication technique that supports a claim of security and meets the objectives stated in FIPS 140-2 Section 3.

FIPS 140-2 also uses the term *Cryptographic Algorithm*, defined next for consistency with FIPS 140-2 and for convenience in this IG.

Cryptographic Algorithm: An algorithm whose intended function is encryption/decryption, key establishment (inclusive of key generation), message authentication, message digest generation, digital signature generation/verification, or random number generation.

Annexes A, C and D provide the definitive current set of approved cryptographic algorithms. A cryptographic algorithm that is not listed in one of the FIPS 140-2 Annexes (A, C or D) is non-approved.

A *non-approved security function* is any function within the scope of the module that relies on a non-approved cryptographic algorithm to support a claim of security.

Notes

Primitive computational and logical operations (e.g. addition, subtraction, multiplication, division, AND, NOT, OR, and XOR) are used in cryptographic algorithms but are not themselves cryptographic algorithms.

A non-approved cryptographic algorithm or proprietary cryptographic algorithm is not a security function if processed data can be treated as plaintext without violating the Objectives stated in FIPS 140-2 Section 3, the applicable requirements in FIPS 140-2 Section 4, or the security rules specified in the module's Security Policy.

Relationship of non-approved cryptographic algorithms and the modes of operation

Non-approved security functions **shall not** be used in the approved mode of operation; however, non-approved cryptographic algorithms may be used in the approved mode of operation if the non-approved algorithms are not a security function. If a non-approved cryptographic algorithm is used by the module in the approved mode but is not a security function, the algorithm **shall** be included in the list of non-approved algorithms in the Security Policy with the caveat "(no security claimed)" appended to its name.

A non-approved cryptographic algorithm **shall not** share the same keys or CSPs that are used by an approved or allowed algorithm for any cryptographic operation in either the approved, or non-approved mode, as this counters Section 3 Security Objectives by potentially releasing sensitive data and/or CSP(s). A non-approved cryptographic algorithm may still access or modify a CSP in the approved mode (under strict conditions laid out in this IG), as long as the CSP is not used as part of a cryptographic operation, such encryption/decryption, key establishment (inclusive of key generation), message authentication, message digest generation or digital signature generation/verification. The only exception to the rule explained in the first sentence of this paragraph, is the use of a non-approved cryptographic algorithm that utilizes an *approved DRBG* for any purpose such as key establishment, stand-alone random number generation, hashing, data obfuscation, etc.

Despite access and modification of the state of the DRBG CSP(s) by a non-approved algorithm, this is allowed in both the approved and non-approved modes of operation. See the examples below for more information.

Possible example scenarios of non-approved cryptographic algorithms in various modes of operation

Example scenarios of non-approved cryptographic algorithms allowed in FIPS mode

1. Use of a non-approved cryptographic algorithm to "obfuscate" a CSP

   For purposes of storage or certificate formatting (e.g. PFX), a module might:

   - XOR a CSP with a secret value

   - Encrypt or decrypt a CSP using a proprietary or non-approved cryptographic algorithm.

   - Store authentication data using MD5 or using HMAC-SHA-1 with a weak HMAC key

   - Format certificate data using a non-approved PKCS #12

   As noted in Section 4.7, "Cryptographic keys and CSPs encrypted using a non-approved algorithm or proprietary algorithm or method are considered in plaintext form, within the scope of this standard."

   All Section 4 requirements must be satisfied when considering the CSP in plaintext form:

   - The report description of CSPs must correctly describe the form of the CSP.

   - The module must support zeroization of any CSPs stored internally in the forms described above.

   - If the obfuscated CSP is imported or exported, the module must meet the requirements for plaintext CSP import or export.

   This conclusion is consistent with IG 7.16 *Acceptable Algorithms for Protecting Stored Keys and CSPs*.

2. Use of an approved, non-approved or proprietary algorithm for a purpose that is not security relevant or is redundant to an approved cryptographic algorithm

   a. Use of MD5 in the TLS 1.0 / 1.1 KDF

      SP 800-135 Rev1 Section 4.2.1 describes the use of MD5 in conjunction with SHA-1 in the key derivation function, concluding that the TLS 1.0/1.1 KDF may be used within the context of the TLS protocol (with provisions for validation of the companion approved functions, SHA-1 and HMAC).

      This use of MD5 does not conflict with the security of the approved security functions.

   b. Storage device use of a PRF (e.g. XTS AES) for memory wear leveling (a technique for prolonging the service life of some kinds of erasable computer storage media). For best results, a method with good statistical properties (i.e. a PRF) may be used for wear leveling, redundant to any other encryption or decryption performed by the module. This use of an algorithm is not for a security purpose; it is to prolong memory life.

   c. A secure channel operated over an insecure communications channel

      Consider a module whose purpose is to provide end-to-end secure communications over an insecure communications channel. That channel may be plaintext or some method which provides insufficient security, assumed to provide no greater security than plaintext.

      Specifically, assume the module communicates over a normal, unprotected Ethernet, provides approved end to end encryption, decryption and message authentication, as well as initial authentication of the peer node, and meets all FIPS 140-2 Section 4 requirements. This module can be validated.

      Consider the same scenario but with wireless communications over WEP, WPA, WPA2 or similar, where the purpose of the module is a ***remedy*** for insecure communications media.

The module must communicate with a WAP using the communications protocols the WAP provides. If the channel is treated as plaintext, and the module provides secure channel services that meet all FIPS 140-2 Section 4 requirements, to deny validation to such a module because the communications media uses non-approved functions defeats the purpose of the module, and is contrary to the intent of the CMVP as a program.

d. Non-approved cryptographic algorithm that uses an approved DRBG for cryptographic purposes

The module uses a non-approved cryptographic algorithm to "obfuscate" a CSP for RAM storage. The key used for "obfuscation" is derived via an approved DRBG. By doing this, the DRBG changes its state, and therefore the DRBG CSPs are modified. Despite the modification and use of the DRBG CSPs within a cryptographic operation, this is allowed because the DRBG is the exception to the rule laid out in this IG.

3. Use of a non-approved cryptographic algorithm as part of an approved algorithm that claims security

a. Use of GHASH within AES GCM

Although GHASH, alone, is a non-approved hashing function, it is used within an approved AES GCM algorithm, and is therefore permitted, even if the vendor claims security on this algorithm. However, if the vendor claims security on this function, then it **shall not** be used in the approved mode for any independent operation outside of the approved algorithm.

Example scenarios of non-approved cryptographic algorithms <u>not allowed in any mode</u>

1. Non-approved cryptographic algorithm that share the same key or CSP as an approved algorithm

a. A DES algorithm is encrypting data using a DES key K1. This key is a part of a Triple-DES key K = (K1, K2, K3) which is a CSP, as it may be used by an approved Triple-DES algorithm. The value $E = DES_{K1}$ (data) is sent outside the module's boundary. An attacker can easily break the single-DES encryption and recover K1, which will lead to the disclosure of the Triple-DES key K.

b. Suppose a module generates, in full compliance with FIPS 186-4, a key pair for an approved RSA signature algorithm. However, the module also has a non-approved RSA signature algorithm not claiming any security. This non-approved RSA signature algorithm could use the same RSA key to generate its "signatures". These non-approved signatures may be broken by an attacker and the signing key may be recovered, allowing the attacker to use this key to sign what *they* want.

The reason the above two examples are prohibited is because they do not follow the above rule which states: "A non-approved cryptographic algorithm **shall not** share the same keys or CSPs that is used by an approved or allowed algorithm for any cryptographic operation in either the approved, or non-approved mode". Even if the vendor claims no security on these non-approved algorithms, they are still not allowed.

Additional Comments

The vendor must provide clear documentation and reasoning as to why the non-approved cryptographic algorithms can be used in an Approved Mode, i.e. not being used to meet the requirements of FIPS 140-2 sections 3 and 4. It is at the discretion of the CMVP to determine if such usage of an algorithm fits within the guidance laid out in this IG.

Section 2 – Cryptographic Module Ports and Interfaces

2.1 Trusted Path

Applicable Levels:	*Levels 3 and 4*
Original Publishing Date:	*12/23/2010*
Effective Date:	
Last Modified Date:	*12/23/2010*
Relevant Assertions:	*AS.02.17, AS.02.18 and AS.07.33*
Relevant Test Requirements:	
Relevant Vendor Requirements:	

Background

The implementation of a trusted path protects plaintext CSPs during the input or output from a cryptographic module when protections by cryptographic mechanisms are not employed (e.g. key establishment methods in FIPS 140-2 Section 4.7).

For example:

- **AS.02.16: (Levels 3 and 4) The physical port(s) used for the input and output of plaintext cryptographic key components, authentication data, and CSPs shall be physically separated from all other ports of the cryptographic module or AS.02.17 must be satisfied.**

- **AS.02.17: (Levels 3 and 4) The logical interfaces used for the input and output of plaintext cryptographic key components, authentication data, and CSPs shall be logically separated from all other interfaces using a *trusted path* or AS.02.16 must be satisfied.**

- **AS.02.18**: **(Levels 3 and 4) Plaintext cryptographic key components, authentication data, and other CSPs shall be directly entered into the cryptographic module (e.g., via a *trusted path* or directly attached cable).**

- **AS.07.33: (Levels 3 and 4) If split knowledge procedures are used, plaintext cryptographic key components shall be directly entered into or output from the cryptographic module (e.g., via a *trusted path* or directly attached cable) without traveling through any enclosing or intervening systems where the key components may inadvertently be stored, combined, or otherwise processed (see Section 4.2).**

Question/Problem

How is a *trusted path* described when the connection (i.e. the source or destination of the path) is not under the direct control of the cryptographic module?

Resolution

If a cryptographic module utilizes the mechanisms of a trusted path at Level 3 or higher to input or output plaintext CSPs (i.e. cryptographic methods are not employed), in addition to the requirements specified in the FIPS 140-2 DTR, the Security Policy **shall** specify and document:

- characteristics of the trusted path,
- specify the controls that are used to maintain the trusted path,
- operator instructions for setup and operation of the trusted path,
- the specific characteristics and specification of the source or target of the trusted path relative to the cryptographic module

The cryptographic module validation certificate **shall** provide a specific caveat regarding the use of the trusted path, e.g.

When operated in FIPS mode and utilizing a Trusted Path as specified in the Security Policy.

Section 3 – Roles, Services, and Authentication

3.1 Authorized Roles

Applicable Levels:	*2, 3, and 4*
Original Publishing Date:	*05/29/2002*
Effective Date:	*05/29/2002*
Last Modified Date:	*08/07/2017*
Relevant Assertions:	
Relevant Test Requirements:	
Relevant Vendor Requirements:	

Question/Problem

From FIPS 140-2 Section 4.3:

An operator is not required to assume an authorized role to perform services where cryptographic keys and CSPs are not modified, disclosed, or substituted (e.g., show status, self-tests, or other services that do not affect the security of the module).

From FIPS 140-2 Section 4.3.3:

Authentication mechanisms may be required within a cryptographic module to authenticate an operator accessing the module, and to verify that the operator is authorized to assume the requested role and perform the services within the role.

Question/Problem

What are the services that do not require an operator, in the approved mode, to assume an authorized role and, therefore, not be authenticated, as required if Security Level 2, 3, or 4 is claimed for Section 4.3?

Resolution

If a Security Level 2 or above is claimed for Section 4.3, an operator in the approved mode **shall** be authenticated when assuming a role for all services utilizing approved security functions, with the following exceptions:

(a) The hash algorithms which are specified in **FIPS 180-4** and **FIPS 202**;

(b) The deterministic random number generators which are specified in **SP 800-90A rev1**. If the DRBG service is provided to an authenticated operator, the entropy source seeding the DRBG **shall** be completely contained within the boundary of the cryptographic module;

(c) Digital signature verification, as specified in "Digital Signature Standard", **FIPS 186-2** and **FIPS 186-4**.

(d) Authentication procedures used for authenticating the operator and/or initialization procedures to setup the operator's authentication credentials; and

(e) Show status, self-tests, or other services that do not permit an operator to modify, disclose or substitute CSPs and do not affect the security of the module or the security of the information being protected by the module.

Additional Comments

1. The reason for the stated exceptions is that the referenced algorithms do not create, disclose or modify the module's CSPs.

2. FIPS 140-2 Section 4.3 talks about "authorized" roles. For the purposes of this IG, an authorized role is any defined role. Some of these defined roles may require an operator to get authenticated before the operator is authorized to assume the role.

3. Performing any service requires an assumption of a role. This IG clarifies under what conditions some of the roles may remain unauthenticated. When the FIPS 140-2 standard states (see the **Background** section above) that an operator is not required to assume an authorized role to perform certain services, this means that while the module may be validated at Security Level 2 or above in Section 4.3, a defined role may not require an authentication of an operator for the role to perform these services.

4. It is stated in (e) in the Resolution section that an unauthenticated service may not, at Security Levels 2 and above, cause a modification of the module's CSP. There exists an exception to this rule. An approved DRBG may be called from an unauthenticated role, or even from a role that includes the non-approved services. Each execution of a DRBG may result in a modification of the DRBG's secret state parameters, which are the module's CSPs (see IG 14.5). This indirect modification of the CSPs is permissible because it does not result in the weakening of the CSPs or in a loss of their secrecy.

5. The zeroization of all of the module's unprotected keys and CSPs performed as required in Section 4.7.6 of FIPS 140-2 is not viewed as a "modification" of these parameters. Therefore, the corresponding zeroization service may be called from an unauthenticated role.

3.2 Bypass Capability in Routers

Applicable Levels:	*ALL*
Original Publishing Date:	*04/01/2009*
Effective Date:	*04/01/2009*
Last Modified Date:	*04/01/2009*
Relevant Assertions:	*AS.03.12 and AS.03.13*
Relevant Test Requirements:	
Relevant Vendor Requirements:	

Background

A router is a particular type of cryptographic module where bypass is typically applicable but has some unique attributes. Typically, a router has an internal IP address table that contains entries for known addresses as well as instructions specifying routing destinations and whether the packets are to be encrypted or passed in plaintext. In addition, if an unknown IP address is found, a router may "drop" the incoming packet or pass it to a predetermined address unchanged (e.g. default gateway).

Question/Problem

Is the cryptographic module subject to the bypass requirements of FIPS 140-2 if packets with an unknown IP address are either dropped or re-directed to a predetermined address (e.g. default gateway)?

Resolution:

The bypass requirements of FIPS 140-2 are not applicable if packets with an unknown IP address are dropped unprocessed.

Packets with an unknown IP address that are re-directed to a predetermined address (e.g. default gateway) are bypassing the module's encryption and the bypass requirements of FIPS 140-2 are applicable.

This IG is also applicable to cryptographic modules that are offering an exclusive bypass capability or no bypass capability at all.

3.3 Authentication Mechanisms for Software Modules

Applicable Levels:	*Levels 2, 3, or 4*
Original Publishing Date:	*05/02/2012*
Effective Date:	
Last Modified Date:	*05/02/2012*
Relevant Assertions:	*AS03.31 and AS03.32*
Relevant Test Requirements:	
Relevant Vendor Requirements:	

Background

A cryptographic module may implement authentication mechanisms to authenticate an operator accessing the module and to verify that the operator is authorized to assume the requested role and perform services within that role. Depending on the security level, a cryptographic module may support role-based or identity-based authentication.

Question/Problem

Can a software module (IG 1.16) rely on the authentication mechanisms employed in the operating environment rather than implemented explicitly by the software module within the software modules logical boundary?

Resolution

If a software cryptographic module supports either role-based or identity-based authentication, the authentication mechanisms **shall** be implemented within the logical boundary of the module with the following *exception*:

- If FIPS 140-2 Section 4.6 *Operating Environment* is validated at Level 2, 3, or 4, the authentication mechanisms employed in the operating environment may be used to meet the FIPS 140-2 Section 4.3 authentication requirements. If role-based authentication is claimed in FIPS 140-2 Section 4.3, then the operating environment **shall** satisfy either the role-based or identity-based requirements in FIPS 140-2 Section 4.3. If identity-based authentication is claimed in FIPS 140-2 Section 4.3, then the operating environment **shall** satisfy identity-based requirements in FIPS 140-2 Section 4.3.

  - If the operating environment requires special configuration settings to satisfy the selected authentication method in FIPS 140-2 Section 4.3, the configuration settings **shall** be defined in the Security Policy, and the Security Policy **shall** indicate that the Crypto Officer Role is responsible for ensuring the configuration settings are properly set for the module to operate in an approved mode of operation.

3.4 Multi-Operator Authentication

Applicable Levels:	*Levels 2, 3 and 4*
Original Publishing Date:	*05/02/2012*
Effective Date:	
Last Modified Date:	*05/10/2017*
Relevant Assertions:	*AS03.16, AS03.17, AS03.18, AS03.19 and AS03.20*
Relevant Test Requirements:	
Relevant Vendor Requirements:	

Background

AS03.16: (Levels 2, 3, and 4) Depending on the security level, the cryptographic module shall perform at least one of the following mechanisms to control access to the module: *role-based authentication* **or** *identity-based authentication.*

AS03.17: (Level 2) If role-based authentication mechanisms are supported by the cryptographic module, the module shall require that one or more roles either be implicitly or explicitly selected by the operator and shall authenticate the assumption of the selected role (or set of roles).

AS03.19: (Level 3 and 4) If identity-based authentication mechanisms are supported by the cryptographic module, the module shall require that the operator be individually identified, shall require that one or more roles either be implicitly or explicitly selected by the operator, and shall authenticate the identity of the operator and the authorization of the operator to assume the selected role (or set of roles).

Question/Problem

A module may implement separately defined operator roles which have different authentication claims. For example, the Crypto Officer (CO) role implements *identity-based authentication* while the User role implements *role-based authentication* (Case 1). In another example, the CO role implements *role-based authentication* while the User role does not implement any *authentication* (Case 2). There is also a possibility of the CO and User roles each supporting role-based as well as the identity-based authentication (Case 3): some of the operators who are assuming a given role are authenticated using the role-based credentials, while others, who will also assume this role, pass an identity-based authentication. Are these implementations compliant with the requirements of Section 4.3 of FIPS 140-2, and, if so, at what security level?

For the above scenarios, it is assumed that approved security services are included in each assumed role. Should there be an exception to the operator authentication requirement when the approved security functions do not affect the security of the module?

Resolution:

Following are the resolutions for the three scenarios from the Question/Problem section above.

1. The first case (Case 1) is compliant to FIPS 140-2 Section 4.3 because for the purposes of the FIPS 140-2 validation, *identity-based authentication* is considered to be meeting the *role-based authentication* requirement. Both the CO and the User operators get authenticated to access the approved security services. The section security level is 2 because it is the lower of the two authentication methods described.

    The security policy **shall** identify all roles, and for each role, the authentication method (i.e. either *role-based* or *identity-based*).

2. In the second case (Case 2) the module is compliant to FIPS 140-2 Section 4.3 level 2 only if the unauthenticated User role does not call any services that affect the module's security. See IG 3.1 for the

definition of such services. Otherwise, FIPS 140-2 Section 4.3 is annotated at level 1 and only the level 1 assertions are addressed.

3. The Case 3 scenario is also compliant with FIPS 140-2. The vendor can claim compliance with Section 4.3 only at security level 2. The test report addresses each role at security level 2. The security policy **shall** explain how the authentication may be performed for each role.

Additional Comments

1. **IG 3.1** addresses authenticated roles for approved security services and non-authenticated services.

2. In Case 3, the module can only be validated at level 2 in Section 4.3 because the role-based authentication is also available to the module.

3. Other mixed cases are also possible. There is sufficient information in this Implementation Guidance to determine how to treat each of these cases and what will be the overall security level of the module's validation in Section 4.3. For example, the User role can have both a role-based and an identity-based authentication, while the Crypto Officer role always requires an identity-based authentication. As shown above, such module is validated at security level 2 in Section 4.3, unless the User role only calls the services that are exceptions identified in IG 3.1 as not affecting the module's security. If the latter case, the module's Section 4.3 may be validated at security level 3.

4. When the module supports both the role-based and the identity-based authentication, either within the same role (as in Case 3 above) or by the different roles (as in Case 1), the testing laboratory, when writing the Test Report, **shall** select the "Identity Auth." option in the Module Information form. This will require the testing laboratory to address in the Test Report both the level 2 (role-based) and the level 3 (identity-based) assertions.

3.5 Documentation Requirements for Cryptographic Module Services

Applicable Levels:	*All*
Original Publishing Date:	*07/25/2013*
Effective Date:	
Last Modified Date:	*11/15/2016*
Relevant Assertions:	*AS.03.14, AS.14.07*
Relevant Test Requirements:	*TE.03.14.01, TE.14.07.01*
Relevant Vendor Requirements:	

Background

From FIPS 140-2 Section 4.3.2 and Appendix C:

AS03.07: (Levels 1, 2, 3, and 4) Services shall refer to all of the services, operations, or functions that can be performed by the cryptographic module.

AS03.08: (Levels 1, 2, 3, and 4) Service inputs shall consist of all data or control inputs to the cryptographic module that initiate or obtain specific services, operations, or functions.

AS03.09: (Levels 1, 2, 3, and 4) Service outputs shall consist of all data and status outputs that result from services, operations, or functions initiated or obtained by service inputs.

AS03.10: (Levels 1, 2, 3, and 4) Each service input shall result in a service output.

AS03.14: (Levels 1, 2, 3, and 4) Documentation shall specify:

- **the services, operations, or functions provided by the cryptographic module, both Approved and non-Approved, and**
- **for each service provided by the module, the service inputs, corresponding service outputs, and the authorized role(s) in which the service can be performed.**

AS14.07: (Levels 1, 2, 3, and 4) The security policy shall specify: all services provided by the cryptographic module.

Question/Problem

FIPS 140-2 Section 4.3.2 Roles, Services, and Authentication lays out requirements for service inputs, service outputs, correlation between inputs and outputs, and access control on the services by authorized roles as stated in the Background section above. Nevertheless, it does not specify what operations or functions that can be performed by the cryptographic module are considered as services. The statement that services shall refer to all of the services does not answer the question what is considered as a service and must be documented in a security policy.

FIPS 140-2 Section 2.1 Glossary of Terms does not provide a definition of service. However, the standard does give a few examples to illustrate the intended use of the term Service. *Encryption, authentication, digital signature* and *key management* are mentioned as examples of cryptographic services on page iv. *Origin authentication, data integrity* and *signer non-repudiation* are listed as the services provided by a digital signature. *Show status* and *self-tests* are mentioned in Section 4.3 as examples of services that do not affect the security of the module.

What is the definition of service in the context of FIPS 140-2? What is the expected level of granularity to specify a service in order to meet the referenced requirements? Do all services need to be documented in the security policy, including the services that are not security-relevant or not specified in FIPS 140-2 Section 4.3.2?

Resolution

Services of a cryptographic module are the top-level operations and/or functions that represent the module's main functionality provided through its external interface. The services that are commonly provided by a cryptographic module are among Encryption, Digital Signature operations, Key Derivation Functions, Key Establishment Schemes, Message Authentication, Random Number generation, Secure Hashing, User Authentication, Self-tests, key Zeroization, Show Status, Protocol Handshake, Signature Operations, etc.

FIPS 140-2 states unambiguously that *all* services need to be documented in the module's Security Policy. This applies to the following groups:
1. services that use approved (i.e. including allowed) security functions and mechanisms that are available for use in an approved mode of operation,

2. services that do not use any security functions (i.e. approved or non-approved), but are described in FIPS 140-2 Section 4.3.2 (e.g. Show Status service),

3. services that use non-approved security functions or mechanisms and therefore not available for use in an approved mode of operation,

4. services that may perform actions that are not addressed in the above bullets. An example of such service would be "image manipulation."

The security policy **shall** list each service individually that belongs to groups 1 to 3, as per **AS14.07**. When reporting cryptographic services in group 1, IG 14.1 provides the guidance for level of detail.

For services that belong to group 4, the security policy **shall** either list them individually in the same manner as all other services, or provide a reference to separate external document where these services are documented. A reference **shall** include the document name, version number, release date and how the document can be publicly acquired (e.g. a provided URL).

The description of each service **shall** address the requirements in FIPS 140-2 Appendix C.3.2.

All module security functions listed in **AS01.12 shall** map to at least one defined security service.

Additional Comments

A service is any externally operator invoked operation and/or function that can be performed by a cryptographic module. A service **shall** correspond to a specific task or callable function to be performed by the module.

Services provided by a software module are not required to have one-to-one correspondence to the API functions implemented by the module. A service (e.g. Random Number Generation) may invoke a group of API functions. On the other hand, an API function may provide different services (e.g. symmetric encryption vs. asymmetric encryption) depending on the different values of some or all of its input parameters. A vendor may choose to document services in terms of API functions if appropriate. Nevertheless, API functions are not required to be the only way to specify services.

Section 4 - Finite State Model

Section 5 - Physical Security

5.1 Opacity and Probing of Cryptographic Modules with Fans, Ventilation Holes or Slits at Level 2

Applicable Levels:	*Level 2*
Original Publishing Date:	*02/10/2004*
Effective Date:	*02/10/2004*
Last Modified Date:	*02/10/2004*
Relevant Assertions:	*AS.05.49*
Relevant Test Requirements:	*TE05.49.01*
Relevant Vendor Requirements:	*VE.05.49.01*

Background

Cryptographic modules typically require the use of heat dissipation techniques that can include the use of fans, ventilation holes or slits. The size of these openings in the modules' enclosure, or the spacing between fan blades, may allow the viewing or possible probing of internal components and structures within the cryptographic module.

Question/Problem

How do the opacity requirements of FIPS 140-2 affect the design of the heat dissipation techniques on those cryptographic modules at Security Level 2? Should the cryptographic module prevent probing through the ventilation holes or slits at Security Level 2?

Resolution

The following are the physical security requirements for multi-chip stand-alone module at Security Level 2 pertaining to opacity and probing:

- the embodiments that are entirely contained within a metal or hard plastic production-grade enclosure that may include doors or removable covers (Security Level 1 requirement); and

- the enclosure of the cryptographic module **shall** be opaque within the visible spectrum.

Probing Requirements

Probing is not addressed at Security Level 2. Probing through ventilation holes or slits is addressed at Security Level 3 (**AS.05.21**).

Opacity Requirements

The purpose of the opacity requirement is to deter direct observation of the cryptographic module's internal components and design information to prevent a determination of the composition or implementation of the module.

A module is considered "opaque" only if it cannot be determined by visual inspection within the visible spectrum using artificial light sources shining through the enclosure openings or translucent surfaces, the manufacturer and/or model numbers of internal components (such as specific IC types) and/or design and composition information (such as wire traces and interconnections).

Component outlines may be visible from the enclosure openings or translucent surfaces as long as the component's manufacturer and/or model numbers, and/or composition and information about the module's design cannot be determined.

All components within the boundary of the cryptographic module must meet the opacity requirements of the standard. Excluded non-security relevant components do not have to meet these requirements.

Additional Comments

Note: Visible light is defined as light within a wavelength range of 400nm to 750nm.

5.2 Testing Tamper Evident Seals

Applicable Levels:	*Levels 2, 3 and 4*
Original Publishing Date:	*09/12/2005*
Effective Date:	*09/12/2005*
Last Modified Date:	*09/12/2005*
Relevant Assertions:	*AS.05.16, AS.05.35, AS.05.36, AS.05.37, AS.05.48, AS.05.50*
Relevant Test Requirements:	
Relevant Vendor Requirements:	

Question/Problem

What level of testing and scope of testing should be applied when testing tamper evident seals?

Resolution

If a module uses tamper evident labels, it **shall** not be possible to remove or reapply a label without tamper evidence. For example, if the label can be removed without tamper evidence, and the same label can be re-applied without tamper evidence, the assertion fails.

Conversely, if any attempt to remove the label leaves evidence, or removal and re-application leaves evidence, or the label is destroyed during removal, the assertion passes. This means that the CST laboratory **shall** have to use creative ways (e.g. chemically, mechanically, thermally) to remove a label without evidence and without destroying the original label, and be able to re-apply the removed label in a manner that does not leave evidence.

Additional Comments

It is out-of-scope for an attacker to introduce new materials to cover up evidence of the attack.

5.3 Physical Security Assumptions

Applicable Levels:	*ALL*
Original Publishing Date:	*03/10/2009*
Effective Date:	*03/10/2009*
Last Modified Date:	*03/10/2009*
Relevant Assertions:	
Relevant Test Requirements:	
Relevant Vendor Requirements:	

Background

Extracted from FIPS 140-2 Section 1 – OVERVIEW:

FIPS 140-1 was developed by a government and industry working group composed of both operators and vendors. The working group identified requirements for four security levels for cryptographic modules to provide for a wide spectrum of data sensitivity (e.g., low value administrative data, million dollar funds

transfers, and life protecting data) and a diversity of application environments (e.g., a guarded facility, an office, and a completely unprotected location). Four security levels are specified for each of 11 requirement areas. Each security level offers an increase in security over the preceding level. These four increasing levels of security allow cost-effective solutions that are appropriate for different degrees of data sensitivity and different application environments. FIPS 140-2 incorporates changes in applicable standards and technology since the development of FIPS 140-1 as well as changes that are based on comments received from the vendor, laboratory, and user communities.

The use of a validated cryptographic module in a computer or telecommunications system is not sufficient to ensure the security of the overall system. The overall security level of a cryptographic module must be chosen to provide a level of security appropriate for the security requirements of the application and environment in which the module is to be utilized and for the security services that the module is to provide. The responsible authority in each organization should ensure that their computer and telecommunication systems that utilize cryptographic modules provide an acceptable level of security for the given application and environment.

The importance of security awareness and of making information security a management priority should be communicated to all users. Since information security requirements vary for different applications, organizations should identify their information resources and determine the sensitivity to and the potential impact of losses. Controls should be based on the potential risks and should be selected from available controls, including administrative policies and procedures, physical and environmental controls, information and data controls, software development and acquisition controls, and backup and contingency planning.

FIPS 140-2 does not specify the required strength of the approved security functions that may be implemented within a cryptographic module at each security level. Allowable strengths are addressed in IG 7.5. Therefore, a Level 1 module may implement the same security strength of an encryption function as a Level 4 module.

The four physical security levels of FIPS 140-2 are focused on the protection of the modules CSPs by the module itself independent of the environment the module is deployed. Therefore, selection of a security level is greatly influenced by the environment the module is to be deployed. At a Level 1 security level, which does not itself provide physical security protection, in the right environment, may be an acceptable solution because the environment provides the required physical security protection features.

A software cryptographic module is not subject to the physical security requirements of this standard. The following resolution assumes the host platform is not subject to the physical security requirements of FIPS 140-2.

Question/Problem

What are the assumptions that have defined the protection, attack types and operator roles in the FIPS 140-2 physical security requirements for which a cryptographic module itself provides at each security level?

Resolution:

Level 1

Protection Provided:
No physical protection of CSPs; access assumed
> Hardware: probing and observation of components assumed.
> Software: access to operating environment, applications and data assumed.

User Assumptions:

> Correct operation of the *approved* cryptographic services and security functions.
> All attacks result in access to CSPs and data (plaintext and ciphertext) held within the module.
> Operator is responsible for the physical protection of the module.

*Value or sensitivity of data protected by the module is assumed negligible in an unprotected environment.

Attack Type:

Passive attack to gain immediate access to CSPs and data held by the module.

Attack Characterization/Testing Assumptions:

No prior access to the module is assumed.
No tools and materials are assumed needed.

Value:

The module provides correct operation of security functions and services. Protection of the plaintext CSPs and data held within the module is provided by the operator of the module (e.g. the environment the module may be used). If the module is used in an unprotected environment, then the module should not hold or maintain unprotected plaintext CSPs or data.

Level 2

Protection Provided:

Observable evidence of tampering.
Physical boundary of the module is opaque to prevent direct observation of internal security components.
Hardware: probing is assumed.
Software: logical access protection of the cryptographic modules unprotected CSPs and data is provided by the evaluated operating system at EAL2.

User Assumptions:

Correct operation of the *approved* cryptographic services and security functions.
All attacks result in access to CSPs and data (plaintext and ciphertext) held within the module.
Operator is responsible for the physical protection of the module.

*Value or sensitivity of data protected by the module is assumed low in an unprotected environment.

Attack Type:

Active attack to gain immediate access to CSPs and data held by the module.

Attack Characterization/Testing Assumptions:

No prior access to the module is assumed.
Readily available low cost tools and materials which are on hand at time of attack.
Attack time is assumed to be low.

Value:

The module provides correct operation of security functions and services. Protection of the plaintext CSPs and data held within the module is provided by the operator of the module (e.g. the environment the module may be used). The operator of the module is aware by tamper evidence that internal information may be compromised. If the module is used in an unprotected environment, then the module should not hold or maintain unprotected plain-text CSPs or data which have a moderate or high value.

Level 3

Protection Provided:

Observable evidence of tampering.
Physical boundary of the module is opaque to prevent direct observation of internal security components.
Direct entry/probing attacks prevented.
Strong tamper resistant enclosure or encapsulation material.
If applicable, active zeroization if covers or doors opened.
Software: logical access protection of the cryptographic modules unprotected CSPs and data is provided by the evaluated operating system at EAL3.

User Assumptions:

Correct operation of the *approved* cryptographic services and security functions.
Non-direct attacks result in access to CSPs and data (plaintext and ciphertext) held within the module.

*Value of data protected by the module is assumed moderate in an unprotected environment.

Attack Type:

Moderately aggressive attack to gain immediate access to CSPs and data held by the module.

Attack Characterization/Testing Assumptions:

Prior access to or basic knowledge of the module is assumed.
Readily available tools and materials.
Actual attack time is assumed to be moderate (this does not include time spend gaining prior access or basic knowledge of module).

Value:

The module provides correct operation of security functions and services. Protection of the plaintext CSPs and data held within the module is provided by the operator of the module (e.g. the environment the module may be used) and by the physical protection mechanisms of the module (e.g. strong enclosure, tamper response for covers and doors, deterrent of probing). The operator of the module is aware by tamper evidence that internal information may be compromised. An attack is pre-meditated but will be of moderate difficulty. If the module is used in an unprotected environment, then the module should not hold or maintain unprotected plain-text CSPs or data which have a high value.

Level 4

Protection Provided:

Observable evidence of tampering.
Physical boundary of the module is opaque to prevent direct observation of internal security components.
Direct entry/probing attacks prevented.
Strong tamper resistant enclosure or encapsulation material.
If applicable, active zeroization if covers or doors opened.
A complete envelope of protection around the module preventing unauthorized attempts at physical access.

Penetration of the module's enclosure from any direction had a very high probability of being detected resulting in immediate zeroization of plaintext CSPs or severe damage to the module rendering it inoperable.

Non-direct attacks prevented.

Software: logical access protection of the cryptographic modules unprotected CSPs and data is provided by the evaluated operating system at EAL4.

User Assumptions:

Correct operation of the *approved* cryptographic services and security functions.

Module is tamper resistant against all physical attacks defined in the standard.

*Value of data protected by the module is assumed high in an unprotected environment.

Attack Type:

Aggressive attack to gain immediate access to CSPs and data held by the module.

Attack Characterization/Testing Assumptions:

Prior access to or advanced knowledge of the module is assumed.

Specialized tools and materials.

Temperature and voltage attacks.

No time restriction on attack.

Value:

The module provides correct operation of security functions and services. Protection of the plaintext CSPs and data held within the module is provided by the operator of the module (e.g. the environment the module may be used) and by the physical protection mechanisms of the module (e.g. strong enclosure, tamper response for covers and doors, complete envelope of protection and penetration detection resulting in immediate zeroization of plaintext CSPs, voltage and temperature assurance). The operator of the module is aware by tamper evidence that the module was attacked. The module **shall** zeroize all unprotected CSPs before an attacker can compromise the module. An attack is pre-meditated, well-funded, organized and determined.

Additional Comments

*Discussion of the value of the data protected by the module does not consider physical protection provided by the operator to supplement the minimum physical security requirements of each level in FIPS 140-2. As an example, a user of Level 1 module may add "guards, guns, vaults and gates" surrounding the module and therefore may be comfortable in protecting more valuable information.

Attack times of low and moderate are subjective and depend on the experience and skill of an attacker and techniques employed. FIPS 140-2 Derived Test Requirements and FIPS 140-1 and FIPS 140-2 Implementation Guidance provide further guidance for the tester for each security level.

5.4 Level 3: Hard Coating Test Methods

Applicable Levels:	*Level 3*
Original Publishing Date:	*01/27/2010*
Effective Date:	
Last Modified Date:	*06/15/2010*
Relevant Assertions:	*AS.05.28, AS.05.39 and AS.05.52*
Relevant Test Requirements:	*TE05.28.02, TE05.39.06 and TE05.52.02*
Relevant Vendor Requirements:	

Background - References

AS.05.28: (Single-Chip - Levels 3 and 4) Either the cryptographic module shall be covered with a hard opaque tamper-evident coating (e.g., a hard opaque epoxy covering the passivation).

TE05.28.02: The tester shall verify that the coating cannot be easily penetrated to the depth of the underlying circuitry, and that it leaves tamper evidence. The inspection must verify that the coating completely covers the module, is visibly opaque, and deters direct observation, probing, or manipulation.

AS.05.39: (Multiple-Chip Embedded - Levels 3 and 4) the multiple-chip embodiment of the circuitry within the cryptographic module shall be covered with a hard coating or potting material (e.g., a hard epoxy material) that is opaque within the visible spectrum.

TE05.39.06: (Option 1 - Utilize a hard opaque material) The tester shall verify by inspection and from vendor documentation that the module is covered with a hard opaque material. The documentation shall specify the material that is used. The tester shall verify that it cannot be easily penetrated to the depth of the underlying circuitry. The tester shall verify that the material completely covers the module and is visibly opaque within the visible spectrum.

AS.05.52: (Multiple-Chip Standalone – Levels 3 and 4) the multiple-chip embodiment of the circuitry within the cryptographic module shall be covered with a hard potting material (e.g., a hard epoxy material) that is opaque within the visible spectrum.

TE05.52.02: (Option 1 – Covered with a hard opaque potting material) Encapsulate within a hard, opaque potting material. The tester shall verify from vendor documentation and by inspection, if internal access is possible, that the circuitry within the module is covered with a hard opaque potting material. The documentation shall specify which potting material is used and its hardness characteristics.

Question/Problem

What kind of testing is expected to be performed at Level 3 to verify that the hard coating or potting material that encapsulates the circuitry is *hard*?

Resolution

Within the scope of FIPS 140-2, the term *hard* is defined as:

Hard / hardness: the relative resistance of a metal or other material to denting, scratching, or bending; physically toughened; rugged, and durable. The relative resistances of the material to be penetrated by another object.

Test methods **shall** be consistent with IG 5.3 that addresses a *moderately aggressive attack* at Level 3.

The test methods **shall** at a minimum address the hardness characteristics of the epoxy or potting material as follows:

1. Attempts to penetrate the material by an instrument (e.g. awl, pointed handheld tool, etc.) using a *moderately aggressive* amount of force to the depth of the underlying circuitry. The use of a drilling or grinding motion is out-of-scope.

2. The use of an instrument with a *moderately aggressive* amount of force to pry or break the material away from the underlying circuitry (e.g. insert a pry instrument at the boundary of the epoxy or potting material and another material/component (e.g. PCB board)).

3. The use of a *moderately aggressive* amount of flexing or bending force to crack or break the material away from or expose the underlying circuitry.

During testing the module should be consistently assessed to determine if serious damage has occurred (i.e. the module will either cease to function or the module is unable to function).

The manufacturing method which is used to apply the epoxy or potting material **shall** be reviewed to determine if voids or pockets may exist that could create an exposure or weakness. The above testing **shall** exploit those areas.

Module hardness testing **shall** be performed at the vendors specified nominal operating temperature for the module and at the vendors specified lowest and highest temperature that the module will not be damaged (e.g. during storage, transportation/shipping, etc.). If no specification is provided, hardness testing **shall** be performed by the laboratory at ambient temperature.

The Security Policy **shall** (**AS.14.05**) specify the nominal and high/low temperature range that the module hardness testing was performed. If the module hardness testing was only performed at a single temperature (e.g. vendor provided only a nominal temperature or the vendor did not provide a specification), the Security Policy **shall** clearly state that the module hardness testing was only performed at a single temperature and no assurance is provided for Level 3 hardness conformance at any other temperature.

At Level 3, testing methods at all embodiments (single-chip, multi-chip embedded and multi-chip standalone) **shall not** consist of drilling, milling, cutting, burning, melting, grinding or dissolving the epoxy or potting material, in order to gain access to the underlying circuitry. These types of "attacks" are addressed by Level 4 physical security and are consistent with *FIPS 140-1 Implementation Guidance* IG 5.7.

Additional Comments

While the above test methods may be applicable at *Physical Security* Level 3 for a module which is protected by a strong enclosure or includes doors or removable covers, this IG does not specifically address those test methods.

5.5 Physical Security Level 3 Augmented with EFP/EFT

Applicable Levels:	*Level 3*
Original Publishing Date:	*12/23/2010*
Effective Date:	
Last Modified Date:	*12/23/2010*
Relevant Assertions:	*AS.05.60*
Relevant Test Requirements:	
Relevant Vendor Requirements:	

Background

AS.05.60: (Level 4) The cryptographic module shall either employ environmental failure protection (EFP) features or undergo environmental failure testing (EFT).

Question/Problem

EFP/EFT is a Level 4 Physical Security requirement. Can a module that only claims Level 3 physical security also claim EFP/EFT?

Resolution

A module that has been designed only to meet Level 3 physical security in FIPS 140-2 Section 4.5 can augment the Level 3 requirements with the Section 4.5 EFP/EFT requirements.

The CMVP provided test reporting tool (CRYPTIK) was modified to allow this scenario where FIPS 140-2 Section 4.5 is claimed at Level 3 and the "EFP/EFT" option is selected in the Module Information panel. This requires the testing laboratory to address both the Level 3 physical security requirements and the Level 4 EFP/EFT assertions while keeping the overall section annotated as Level 3.

As indicated in IG G.13, the validation certificate will be annotated as either:

-Physical Security: Level 3 +EFP
-Physical Security: Level 3 +EFT
-Physical Security: Level 3 +EFP/EFT

Section 6 – Operational Environment

6.1 Single Operator Mode and Concurrent Operators

Applicable Levels:	*All*
Original Publishing Date:	*03/10/2003*
Effective Date:	*03/10/2003*
Last Modified Date:	*04/24/2003*
Relevant Assertions:	*AS.06.04*
Relevant Test Requirements:	*TE06.04*
Relevant Vendor Requirements:	*VE.06.04*

Background

Historically, for a FIPS 140-1 and FIPS 140-2 validated software cryptographic module on a server to meet the single user requirement of Security Level 1, the server had to be configured so that only *one* user at a time could access the server. This meant configuring the server Operating System (OS) so that only a single user at a time could execute processes (including cryptographic processes) on the server. Consequently, servers were not being used as intended.

Question/Problem

AS.06.04 states: "(Level 1 Only) The operating system **shall** be restricted to a single operator mode of operation (i.e., concurrent operators are explicitly excluded)". What is the definition of concurrent operators in this context? Specifically, may Level 1 software modules be implemented on a server and achieve FIPS 140-2 validation? (Note: this question is also applicable to VPN, firewalls, etc.)

Resolution

Software cryptographic modules implemented in client/server architecture are intended to be used on both the client and the server. The cryptographic module will be used to provide cryptographic functions to the client and server applications. When a crypto module is implemented in a server environment, the server application is the user of the cryptographic module. The server application makes the calls to the cryptographic module. Therefore, the server application is the single user of the cryptographic module, even when the server application is serving multiple clients

Additional Comments

This information must be included in the non-proprietary security policy.

6.2 Applicability of Operational Environment Requirements to JAVA Smart Cards

Applicable Levels:	*All*
Original Publishing Date:	*04/08/2003*
Effective Date:	*04/08/2003*
Last Modified Date:	*09/11/2003*
Relevant Assertions:	*AS.06.01*
Relevant Test Requirements:	
Relevant Vendor Requirements:	

Background

FIPS 140-2 states (Section 4.6 Operational Environment) "A limited operational environment refers to a static non-modifiable virtual environment (e.g., a JAVA virtual machine on a non-programmable PC card) with no underlying general purpose operating system upon which the operational environment uniquely resides."

Question

Does the FIPS 140-2 statement mean that a smart card implementing a non-modifiable operating system (e.g., like the ones currently used today in most smart cards) that accept and run JAVA applets (whether validated or not) is a limited operational environment?

Resolution

The CMVP cannot issue a general statement that applies to all JAVA card modules since functionality and design can vary greatly from module to module. The determination is left to the CST laboratories, which have the complete module documentation available to them. In general**,** however, a JAVA smart card module with the ability to load unvalidated applets post-validation is considered to have a *modifiable* operational environment and the Operational Environment requirements of FIPS 140-2 are applicable.

A JAVA smart card module having a modifiable operational environment which either:

    a) is configured such that the loading of any applets is not possible, or

    b) loads only applets that have been tested and validated to either FIPS 140-1 or FIPS 140-2,

could be considered to have a *limited* operational environment and have the FIPS 140-2 Operational Environment requirements section of the module test report marked as *Not Applicable*.

The validated JAVA smart card cryptographic module must use an approved authentication technique on all loaded applets. The module **shall** also meet, at a minimum, the requirements of **AS.09.34**, **AS.09.35**, **AS.10.03** and **AS.10.04**, as well as any other applicable assertions. Validation of the cryptographic module is maintained through the loading of applets that have either been tested and validated during the validation effort of the smart card itself or through an independent validation effort (i.e., the applet itself has its own validation certificate number).

The security policy of the validated smart card module must state whether:

- The module can load applets post-validation, validated or not (Note: if the module can load non-validated applets post-validation, the security policy must clearly indicate that the module's validation to FIPS 140-1 or FIPS 140-2 is no longer valid once a non-validated applet is loaded);

- Any applets are contained within the validated cryptographic module and, if so, must list their name(s) and version number(s).

Additional Comments

The name(s) and version number(s) of all applets contained within a validated cryptographic module **shall** be listed on the module's certificate and CMVP website entry.

6.3 Correction to Common Criteria Requirements on Operating System

Applicable Levels:	*All*
Original Publishing Date:	*03/29/2004*
Effective Date:	*03/29/2004*
Last Modified Date:	*03/29/2004*
Relevant Assertions:	*AS.06.10, AS.06.21 and AS.06.27*
Relevant Test Requirements:	*TE06.10, TE06.21 and TE06.27*
Relevant Vendor Requirements:	*VE.06.10, VE.06.21 and VE.06.27*

Background

Depending on how assertions **AS.06.10**, **AS.06.21** and **AS.06.27** are read, they could be interpreted as the OS upon which the module is running on has to meet ALL of the listed PPs in Annex B at EAL2, EAL3 and EAL4 respectively. This is because of the plural at the end of the "Protection Profile**s**".

Question/Problem

Must the OS upon which the module is running on has to meet ALL of the listed PPs in Annex B at EAL2, EAL3 and EAL4 respectively?

Resolution

No, the requirements should be interpreted to read as follows:

- For **AS.06.10**:

  an operating system that meets the functional requirements specified in **a** Protection Profile listed in Annex B and is evaluated at the CC evaluation assurance level EAL2

- For **AS.06.21**, the first sentence:

  an operating system that meets the functional requirements specified in **a** Protection Profile listed in Annex B.

- For **AS.06.27**, the first sentence:

  an operating system that meets the functional requirements specified in **a** Protection Profile listed in Annex B.

6.4 Approved Integrity Techniques

Applicable Levels:	*All*
Original Publishing Date:	*01/21/2005*
Effective Date:	*01/21/2005*
Last Modified Date:	*01/21/2005*
Relevant Assertions:	*AS.06.08*
Relevant Test Requirements:	*TE06.01.01-02*
Relevant Vendor Requirements:	*VE.06.08.01*

Background

FIPS 140-2 Section 4.6.1 states that "A cryptographic mechanism using an approved integrity technique (e.g. approved message authentication code or digital signature algorithm) **shall** be applied to all cryptographic software and firmware components within the cryptographic module."

Question/Problem

What is an *approved integrity technique*, as specified in **AS.06.08**, and when must be it performed?

Resolution

An *approved integrity technique* is a keyed cryptographic mechanism that uses an approved and validated cryptographic security function. This includes a digital signature scheme, an HMAC or a MAC. Approved security functions are listed in FIPS 140-2 Annex A.

The approved integrity technique is considered a *Power-Up Test* and **shall** meet all power-up test requirements.

Section 7 – Cryptographic Key Management

7.1 moved to D.2

7.2 Use of IEEE 802.11i Key Derivation Protocols

Applicable Levels:	*All*
Original Publishing Date:	*01/21/2005*
Effective Date:	*01/21/2005*
Expiration Date:	
Last Modified Date:	*01/27/2010*
Relevant Assertions:	*AS.07.17*
Relevant Test Requirements:	*TE07.17.01-02*
Relevant Vendor Requirements:	*VE.07.17.01*

Background

FIPS 140-2 Annex D provides a list of the FIPS approved key establishment techniques applicable to FIPS PUB 140-2.

The commercially available schemes referred to in FIPS 140-2 Annex D are concerned with the derivation of a shared secret, or, as it is sometimes called, "the keying material." The IEEE 802.11i standard describes how to derive keys from a secret shared between two parties. It does not specify how to establish this commonly shared secret.

Question/Problem

Assuming that the shared secret is established using a key establishment technique specified in Annex D, can a cryptographic module use the 802.11i key derivation techniques to derive a data protection key, a key wrapping key and other keys for use in a FIPS approved mode of operation?

Resolution

Implementations of the IEEE 802.11i protocol operating in a FIPS approved mode of operation must meet the following requirements:

1. To derive a data protection key, a key wrapping key and other keys for use in a FIPS approved mode of operation, the following requirements **shall** be met:

    a) the shared secret (the keying material) **shall** be established using a FIPS approved method specified in FIPS 140-2 Annex D; and

    b) the key derivation function **shall** be implemented as defined IG 7.10.

2. The data protection method defined in the 802.11i protocol **shall** be AES CCM, which is an approved security function for use in a FIPS approved mode of operation as specified in FIPS 140-2 Annex A.

3. The keying material may be established via manual methods as specified in FIPS 140-2. The key derivation function as defined in IG 7.10 may then be applied.

References

Amendment 6: IEEE 802.11Medium Access Control (MAC) Security Enhancements, IEEE P802.11i/D10.0, April 2004. Section 8.5.1.2. Pairwise Key Hierarchy.

7.3 moved to C.2

7.4 Zeroization of Power-Up Test Keys

Applicable Levels:	*All*
Original Publishing Date:	*09/12/2005*
Effective Date:	*09/12/2005*
Last Modified Date:	*02/23/2007*
Relevant Assertions:	*AS.07.41*
Relevant Test Requirements:	*TE07.41.01-04*
Relevant Vendor Requirements:	*VE.07.41.01*

Background

FIPS 140-2 Section 4.7.6 states that "The cryptographic module **shall** provide methods to zeroize all Plaintext secret and private cryptographic keys and CSPs within the module."

Question/Problem

Are cryptographic keys used by a module ONLY to perform FIPS 140-2 Section 4.9.1 Power-Up Tests (e.g. cryptographic algorithm Known Answer Tests (KAT) or software/firmware integrity tests) considered CSPs and is zeroization required under FIPS 140-2 Section 4.7.6?

Resolution

Cryptographic keys used by a cryptographic module ONLY to perform FIPS 140-2 Section 4.9.1 Power-Up Tests are not considered CSPs and therefore do not need to meet the FIPS 140-2 Section 4.7.6 zeroization requirements.

7.5 Strength of Key Establishment Methods

Applicable Levels:	*All*
Original Publishing Date:	*11/23/2005*
Effective Date:	*06/29/2005*
Last Modified Date:	*05/10/2017*
Relevant Assertions:	*AS.07.19*
Relevant Test Requirements:	*TE07.19.01-02*
Relevant Vendor Requirements:	*VE.07.19.01*

Background

FIPS 140-2 **AS.07.19** states that "Compromising the security of the key establishment method (e.g., compromising the security of the algorithm used for key establishment) **shall** require as many operations as determining the value of the cryptographic key being transported or agreed upon."

SP 800-57, *Recommendation for Key Management – Part 1: General (Revised)* (March 2007), Section 5, Sub-Section 5.6.1, Comparable Algorithm Strength, contains Table 1, which provides comparable security strengths for the approved algorithms.

Table 1: Comparable Strengths				
Bits of security	**Symmetric key algorithms**	**FFC (e.g., DSA, D-H)**	**IFC (e.g., RSA)**	**ECC (e.g., ECDSA)**
112	3TDEA	$L = 2048$ $N = 224$	$k = 2048$	$f = 224\text{-}255$
128	AES-128	$L = 3072$ $N = 256$	$k = 3072$	$f = 256\text{-}383$
192	AES-192	$L = 7680$ $N = 384$	$k = 7680$	$f = 384\text{-}511$
256	AES-256	$L = 15{,}360$ $N = 512$	$k = 15{,}360$	$f = 512+$

1. Column 1 indicates the number of bits of security provided by the algorithms and key sizes in a particular row. Note that the bits of security are not necessarily the same as the key sizes for the algorithms in the other columns, due to attacks on those algorithms that provide computational advantages.

2. Column 2 identifies the symmetric key algorithms that provide the indicated level of security (at a minimum), where 3TDEA is specified in **SP 800-67**, and AES is specified in FIPS 197. 3TDEA is TDEA with three different keys.

3. Column 3 indicates the minimum size of the parameters associated with the standards that use finite field cryptography (FFC). Examples of such algorithms include DSA as defined in FIPS 186-4 for digital signatures, and Diffie-Hellman (DH) and MQV key agreement as defined in **SP 800-56A**, where L is the size of the public key, and N is the size of the private key.

4. Column 4 indicates the value for k (the size of the modulus n) for algorithms based on integer factorization cryptography (IFC). The predominant algorithm of this type is the RSA algorithm. RSA is specified in ANSI X9.31 and the PKCS#1 document. These specifications are referenced in FIPS 186-4 for digital signatures. The value of k is commonly considered to be the key size.

5. Column 5 indicates the range of f (the size of n, where n is the order of the base point G) for algorithms based on elliptic curve cryptography (ECC) that are specified for digital signatures in ANSI X9.62 and adopted in FIPS186-4, and for key establishment as specified in ANSI X9.63 and **SP 800-56A**. The value of f is commonly considered to be the key size.

For example, if a 256 bit AES is to be transported utilizing RSA, then k=15360 for the RSA key pair. A 256 bit AES key transport key could be used to wrap a 256 bit AES key.

For key strengths not listed in Table 2 above, the correspondence between the length of an RSA or a Diffie-Hellman key and the length of a symmetric key of an identical strength can be computed as:

If the length of an RSA key L (this is the value of k in the fourth column of Table 2 above), then the length x of a symmetric key of approximately the same strength can be computed as:

$$x = \frac{1.923 \times \sqrt[3]{L \times \ln(2)} \times \sqrt[3]{\left[\ln\left(L \times \ln(2)\right)\right]^2} - 4.69}{\ln(2)}$$

(1)

If the lengths of the Diffie-Hellman public and private keys are L and N, correspondingly, then the length y of a symmetric key of approximately the same strength can be computed as:

$$y = \min(x, N/2),$$

(2)

where x is computed as in formula (1) above.

Question/Problem

What does FIPS 140-2 assertion **AS.07.19** mean in the context of **SP 800-57**?

Resolution

The requirement applies to the key establishment methods found in FIPS 140-2 Section 4.7.

If a key is established via a key agreement or key transport method, the transport key or key agreement method **shall** be of equal or greater strength than the key being transported or established. For example, it is acceptable to have a 2048-bit RSA key (112-bit strength) transported using an AES key.

If the apparent strength of the largest key (taken at face value) that can be established by a cryptographic module is greater or equal than the largest comparable strength of the implemented key establishment method, then the module certificate and security policy will be annotated with, in addition to the other required caveats, the caveat "(Key establishment methodology provides xx bits of encryption strength)" for that key establishment method. For example, if a 256 bit AES is to be transported utilizing RSA with a value of k=2048 for the RSA key pair, the caveat would state "RSA (key wrapping, key establishment methodology provides 112 bits of encryption strength)".

Furthermore, if the module supports, for a particular key establishment method, several key strengths, then the caveat will state either the choice of strengths provided by the keys while operated in FIPS mode, if there are only two possible effective strengths, or a range of strengths if there are more than two possible strengths. For example, if a module implements 2048 and 3072-bit public key Diffie-Hellman with the private keys of 224 and 256 bits then the caveat would state "Diffie-Hellman (key agreement; key establishment methodology provides 112 and 128 bits of encryption strength)". The security policy **shall** provide details about the non-compliant key sizes. If, on the other hand, a module implements, in support of a key wrapping protocol, the RSA encryption/decryption with the RSA keys of 2048, 4096 and 15360 bits, then the caveat would say "RSA (key wrapping; key establishment methodology provides between 112 and 256 bits of encryption strength)". These caveats provide clarification to Federal users on the actual strength the module is providing even though Table 2 below states that the strength is sufficient.

Additional Comments

SP 800-57, *Recommendation for Key Management – Part 1: General (Revised)* (March 2007) also provides the following information in Section 5.6.2:

Table 2 provides recommendations that may be used to select an appropriate suite of algorithms and key sizes for Federal government unclassified applications. Between 2011 and 2030, a minimum of 112 bits of security **shall** be provided. Thereafter, at least 128 bits of security **shall** be provided.

1. Column 1 indicates the estimated time periods during which data protected by specific cryptographic algorithms remains secure. (i.e., the algorithm security lifetimes).

2. Column 2 identifies appropriate symmetric key algorithms and key sizes: 3TDEA are specified in **SP 800-67**, the AES algorithm is specified in **FIPS 197**, and the computation of Message Authentication Codes (MACs) using block ciphers is specified in **SP 800-38**.

3. Column 3 indicates the minimum size of the parameters associated with FFC, such as DSA as defined in **FIPS 186-4**.

4. Column 4 indicates the minimum size of the modulus for IFC, such as the RSA algorithm specified in **ANSI X9.31** and **PKCS#1** and adopted in **FIPS 186-4** for digital signatures.

5. Column 5 indicates the value of f (the size of n, where n is the order of the base point G) for algorithms based on elliptic curve cryptography (ECC) that are specified for digital signatures in ANSI X9.62 and adopted in **FIPS 186-4**, and for key establishment as specified in **ANSI X9.63** and **SP 800-56A**. The value of f is commonly considered to be the key size.

Table 2: Recommended algorithms and minimum key sizes				
Algorithm security lifetimes	**Symmetric key Algorithms (Encryption & MAC)**	**FFC (e.g., DSA, D-H)**	**IFC (e.g., RSA)**	**ECC (e.g., ECDSA)**
Through 2030 (min. of 112 bits of strength)	3TDEA AES-128 AES-192 AES-256	Min.: $L = 2048$ $N = 224$	Min.: $k=2048$	Min.: $f=224$
Beyond 2030 (min. of 128 bits of strength)	AES-128 AES-192 AES-256	Min.: $L = 3072$ $N = 256$	Min.: $k=3072$	Min.: $f=256$

The algorithms and key sizes in the table are considered appropriate for the protection of data during the given time periods. Algorithms or key sizes not indicated for a given range of years **shall** not be used to protect information during that time period. If the security life of information extends beyond one time period specified in the table into the next time period (the later time period), the algorithms and key sizes specified for the later time **shall** be used. The following examples are provided to clarify the use of the table:

a. If information is encrypted in 2005 and the maximum expected security life of that data is only five years, any of the algorithms or key sizes in the table may be used. But if the information is protected in 2005 and the expected security life of the data is six years, then 2TDEA would not be appropriate.

b. If a CA signature key and all certificates issued under that key will expire in 2005, then the signature and hash algorithm used to sign the certificate needs to be secure for at least five years. A certificate issued in 2005 using 1024 bit DSA and SHA-1 would be acceptable.

c. If information is initially signed in 2009 and needs to remain secure for a maximum of ten years (i.e., from 2009 to 2019), a 1024 bit RSA key would not provide sufficient protection between 2011 and 2019 and, therefore, it is not recommended that 1024 bit RSA be used in this case. It is recommended that the algorithms and key sizes in the "Through 2030" row (e.g., 2048 bit RSA) should be used to provide the cryptographic protection. In addition, the signature must be generated using a hash algorithm of comparable or greater strength, such as SHA-224 or SHA-256.

7.6 moved to W.5

7.7 Key Establishment and Key Entry and Output

Applicable Levels:	*All*
Original Publishing Date:	*01/24/2008*
Effective Date:	*01/24/2008*
Last Modified Date:	*02/06/2017*
Relevant Assertions:	*General*
Relevant Test Requirements:	
Relevant Vendor Requirements:	

Question/Problem

Given different configurations of cryptographic modules, how can a modules key establishment and key entry and output states be easily mapped to the FIPS 140-2 Section 4.2 *Cryptographic Module Ports and Interfaces*, Section 4.7.3 *Key Establishment* and Section 4.7.4 *Key Entry and Output*? Are there any special considerations for *Sub-Chip Cryptographic Subsystems* (IG **1.20**)?

Resolution

Using the following guidelines, first determine how keys are established to a module. Once the establishment method is determined, the Key Entry format table will indicate the requirements on how keys **shall** be entered or output. The following is based on the requirements found in FIPS 140-2 in Sections 4.2 and 4.7.

CM: a FIPS 140-2 *validated* Cryptographic Module

GPC: General Purpose Computer

EXT: a validated Cryptographic Module which lies external or outside of the physical boundary. See the CM software physical boundary diagram for an example.

INT: a validated Cryptographic Module which lies internal or inside of the physical boundary. See the CM software physical boundary diagram for an example.

App: a non-validated non-crypto general purpose software application operating inside of the boundary in regard to the reference diagrams CM software physical boundary.

Key Establishment – Table 1		
MD: Manual Distribution	**ME: Manual Entry (Input / Output)**	
ED: Electronic Distribution	**EE: Electronic Entry (Input / Output)**	
CM Software[1] from GPC Keyboard		**MD / ME**
CM Software[1] to/from GPC Key Loader (e.g., diskette, USB token, etc.)		**MD / EE**
CM Software[1] to/from GPC EXT Ports (e.g., network port)		**ED / EE**
CM Software[1] to/from CM Software[1] via GPC INT Path		**N/A**
CM Software[1] to/from App Software via GPC INT Path		**N/A**
CM Software[1] to/from INT CM Hardware via GPC INT Path		**N/A**
CM Software[1] to/from EXT CM Hardware running on a non-networked GPC (key loader)		**MD / EE**
CM Software[1] to/from EXT CM Hardware running on a networked GPC		**ED / EE**
INT CM Hardware to/from App Software via GPC INT Path		**ED / EE**
INT CM Hardware (Sub-Chip Cryptographic Subsystem) to/from INT CM Hardware (Sub-Chip Cryptographic Subsystem) via Single-Chip INT Path at Levels 1 and 2		**N/A**
INT CM Hardware (Sub-Chip Cryptographic Subsystem) to/from INT CM Hardware (Sub-Chip Cryptographic Subsystem) via Single-Chip INT Path at Levels 3 and 4		**ED / EE**
INT CM Hardware from GPC Keyboard via GPC INT Path		**MD / EE**
INT CM Hardware to/from direct attach key loader		**MD / EE**
INT CM Hardware from direct attach keyboard		**MD / ME**
EXT CM Hardware to/from networked GPC		**ED / EE**
EXT CM Hardware to/from directly attached key loader (a non-networked GPC could be considered and used as a key loader)		**MD / EE**
EXT CM Hardware from direct attach keyboard		**MD / ME**
[1] Must meet requirements of AS.06.04, AS.06.05 and AS.06.06 - These requirements cannot be enforced by administrative documentation and procedures, but must be enforced by the cryptographic module itself.		

The following illustration provides reference to the above Key Establishment table.
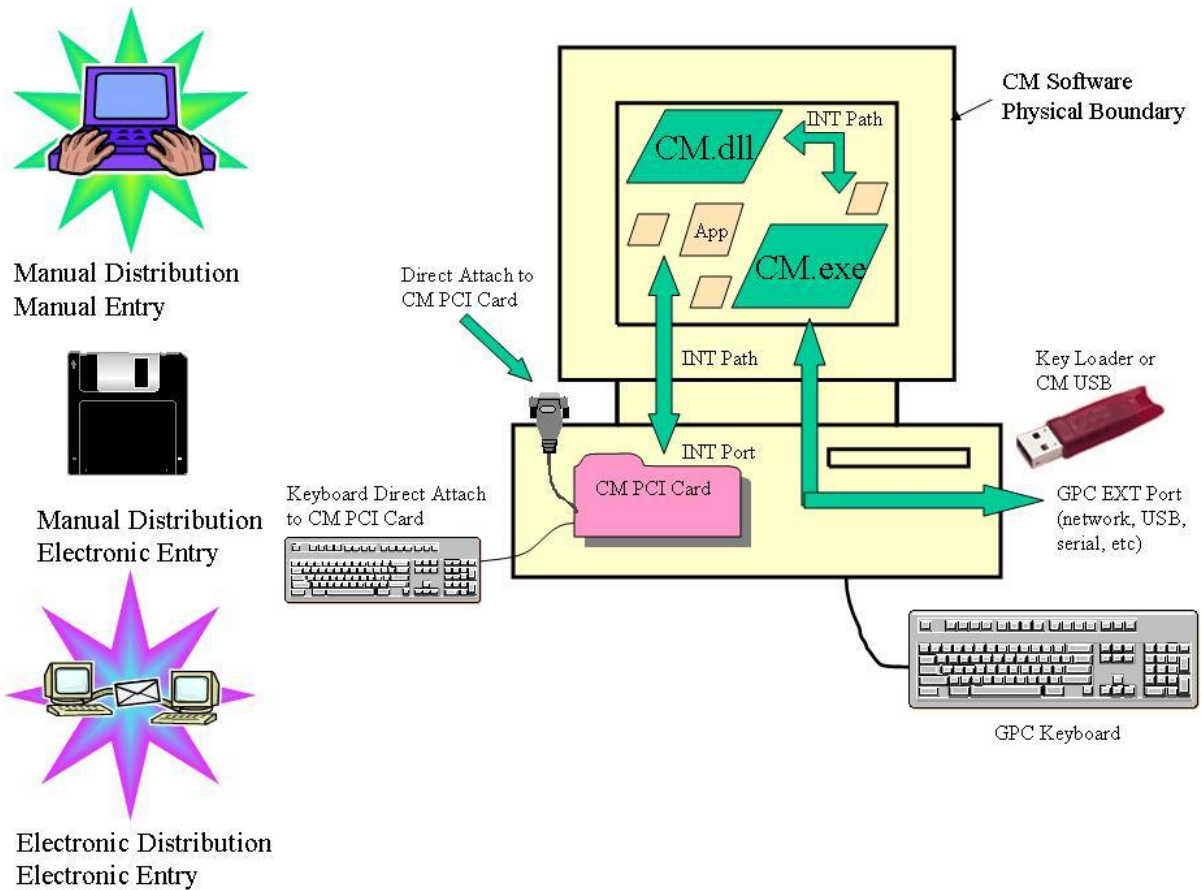
Key Entry Format – Table 2

			Distribution (Establishment)								
			Manual				Electronic				
Entry (Input / Output)	Manual	Keyboard, Thumbwheel, Switch, Dial									
		1	2	3	4						
		P/KT	P/KT	KT/SK	KT/SK						
	Electronic	Smart Cards, Token, Diskettes and Key Loaders				Key Establishment — Key Transport or Key Agreement					
		1	2	3	4	1	2	3	4		
		P/KT	P/KT	KT/SK	KT/SK	KE	KE	KE	KE		

Legend:
P/KT: May be Plaintext or by Key Transport
KE: Key Establishment
KT/SK: Key Transport or Plaintext Split Knowledge (via separated physical ports or via trusted path)

At Levels 3 and 4, plaintext key components may be entered either via separate physical ports or logically separated ports using a trusted path. Manual entry of plaintext keys must be entered using split knowledge procedures. Keys may also be entered manually using a key transport method. If automated methods, a key establishment method **shall** be used.

Additional Comments

This IG reaffirms that keys established using *manual transport methods* and *electronically input or output* to a cryptographic module may be input or output in <u>plaintext</u> at Levels 1 and 2.

Level 1 Software – General Purpose Operational Environment
AS.06.04: (Level 1 Only) The operating system shall be restricted to a single operator mode of operation (i.e., concurrent operators are explicitly excluded).

AS.06.05: (Level 1 Only) The cryptographic module shall prevent access by other processes to plaintext private and secret keys, CSPs, and intermediate key generation values during the time the cryptographic module is executing/operational. Processes that are spawned by the cryptographic module are owned by the module and are not owned by external processes/operators.

AS.06.06: (Leve.1 1 Only) Non-cryptographic processes shall not interrupt the cryptographic module during execution.

A Software Cryptographic Module (SCM) requires the use of an underlying General Purpose Computer (GPC) and Operational Environment (OE) to execute/operate. A SCM is conceptually comprised of two sub-elements: a Physical Cryptographic Module (PCM) and the Logical Cryptographic Module (LCM) boundary. The LCM is executes/operates within the PCM. The LCM is the collection of executable code that encompasses the cryptographic functionality of the SCM (e.g., dll's,. exe's). Other general-purpose application software (App) (e.g., word processors, network interfaces, etc.) may reside within the PCM. Therefore, the PCM encompasses the following elements: GPC, OE, LCM and App. The LCM relies on the OE and GPC for memory management, access to ports and interfaces, and other services such as the requirements of AS.06.04, AS.06.05 and AS.06.06. The LCM has no operational control over other App elements within the PCM of the SCM. The SCM, which is comprised of all the various sub-elements (GPC, OE, LCM and App), is restricted to a single operator mode of operation, such that the single operator has a level of confidence in the SCM environment as a whole. The CMVP views the non-LCM elements (GPC, OE and App) as implicitly excluded.

Example: If the LCM generates keys, it must use a FIPS approved RNG. That key may be stored within the PCM but must meet **AS.06.05** unless the LCM wishes the key to be exported. If exported, refer to Table 1 for the key establishment and key entry requirements. If a key is generated outside of the LCM, then the generation method is out-of-scope but the key must be imported per Table 1 requirements.

It is the burden of the operator of the SCM to understand the environment the SCM is running. If that environment is not acceptable, then there are alternative solutions (hardware cryptographic modules and/or Level 2, 3 or 4 software cryptographic modules) that should be considered.

If the operating system requirements of **AS.06.04**, **AS.06.05** and **AS.06.06** cannot be met, then the SCM cannot be validated at Level 1. The vendor provided documentation **shall** indicate how these requirements are met (**AS.14.02**). These requirements cannot be enforced by administrative documentation and procedures, but must be enforced by the cryptographic module itself.

7.8 The Use of Post-Processing in Key Generation Methods

Applicable Levels:	*All*
Original Publishing Date:	*03/10/2009*
Effective Date:	*03/10/2009*
Last Modified Date:	*08/01/2016*
Relevant Assertions:	*AS.07.11 and AS.07.16*
Relevant Test Requirements:	
Relevant Vendor Requirements:	

Background

FIPS 140-2 Section 4.7.2 states that "… approved key generation methods are listed in Annex C to this standard. If an approved key generation method requires input from an RNG, then an approved RNG that meets the requirements specified in FIPS 140-2 Section 4.7.1 **shall** be used."

Question/Problem

There exists a NIST standard for key generation, **SP 800-133**. This standard, however, does not include the so-called post-processing, which, instead, has been documented in the FIPS 140-2 IG 7.8. The post-processing has been used very infrequently by the vendors. The remaining key generation methodology is adequately addressed in **SP 800-133**.

Why have two separate documents (IG 7.8 and **SP 800-133**) to illustrate an almost identical functionality?

Resolution

The vendor has an option to perform a qualified post-processing that would apply to U, an output of an approved DRBG, before the updated value of U is passed to the **SP 800-133**-compliant portion of the key generation process. The post-processing methodology is not shown in **SP 800-133** and, therefore, not addressed in IG D.12.

Qualified Post-Processing Algorithms

The value of U in the **SP 800-133** key generation mechanism is the output of an approved DRBG. As explained earlier, this DRBG output may be further modified by applying a qualified post-processing algorithm *before* it is used to compute the secret value K. When post-processing is performed on DRBG output, the output of the post-processing operation **shall** be used in place of any use of the DRBG output. This output from the post-processing operation becomes the new U.

Let M be the length of the output requested from the DRBG by a consuming application, and let R_M be the set of all bit strings of length M. When the output is to be used for keys, M is typically a multiple of 64; however, these algorithms are flexible enough to cover any output size. Let R_N be the set of all bit strings of length N, and let F: $R_N \rightarrow \{0,1, \ldots , k\text{-}1\}$ be a function on N-bit strings with integer output in the range 1 to k, where k is an arbitrary positive integer. Let $\{P_1, P_2, \ldots, P_k\}$ be a set of permutations (one-to-one functions) from R_M back to R_M. The P_j's may be fixed, or they may be generated using a random seed or secret value. Examples of F and P_i are given below.

Let r_1 be randomly selected from the set R_N (i.e., r_1 is a random N-bit value), and let r_2 be randomly selected from the set R_M (i.e., r_2 is a random M-bit value). Both r_1 and r_2 **shall** be outputs from an approved DRBG, such that $N \leq M$. (The case $r_1 = r_2$ is permissible.) The post processor's output is the M-bit string $P_{F(r_1)}(r_2)$.

The apparent complexity of this post-processing should not be of any concern to vendors and testing laboratories. The post-processing step is optional. Vendors are not encouraged to design the post-processing into the cryptographic modules.

Examples of F(r_1) used for Post Processing

The function F may be simple or fairly complex.

Let k be the number of desired permutations, and let r_1 represent an N-bit output of an approved DRBG. Two examples are provided:

1. A very simple example of a suitable F is the following, where k is assumed to be an integer in the range 1 to 2^N.

$$F(r_1) = r_1 \bmod k.$$

   Here, r_1 is interpreted as an integer represented by the bit string r_1.

2. A more complex example is:

$$F(r_1) = \text{HMAC}(key, r_1) \bmod k,$$

   using a hashing algorithm and a fixed key in the HMAC computation. In this case, k could be as large as 2^{outlen}, or as small as 1, where *outlen* is the length of the hash function output in bits. (Having a single permutation, while permitted, would certainly not require the use of a keyed hash to "choose" it. On the other hand, $k = 2$ might make sense in the right application.)

Note that in both of these examples, the k permutations are selected with (nearly) equal probability, but this is not a requirement imposed by this post-processing algorithm.

Examples of P_i used for Post-Processing.

Depending on the requirements of the application, the P_i may be very simple or quite complex. The security of the key generation method depends on the P_i being *permutations*.

1. An example of a very simple permutation P_i is bitwise XOR with a fixed mask A_i: $P_i(r_2) = (r_2 \text{ XOR } A_i)$, where r_2 and A_i are M-bit vectors. Continuing this example, if there are four such masks ($k = 4$), the simple function $F(r_1)$ that maps r_1 into an integer represented by the two rightmost bits of r_1 (say, '01' corresponds to 1, '02' corresponds to 2, '03' corresponds to 3, and '00' corresponds to 4) could be used to choose among them. Then the post-processor's output $P_{F(r_1)}(r_2)$ would be r_2 XOR

   $A_{F(r_1)}$. Note that in this example, $2 \le N \le M$, where N is the length of r_1, and M is the length of r_2.

   [This should not be confused with the XORing defined in equation (1) above. The equation in (1) is applied after each of the U and V values is calculated, including any qualified post-processing, if applicable.]

2. A more complex example would be the use of a codebook to affect a permutation. For example, $P_i(r_2) = $ Triple-DES(key_i, r_2) could be used on a DRBG whose outputs were 64-bit strings. Similarly, $P_i(r_2) = $ AES(key_i, r_2) could be used to effect permutations on a DRBG with 128-bit outputs.

   Suppose that there are ten 256-bit AES keys ($k = 10$). Let $F(r_1) = $ SHA256(r_1) mod 10. Then the post-processed output $P_{F(r_1)}(r_2)$ would be AES($key_{\text{SHA256}(r1) \bmod 10}$, r_2). Note that in this case, $4 \le N \le M$, where N is the length of r_1, and M is the length of r_2 (the minimum length of r_1 is determined by the modulus value 10, which is represented in binary as 4 bits).

   A similar example, but one with a *much* larger value for k, (e.g., $k = 2^{128}$), might use $key_i = $ SHA256(128-bit representation of i). Let $F(r_1) = $ SHA256(r_1). The output $P_{F(r_1)}(r_2)$ of the post-processor would be AES(SHA256(r_1), r_2). Note that is this case, $N = M = 128$.

3. An example of a permutation somewhere between these extremes of complexity is a byte-permutation 'SBOX$_i$', which will be applied to each byte of input, with the final output being the concatenation of the individually permuted bytes:

$$P_i(B_1\|B_2\| \ldots \|B_{M/8}) = SBOX_i(B_1)\|SBOX_i(B_2)\|\ldots\|SBOX_i(B_{M/8})$$

For specificity, suppose that $M = 128$; there are just 2 byte permutations to choose from, $SBOX_0$ and $SBOX_1$; and F maps 8-bit strings to their parity: $F(r_1) = 0$ if r_1 has an even number of 1's, and $F(r_1) = 1$ if r_1 has an odd number of 1's. Note that in this case, $N = 8$.

The post-processor's output $P_{F(r_1)}(r_2)$, on the input pair r_1 and $r_2 = B_1\|B_2\| \ldots \|B_{16}$ would be $SBOX_{parity(r1)}(B_1) \| SBOX_{parity(r1)}(B_2) \|\ldots\| SBOX_{parity(r1)}(B_{16})$. To complete the example, suppose that the two byte permutations are specified as: $SBOX_0$ = the AES SBOX, and $SBOX_1$ is the inverse permutation to the same AES SBOX.

Additional Comments

1. If the vendor chooses to perform the post-processing, the vendor **shall** explain the details of how it works. If possible, the vendor should map their method into one of the examples shown in this Implementation Guidance.

2. Although some security strength may be lost during post-processing, the loss is small enough to be ignored for the purposes of FIPS 140-2 validation.

3. The post-processing may apply whenever the module generates either a symmetric cryptographic key or a seed to be used when generating the asymmetric keys.

Test Requirements

Code review, vendor documentation review, and mapping of the module's post-processing procedures into the methods described in this Implementation Guidance.

7.9 Procedural CSP Zeroization

Applicable Levels:	*All*
Original Publishing Date:	*03/24/2009*
Effective Date:	*03/24/2009*
Last Modified Date:	*03/24/2009*
Relevant Assertions:	*AS.07.41, AS.07.42*
Relevant Test Requirements:	
Relevant Vendor Requirements:	

Background

FIPS 140-2 Section 4.7.6 states "A cryptographic module **shall** provide methods to zeroize all plaintext secret and private cryptographic keys and CSPs within the module."

Question/Problem

A module **shall** provide methods to zeroize all plaintext permanent, temporary and ephemeral CSPs within the module. These methods may be operational (i.e. a callable service invoked by the operator of a module), or methods commonly referred to as *procedural zeroization* methods. What are acceptable methods?

Resolution

The zeroization methods required in **AS.07.41** are operational or procedural methods that will provide an operator of a module a method to zeroize all permanent, temporary and ephemeral plaintext CSPs. This **shall** be done with a level of assurance that the CSPs cannot be easily recovered. However, this **shall** not include

methods of recovery that require substantial skill and methods that may be employed by governmental or other well-funded institutions. As an operational or procedural method, the time necessary to perform the zeroization **shall** be reasonable based on the method employed.

- o For software modules, a procedural method may include the uninstallation of the cryptographic module application, *and* reformatting of and overwriting, at least once, the platform's hard drive or other permanent storage media. Only performing the procedural uninstallation of the cryptographic module application is not an acceptable method.

- o For space-based modules, a procedural method that relies on the de-orbit destruction is acceptable *only if* the vendor of the module provides analysis that indicates the components where plaintext CSPs may reside have a high probability of destruction and non-recovery.

- o All procedural or operational zeroization methods **shall** be performed by the operator of the module while the operator is in control of the module (i.e. present to observe the method has completed successfully or controlled via a remote management session). If the method is not under the direct control of the operator, then rationale **shall** be provided on how the zeroization method(s) are employed such that the secret and private cryptographic keys and other CSPs within the module cannot be obtained by an attacker.

- o Except for space-based modules, physical destruction of the module is not considered an acceptable zeroization method.

Additional Comments

TE07.41.03: is revised as follows:

TE07.41.03: The tester **shall** initiate zeroization and verify the key destruction method is performed in a sufficient time that an attacker cannot access plaintext secret and private cryptographic keys and other plaintext CSPs while under the direct control of the operator of the module (i.e. present to observe the method has completed successfully or controlled via a remote management session). If the method is not under the direct control of the operator, then rationale **shall** be provided on how the zeroization method(s) are employed such that the secret and private cryptographic keys and other CSPs within the module cannot be obtained by an attacker.

7.10 Using the SP 800-108 KDFs in FIPS Mode

Applicable Levels:	*All*
Original Publishing Date:	*10/22/2009*
Effective Date:	*10/22/2009*
Last Modified Date:	*10/22/2009*
Relevant Assertions:	*AS.07.11 and AS.07.16*
Relevant Test Requirements:	
Relevant Vendor Requirements:	

Background

When a key is shared between two entities, it may be necessary to derive additional keying material using the shared key. **SP 800-108** provides Key Derivation Functions (KDFs) for deriving keys from a shared key; in **SP 800-108**, the shared key is called a pre-shared key. The shared key may have been generated, entered or established using any method approved or allowed in FIPS mode.

Note that IG D.2 contains key establishment methods, and includes KDFs that are used during key agreement to derive keying material from a shared secret, which is the result of applying a Diffie-Hellman or MQV primitive. The keying material may be used as a key directly or to derive further keying material.

IG 7.2 defines IEEE 802.11i KDFs that may be used to derive further keying material.

Question/Problem

Where do the KDFs from **SP 800-108** fit in the key establishment process, and under what conditions can these KDFs be used in FIPS mode? Are there any other allowed methods for deriving additional keys from a pre-shared key?

Resolution

All key derivation methods listed in **SP 800-108** will be allowed in FIPS mode if the Key Derivation Key K_I, as introduced in Section 5 of **SP 800-108** has been generated, entered or established using any method approved or allowed in FIPS mode.

Note that the KDFs described IG 7.2 are included in **SP 800-108**, thus making IG 7.2 obsolete.

Other KDFs that are allowed for key derivation from shared keying material are:

1. The KDF specified in the Secure Real-time Transport Protocol (SRTP) defined in RFC 3711.

Additional Comments

A key hierarchy as specified in Section 6 of **SP 800-108** may be used.

7.11 Moved to W.6

7.12 Key Generation for RSA Signature Algorithm

Applicable Levels:	*All*
Original Publishing Date:	*05/02/2012*
Effective Date:	
Transition End Date:	*12/31/2013*
Last Modified Date:	*01/11/2016*
Relevant Assertions:	*AS07.16*
Relevant Test Requirements:	*TE07.16.01-02*
Relevant Vendor Requirements:	*VE07.16.01-02*

Background

FIPS 140-2 Annex A lists the approved security functions for FIPS 140-2. For asymmetric key digital signature standards, references address RSA signature generation, verification and key generation. Some of these referenced RSA standards include the specification of the RSA key generation procedure while others, such as RSASSA-PKCS1-v1_5 and RSASSA-PSS only define the requirements for signature generation and verification. These latter references do not address the generation of keys used in signature generation and verification.

Question/Problem

What methods for RSA key generation may be used when the module claims compliance with the RSA signature standards that do not explicitly address an RSA key generation method?

Resolution

If the module performs signature verification only, then the module does not need to possess a private RSA key and therefore does not need to generate it. The RSA public key parameters might be entered into the module or loaded at the time of manufacturing.

If the module performs an RSA Signature generation then the RSA private and public keys may either be loaded into the module (externally or pre-loaded at the time the module is manufactured) or generated by the module. If the module generates RSA signature keys then this key generation procedure **shall** be an approved method. The approved methods are described in **FIPS 186-4** or **ANSI X9.31**. The module's RSA Signature CAVP algorithm certificate **shall** indicate that the RSA key generating algorithm has been tested and validated for conformance to the methods in **FIPS 186-4** or **ANSI X9.31**.

Additional Comments

The Transition End Date is based on IG G.15 *FIPS 186-2 to FIPS 186-4 Validation Transition Plan* Clause 2.2.b: *Conformance to FIPS 186-2* after December 31, 2013.

This Implementation Guidance does not address RSA key generation for use in the approved key establishment protocols. The user should follow the requirements of **SP 800-56B**.

7.13 Moved to W.1

7.14 Entropy Caveats

Applicable Levels:	*All*
Original Publishing Date:	*08/07/2015*
Effective Date:	*08/07/2015*
Last Modified Date:	*08/07/2015*
Relevant Assertions:	*AS.07.13*
Relevant Test Requirements:	*TE.07.13.01 and TE.07.13.02*
Relevant Vendor Requirements:	

Background

Section 4.7 of FIPS 140-2 states that **"***compromising the security of the key generation method (e.g., guessing the seed value to initialize the deterministic RNG) shall require as least as many operations as determining the value of the generated key***.**" **TE.07.13.02** further states that "*The tester **shall** determine the accuracy of any rationale provided by the vendor. The burden of proof is on the vendor; if there is any uncertainty or ambiguity, the tester **shall** require the vendor to produce additional information as needed.*"

There are some module designs where it may be impossible to know how much entropy has been supplied for key generation. For example, a module designed as a software library with an API allowing the caller to supply random buffer to use as a seed for random number generation, the module would be passively accepting the entropy "infusions" from third-party applications. From such module's perspective, it is only possible to talk about the number of bytes/bits size of the received random filed, not of the amount entropy in it. Does it mean that the requirement in AS.07.13 cannot be tested and therefore the module cannot be validated?

To be fair, in this case the module is not necessarily non-compliant with AS.07.13; it is just impossible to determine within the scope of the CST lab testing the module would be compliant in all possible deployments. This Implementation Guidance weighs this and similar issues and shows how to identify the cases when

compliance with that entropy requirements of FIPS 140-2 cannot be directly verified by the testing labs and how to inform the user of potential weakness or lack of assurance for the true strengths of the cryptographic keys generated by such modules.

Question/Problem

When is it necessary for the module to provide the evidence of the amount of generated entropy?

How to handle the case when the amount of generated entropy is sufficient to meet the minimum key strength requirement (112 bit) but not necessarily sufficient to account for an *apparent* strength of the generated keys?

What information **shall** the testing laboratory provide in the test report submitted to the CMVP? What information **shall** be included in the module's certificate and the Security Policy to indicate the various forms of compliance with the AS.07.13 requirement?

Resolution

We identify the main "logical" cases and for each case indicate whether the module can be validated and what certificate caveat, if any, **shall** be used.

1. The module is either generating the entropy itself or it is making a call to request the entropy from a well-defined source.

    Examples include
    (a) A hardware module with an entropy-generating NDRNG inside the module's cryptographic boundary.

    What is required: (i) the testing lab **shall** corroborate the entropy strength estimate as provided by the vendor, (ii) the Security Policy **shall** state the minimum number of bits of entropy generated by the module for use in key generation.

    If the amount of entropy used to generate the module's cryptographic keys employed in an approved mode is less than 112 bits then this module **cannot** be validated.

    If the amount of entropy used to generate the module's cryptographic keys is at least 112 bits while the module generates keys with an apparent cryptographic strength greater than the amount of the available entropy, the following caveat **shall** be included in the module's certificate: *The module generates cryptographic keys whose strengths are modified by available entropy*. The apparent cryptographic strength of a key is addressed under the Additional Comments below.

    (b) A software module that contains an approved RNG/DRBG that is seeded exclusively from one or more known entropy sources located within the operational environment inside the module's physical boundary but possibly outside the logical boundary. For instance, a software library on a Linux platform making a call to /dev/random for seeding its DRBG.

    What is required: (i) the testing lab **shall** corroborate the entropy strength estimate of the sources as provided by the vendor, (ii) the Security Policy **shall** state the minimum number of bits of entropy requested per each GET function call.

    If the amount of entropy used to generate the module's cryptographic keys employed in an approved mode is less than 112 bits then this module cannot be validated.

    If the amount of entropy used to generate the module's cryptographic keys is at least 112 bits while the module generates keys longer than the amount of available entropy, the following caveat **shall** be included in the module's certificate: *The module generates cryptographic keys whose strengths are modified by available entropy*.

(c) A software module that contains an approved RNG/DRBG that issues a GET command to obtain the entropy from a source located outside the module's physical boundary.

What is required: (i) the testing lab **shall** corroborate – to the extent it is possible, given that the entropy source is not subject to this module's testing and validation – the entropy strength estimate as provided by vendor, (ii) the Security Policy **shall** state the minimum number of bits of entropy requested per each GET function call, (iii) the following caveat **shall** be added to the module's certificate: *No assurance of the minimum strength of generated keys*.

If the claimed amount of obtained entropy used to generate the module's cryptographic keys employed in an approved mode is known to be less than 112 bits then this module cannot be validated.

If the amount of entropy used to generate the module's cryptographic keys can be at least 112 bits while the module generates keys with an apparent cryptographic strength greater than the amount of the available entropy, the following caveat **shall** be included in the module's certificate: *The module generates cryptographic keys whose strengths are modified by available entropy*.

2. The module is passively receiving the entropy while exercising no control over the amount or the quality of the obtained entropy.

Examples include:

(a) A hardware module with an approved RNG/DRBG inside the module's cryptographic boundary. The approved RNG/DRBG is either seeded via a seed loader from outside the module's cryptographic boundary or the seed is pre-loaded at factory.

What is required: (i) the Security Policy **shall** state the minimum number of bits of entropy believed to have been loaded and justify the stated amount (from the length of the entropy field and from any other factors known to the vendor), (ii) the following caveat **shall** be added to the module's certificate: *No assurance of the minimum strength of generated keys*.

If the amount of claimed entropy used to generate the module's cryptographic keys employed in an approved mode is known to be less than 112 bits then this module cannot be validated.

If the amount of entropy used to generate the module's cryptographic keys can be at least 112 bits while the module generates keys with an apparent cryptographic strength greater than the amount of the available entropy, the following caveat **shall** be included in the module's certificate: *The module generates cryptographic keys whose strengths are modified by available entropy*.

(b) A software module that contains an approved RNG/DRBG that receives a LOAD command (or its logical equivalent) with entropy obtained from either inside the operational environment within the physical boundary of the module or, via an I/O port, from an external source that is outside the physical boundary.

What is required: (i) the Security Policy **shall** state the minimum number of bits of entropy believed to have been loaded and justify the stated amount (from the length of the entropy field and from any other factors known to the vendor), (ii) the following caveat **shall** be added to the module's certificate: *No assurance of the minimum strength of generated keys*.

If the amount of entropy used to generate the module's cryptographic keys employed in an approved mode is *known to be* less than 112 bits then this module cannot be validated.

If the amount of entropy used to generate the module's cryptographic keys can be at least 112 bits while the module generates keys with an apparent cryptographic strength greater than the amount of the available entropy, the following caveat **shall** be included in the module's certificate: *The module generates cryptographic keys whose strengths are modified by available entropy*.

3. The module uses a *hybrid* approach to obtaining entropy for key generation. Some entropy is passively received while the module is exercising no control over the amount or the quality of the obtained entropy. Another portion of the entropy is obtained when the module is either generating the entropy by itself or is making a GET call to request the entropy from a well-defined source inside the module's physical boundary. For instance, a software library on a Linux platform may be making a call to /dev/random for seeding its DRBG while it is also providing an API allowing the calling application to supply an additional random buffer to use in seeding its DRBG.

What is required: The testing lab **shall** examine the design of seeding the DRBG from multiple sources and corroborate an entropy strength estimate as provided by vendor; the lab will need to understand the work of the NDRNG within the operational environment and be able to verify vendor's claim about the amount of entropy loaded into the software cryptographic module.

If the review of the design of seeding the DRBG reveals that the entropy data obtained passively can **only** add to the entropy obtained actively and the module will block the seeding until a minimal threshold amount of actively obtained entropy is reached, then

The Security Policy **shall** state the minimum number of bits of entropy that can be guaranteed to be actively obtained and, in addition, **shall** state the number of bits believed to have been loaded, and justify the stated amounts (from the lengths of the entropy fields and from any other factors known to the vendor).

If between the active and passive entropy calls the module cannot possibly accumulate at least 112 bits of entropy when generating cryptographic keys then this module cannot be validated.

If the amount of entropy obtained actively *may be* less than 112 bits then the following caveat **shall** be added to the module's certificate: *No assurance of the minimum strength of generated keys*.

If the amount of entropy obtained actively to generate the module's cryptographic keys *is known to be* at least 112 bits and the module generates keys with an apparent strength greater than the amount of the available actively obtained entropy, then the following caveat **shall** be included in the module's certificate: *The module generates cryptographic keys whose strengths are modified by available entropy*.

If the review of the design of the DRBG seeding reveals that the entropy data obtained passively can preempt the seeding of the DRBG in a way that causes the module to unblock the seeding even when the minimal threshold amount of entropy obtained actively has not been reached at any time when the caller uses the API for supplying the passive data, then

The Security Policy **shall** state the minimum number of bits of entropy believed to have been loaded and justify the stated amount (from the length of the entropy field and from any other factors known to the vendor).

If the module cannot possibly accumulate at least 112 bits of entropy when generating cryptographic keys then this module cannot be validated.

The following caveat **shall** be added to the module's certificate: *When entropy is externally loaded, no assurance of the minimum strength of generated keys*.

Additional Comments

1. Unless the design of the module falls under the case for which a specific caveat is explicitly allowed under a particular scenario described in this Implementation Guidance, the vendor may not use the caveat. In particular, the vendor cannot use the No assurance of the minimum strength of generated keys caveat and get their module validated if the scenario that applies to this module requires an explicit estimation of the generated entropy.

2. If a software module's design requires entropy estimation then the module's Security Policy **shall** contain a statement that if porting to an untested platform is allowed then when running a module on such an untested platform the "*No assurance of the minimum strength of generated keys*" caveat applies regardless of what caveat, if any, is applicable to the original validation.

3. This implementation guidance only covers the applicability of entropy estimation and the way to document the amount of the available entropy. The actual methodology for entropy estimation is addressed in IG 7.15.

4. The "apparent" key strength referenced in this Implementation Guidance refers to the key strength corresponding to the length of the key alone, without taking into the consideration any other factors such the amount of the available entropy or the methodology used when generating or establishing this key.

   Thus an AES key has the apparent strength equal to its length; a three-key Triple-DES key has the apparent strength of 168 bits (even though there exist the man-in-the middle attacks that reduce the strength of this key to 112 bits); an RSA 2048 and 3072 private keys have the apparent strengths of 112 and 128 bits, correspondingly; a DSA or a Diffie-Hellman private key has the apparent strength of half of its bit length (even though the overall algorithm strength is largely determined by the size of the public key); an ECDSA or an EC Diffie-Hellman private key has the apparent strength of half of its bit length; an HMAC key has the apparent strength equal to its bit length.

Test Requirements

The vendor and tester evidence **shall** be provided under **TE.07.13.01** and **TE.07.13.02**.

7.15 Entropy Assessment

Applicable Levels:	*All*
Original Publishing Date:	*08/07/2015*
Effective Date:	*08/07/2015[1]*
Last Modified Date:	*11/12/2015*
Relevant Assertions:	*AS.07.13*
Relevant Test Requirements:	*TE.07.13.01 and TE.07.13.02*

---

[1] There are some cases of modules incorporating third-party hardware entropy sources that may not meet all documentation and test requirements set forth in this IG due to a lack of cooperation from the third-party vendor or other legal constraints. To allow adequate time to adapt to the documentation and test requirements in this IG to vendors that use third-party hardware sources, until December 31, 2016 the CMVP allows vendor-affirmation by the vendor of the module in lieu of full testing of the entropy source. The vendor-affirmation statement must be signed by a corporate officer of the company sponsoring the validation and contain an estimate of the assumed amount of entropy from the third-party and a stated assumption of residual security risks that may result from the incomplete testing of the third-party entropy source. The laboratory must include this vendor affirmation in the entropy report for the tested module. Note that the CMVP expects all laboratories and vendors to work in good faith to test the entropy sources fully and resort to this provision only in extreme cases. The CMVP reserves the right to consider a limited number of special cases by vendors who may be able to substantiate a hardship case as the result of the December 31, 2016 deadline. The CMVP will work with them on a case-by-case basis to minimize the negative impact.

Relevant Vendor Requirements:	

Background

Section 4.7 of FIPS 140-2 states that "*compromising the security of the key generation method (e.g., guessing the seed value to initialize the deterministic RNG) shall require as least as many operations as determining the value of the generated key.*" **TE.07.13.02** further states that "*The tester shall determine the accuracy of any rationale provided by the vendor. The burden of proof is on the vendor; if there is any uncertainty or ambiguity, the tester shall require the vendor to produce additional information as needed.*"

Note that the FIPS 140-2 standard is not asking to compare the length of the seed of a random number generator to the length of a generated key. The question is about comparing the *numbers of operations* that are required to guess the seed and to determine the key. These numbers depend on the amount of entropy produced by the source that generated the seed.

Question/Problem

As of the last modified date of this **IG**, standards do not yet exist for the embodiment or construction of an entropy source or the mechanisms to gather entropy.

As of the last modified date of this **IG**, test methods do not yet exist for determining the conformance of an entropy embodiment, construction or a gathering mechanism.

As of the last modified date of this **IG**, statistical methods to determine the conformance of an entropy embodiment, construction or a gathering mechanism have not been standardized.

The FIPS 140-2 DTR states the tester **shall** verify that the vendor provided documentation that provides rationale stating how compromising the security of the key generation method (e.g., guessing the seed value to initialize the deterministic RNG) **shall** require as least as many operations as determining the value of the generated key. The tester **shall** determine the accuracy of any rationale provided by the vendor.

What information **shall** the testing laboratory provide in the test report submitted to the CMVP? How should the tester determine the accuracy of any rationale provided by the vendor?

Resolution

This IG must be used together with IG 7.14 Entropy Caveats that shows various scenarios for reporting the relationship between the amount of gathered entropy and the apparent (that is, length-based) strength of the cryptographic keys established by the module. Depending on the applicable scenario, as explained in IG 7.14 Entropy Caveats, an entropy estimate may or may not be required. If entropy estimation is required, the testing laboratory and the vendor **shall** follow the directions given in this IG.

The testing laboratory **shall** provide the following documentation as a **PDF** addendum to the submitted test report to meet the requirements of **AS.07.13** and **AS.07.16**:
1. A detailed logical diagram **shall** illustrate all of the components, sources and mechanisms that constitute an entropy source. These components may include the Linear Feedback Shift Registers LFSRs, noisy diodes, thermal sampling, entropy service calls from other FIPS 140-2 validated modules, clock readings, memory cache hits, as well as various human-induced measurements, such as the time intervals between the keystrokes, mouse movements, etc.

2. Tester's arguments in support of the accuracy of vendor-provided rationale.

3. (Optional but *strongly* recommended) Results of statistical testing using an appropriate set of tests. The statistical testing may either be performed by the testing laboratory or by the vendor. The explanation of the test results **shall** include the assumptions that have been made, how many bits of data have been collected, what the p-value (or an equivalent parameter) of the test is, and what

numerical values were obtained to demonstrate that the test results supported the vendor provided rationale. Typically, it takes several statistical tests to obtain a reasonable estimate of entropy. Some tests establish the degree of confidence in the independence of the observed values. Other tests may examine the short and long runs of bits and again, check the behaviors of these runs for their consistency with the claimed properties of the tested source. NIST SP 800-22-rev1a and the current draft of NIST **SP 800-90B** *may* be used as informative guidance. The rationale **shall** be mathematically sound and consistent with vendor claims of the strengths of the generated cryptographic keys.

The CMVP will determine during the report review if the information provided in the testing laboratory's test report is acceptable. During the report review coordination process the testing laboratory may follow up with additional details to support the previously provided rationale.

This **IG** may be rescinded or modified when standards are published and conformance testing developed for entropy security strength testing. A suitable transition period will be granted to vendors.

Additional Comments

1. If the module is using a non-deterministic RNG approved for use in classified applications as allowed in Section 4.7.1 of FIPS 140-2 then provided entropy is assumed to provide N bits of entropy based on the length N of the entropy field (unless the vendor chooses to state that a smaller amount of entropy has been received).

2. Following are some examples of the heuristic analysis of entropy that the testing laboratory may perform:

   The vendor may say that 6 bits of entropy are gathered by measuring the time intervals between the human touches of the keyboard; 10 bits of entropy come from the decimal fraction in the value of the time of day when a certain event took place, another 10 bits come from the timing or frequency or another property of software interruptions measured by the module. These are all reasonable estimates for a wide range of devices although their validity can only be accepted by the CMVP in the context of the particular module being validated. If the time of day is measured, for example, every 3 seconds or less frequently, it can be argued that if this time is represented as hh:mm:ss.zzz, where zzz is the decimal fraction of a second measured up to the third decimal point (thus three "z" in the above expression), then the zzz values of different measurements are nearly independent and each can take 1,000 different values, thus yielding approximately 10 bits of entropy. The independence of clock measurements at different frequency is very important. The best case is when the module has different time sources, entirely independent down to the hardware. If the time measurements were taken every 0.5 seconds or so, then the three digit zzz values would not be independent and therefore the 10-bit entropy value could not be claimed. In this case, the CMVP would accept a claim of 7 bits of entropy. The reason is that if the time measurements are taken every 500 milliseconds as in this example, then the values made out of the second and third "z" after the decimal point are 'almost' independent (and there are 100 of them) and the first z has some randomness in it as well, so the resulting variability of the zzz values is somewhat similar to having 128 equally likely scenarios (100 plus a little more thanks to the first z) and this is leading to the 7 bits of entropy. The CMVP may even accept a claim of 8 bits of entropy in this case if a slightly more sophisticated argument is made to support such a claim.

   If the entropy is generated by a physical device, again a heuristic argument should be made. If this device is a radioactive isotope such that the average decay rate is known and the random value is the number of atoms that have decayed in a particular time period, the lab should state some known facts about the mean rate of the decay and also about either the distribution or at least about the variance of the number of the decaying atoms and give a rough estimate of the generated entropy. Note that in this scenario, not all outcomes (numbers of the decayed atoms) are equally likely, the values around the mean come with the highest probability, an IID claim (that the random variables are Independent and Identically Distributed) most likely cannot be made and therefore the vendor should either use the "min-entropy" estimate for the non-IID sources or come up with another reasonable and statistically sound estimate of generated uncertainty.

If the entropy is generated by oscillating rings, the vendor will need to explain the design of the random noise generator. The design description in http://csrc.nist.gov/groups/ST/rbg_workshop_2012/shankar.pdf may serve as an example. However, to compete the description of the entropy source from the referenced presentation, the vendor still needs to explain, at least heuristically, how the jitters are measured, how these measurements are used to generate the seed value for an approved RNG or DRBG and how much entropy the seed value carries. A special consideration **shall** be given to the speed of generating bits and the frequency of recording the results in claiming their independence.

If the RNG is reseeded frequently, the overall entropy increases if the lab can make a reasonable heuristic claim of the independence of the individual entropy values. Obviously, if the entropy comes from the minute value in the time of day and the module measures this time value every second, there is not much uncertainty in the minute field after the first measurement. The decaying isotope is, however, going to continue to decay independently (in some sense, and after adjusting by the number of the remaining atoms) of its history and therefore in this case the entropy values can be added without providing any further justification.

If the entropy is coming from an operational environment of the module, then again, some analysis should be made of the source of entropy. If this source is the /dev/random or the /dev/urandom function in one of the common operating system (OS), the justification of the generated entropy (possibly, provided by the vendor of the OS) will be required. In addition, the lab may refer to an independently published analysis of dev/random and dev/urandom. See, for example, "The Linux Pseudorandom Number Generator Revisited," Lacharme at al. 2012, (https://eprint.iacr.org/2012/251.pdf) .

The /dev/random justification is the easier of the two. This OS entropy source will satisfy a request for a random value only when it believes it has collected "enough" entropy; that is, when its own estimate of the collected entropy is such that a module's request can be met. For example, if the module needs to generate a 128-bit AES key and therefore the module requests 128-bits of entropy then the dev/random call would block until it is able to generate this much entropy. This, the module cannot generate the aforementioned AES key until enough entropy is gathered and the call to /dev/random returns.

In case of the /dev/urandom request, the call to this OS entropy generator is non-blocking. The data obtained from the non-blocking call is not guaranteed to possess the desired amount of entropy. How can the vendor provide the assurance that the requirements of the FIPS 140-2 AS.07.13 assertion are satisfied?

To meet these requirements, the vendor must first demonstrate that the initial call (that is, the first call after the module has been powered up or instantiated) to /dev/urandom returns the claimed amount of entropy. A possible way to achieve this is to analyze the sequence of events that precedes this initial call. If, for example, this sequence includes several restarts of the module and if each of these restarts includes several events that are measured and that provide the desired uncertainty, then a heuristic claim about the entropy in the initial call can be made. These events may include the times between the restarts, the measurements of an operator activity during the restarts (mouse clicks, etc.), the values stored in certain memory locations that are known to be unpredictable during the restarts. The events accumulated in one restart are accumulated to the events from previous restarts and persisted on the system for later use. This argument has a good chance of succeeding for the stand-alone modules; the embedded modules normally do not require multiple restarts so the use of dev/urandom in such modules is harder to justify.

If the vendor can justify having the desired amount of entropy returned on the first call to /dev/urandom, then the vendor can continue to claim that at least this much entropy (not necessarily independent of the initial entropy of the first call) is generated on each subsequent call. To see this, suppose that the OS collects a pool A with 256 bits of entropy prior to returning the first /dev/urandom request. Suppose that the request is returned in the form of E1 = SHA256(A). The module can then claim that the keys generated using the entropy received from the /dev/urandom call

possess 256 bits of entropy. If, however the request is returned in the form of E1 = SHA1(A), then E1 possesses only 160-bits of entropy.

Suppose now that the OS generates the entropy pool B between the first and the second /dev/urandom calls. The field returned by /dev/urandom to the module is E2 = SHA256(B‖SHA256(A)). (A particular implementation may use a different formula for E2, but again with a dependency on both B and SHA256(A).) As long as B is not an empty field and is not a function of SHA256(A) then regardless of the amount of entropy in B this returned field E2 contains at least 256 bits of entropy. Therefore, keys generated from the randomness in the second /dev/urandom call also possess at least 256 bits of entropy (not necessarily an independent entropy from the first call.) Similarly, if E2 = SHA1(B‖SHA1(A)) would result in E2 containing 160-bits of entropy.

Note that the entropy estimates in the above example cannot be added automatically. That is, because B and A are not necessarily independent, one cannot claim that E1 ‖ E2 contains more entropy than either E1 or E2 alone. if, A could only be shown to possess 128 bits of entropy and B could not be demonstrated to have any specific new entropy amount independent of A (a typical scenario when running /dev/urandom multiple times) then the entropy collected from E1 and E2 (that is, from the first and second calls to /dev/urandom) would only amount to 128 bits, not 256 bits.

3. Here is a possible way to estimate the generated min entropy that the CMVP will allow until a further notice. This is a dramatic simplification of one of the methods proposed in the current draft of **SP 800-90B**. This method of entropy estimation, if shown by the lab or the vendor to be applicable to a given module, would be allowed prior to the publication of **SP 800-90B** and during the transition period that would follow. At some point in the future, the CMVP would expect all vendors to comply with **SP 800-90B**.

This method would only apply if unprocessed (non-whitened) noise sources (and any conditioning components, if applicable) are IID (independent and identically distributed random variables). See Section 9.1.1 of the August 2012 draft of **SP 800-90B** or any Statistics textbook for an explanation of this notion. The sources do not have to produce the uniform distribution of the outcomes: the probabilities of different outcomes may be different. However, the probability distributions are identical between the sources (or between the different consecutive readings of each source's random output) and these probabilities do not depend on the outcomes of other events generated by these sources.

The August 2012 draft of **SP 800-90B** shows the sequence of statistical tests that would allow the vendor to test if the noise sources are indeed IID. These tests are quite complicated. Furthermore, if the tests support the IID assumption the draft **SP 800-90B** standard presents a complicated and, arguably, a very conservative method of estimating the min entropy.

The alternative this IG offers is for the vendor to present the heuristic arguments in favor of the IID assumption. Any reasonable argument will be considered by the CMVP and if the sources are truly IID it should not be difficult for the vendor and the lab to make such arguments.

Once the IID assumption has been accepted (or, in a more formal way, the IID hypothesis has not been rejected) the vendor may estimate the min entropy as follows (compare this to the algorithm in Section 9.2 of the August 2012 draft of **SP 800-90B**.)

Find the probability p_{max} of the most common outcome among all the possible events generated by the noise source. If this probability is already known, then use it. (Give the justification to why this probability is what it is claimed to be.) If p_{max} is not known, then, following the draft of **SP 800-90B**, take a dataset with N samples and count the occurrences of the most common value in the dataset. Again, following the draft of **SP 800-90B**, count the number of occurrences of this most common value in the dataset and denote the result C_{max}. Set $p_{max} = C_{max} / N$.

The **SP 800-90B** draft then tells how to establish the 99% confidence interval for p_{max} and then compute the min entropy estimate based on the upper bound of this confidence interval. However, at

this time the CMVP will accept a far less conservative and simpler-to-compute estimate of min entropy from the value of $p_{\max}$ itself. Simply set $H = -\log_2(p_{\max})$ and use this value H as the entropy. For example, if the most common event happens with the probability $1/2^{128}$, the estimated min entropy is 128 bits regardless of the probabilities of the occurrences of other less frequent events generated by the same source.

4. This IG applies to the generation of both symmetric cryptographic keys and seeds that serve as the starting points for the asymmetric algorithm key generation (such as the RSA keys). Once the analysis of the generated entropy has been made according to this IG, the TE.07.16.01 and TE.07.16.02 assessments, using **SP 800-133** or IG 7.8, **shall** show how the generated keys can be assured of possessing sufficient entropy to account for the target key strength.

Test Requirements

The vendor and tester evidence **shall** be provided under **TE.07.13.01** and **TE.07.13.02**.

7.16 Acceptable Algorithms for Protecting Stored Keys and CSPs

Applicable Levels:	*All*
Original Publishing Date:	*08/07/2015*
Effective Date:	*08/07/2015*
Last Modified Date:	*11/12/2015*
Relevant Assertions:	*AS07.21 13*
Relevant Test Requirements:	*TE07.21.01*
Relevant Vendor Requirements:	*VE07.21.01-02*

Background

Rules for key storage are described in some general terms in FIPS 140-2. The standard, however, does not list **any Approved or allowed methods for encrypting keys or CSPs stored within the cryptographic module.**

Question/Problem

In Section 4.7.5 of FIPS 140-2 it is stated that "cryptographic keys stored within a cryptographic module **shall** be stored either in plaintext form or encrypted form." What does this mean? The above statement may appear to indicate that there are no requirements on key storage inside the module. However, the zeroization requirement does apply to "all plaintext secret and private cryptographic keys and CSPs within the module." Keys and CSPs that are cryptographically protected are not plaintext and are exempt from this requirement. Therefore, it is necessary to know what constitutes, in this context, an acceptable "protection" of stored keys and CSPs.

In particular, it should be made clear whether the encryption of a stored key or a CSP using a symmetric-key-encryption algorithm such as AES or the Triple-DES needs to satisfy the same requirements that apply to the protection of the cryptographic keys that are transported in and out of the module. The latter requirements and methods of meeting them are described in **SP 800-38F**.

Resolution

Keys and CSPs may be stored within a module in any form – encrypted or unencrypted. To make a claim that keys and CSPs are stored encrypted, or, more precisely, "protected", the module **shall** protect them using one of the following algorithms:

- An AES or a Triple-DES encryption using any approved mode of AES or the Triple-DES as defined in Annex A of FIPS 140-2
- An RSA-based key encapsulation that may either comply with the requirements of **SP 800-56B** or be allowed by IG D.9.
- An approved hash algorithm for a CSP such as a password that does not need to be recovered but is used to check if it matches any other values.

The requirements of **SP 800-131A** for the encryption and key encapsulation key sizes apply if a stored key or CSP is claimed to be protected.

Additional Comments

1. Even though this guidance does not mandate the use of authenticated encryption algorithms from **SP 800-38F** it is *highly* recommended vendors adopt them because these algorithms are specifically designed to protect the confidentiality and the authenticity/integrity of cryptographic keys.

2. The AES and Triple-DES algorithm implementations used to protect stored keys and CSPs **shall** be tested by the CAVP.

3. It follows from this IG and IG D.9 that if the AES or the Triple-DES encryption is used then the requirements for encrypting stored keys and CSPs are different from those when keys are transported in and out of the module. It is, however, strongly recommended that the rules of IG D.9 are followed in this case as well.

   If an RSA-based key encryption (encapsulation) is used to protect keys and CSPs the requirements are the same regardless of whether or not the protected key or CSP leaves the module's boundary.

4. If the AES or the Triple-DES encryption is used to protect a stored key, the key encryption key may be established as shown in **SP 800-132**.

7.17 Zeroization of One Time Programmable (OTP) Memory

Applicable Levels:	*All*
Original Publishing Date:	*11/15/2016*
Effective Date:	*11/15/2016*
Last Modified Date:	
Relevant Assertions:	*AS.07.41*
Relevant Test Requirements:	*TE07.41.01-04*
Relevant Vendor Requirements:	*VE.07.41.01*

Background

The One Time Programmable (OTP) memory is a form of digital memory where the setting of each bit is locked by a fuse or antifuse. It provides a flexible, field-programmable alternative to Read Only Memory (ROM).

Question/Problem

If OTP is used within a module, how can the module meet FIPS 140-2 zeroization requirements? Are there specific zeroization requirements for OTP implementations?

Resolution

OTP can be used for storing plaintext secret and private cryptographic keys and CSPs within the module. However, the module **shall** be implemented in the following way in order to meet FIPS 140-2 zeroization requirements:

1. Given that the OTP should be writable during module operation, the module **shall** provide the operator with the ability to zeroize all of the keys and CSPs stored in OTP by overwriting the memory with 0s or 1s. This will likely decommission the module but will prevent attackers from gaining knowledge of secret data stored within the OTP.

2. After OTP keys have been zeroized, the module **shall** recognize the zeroized value as invalid and restrict the use to this value. This might be by using an additional bit that is flipped, or else code that knows the zeroization value is invalid such as an integrity value that is not correct after zeroization.

3. OTP storage of data is more likely than other types of data storage to have integrity values associated with the information. Therefore, any integrity value on the OTP **shall** be treated as a CSP (and thus subject to zeroization) <u>unless</u> the vendor demonstrates that it does not leak information about the original key. This follows the definition of a CSP in that a key is considered a CSP if disclosing or modifying it could compromise the security of the module. Therefore, if integrity values are stored on the OTP, the tester or vendor **shall** either have the ability to zeroize these values, or provide evidence to the CMVP that disclosure or modification of these values would not compromise the security of the module or the values in which the integrity values are protecting.

Additional Comments

Once data is fused onto the OTP, the process is irreversible. Therefore, keys and CSPs that are stored on the OTP are unlikely to be modified, written to, or stored during module operation. This makes the data that is on the OTP prior to module operation likely intended for long time term use and unchanged throughout the lifetime of the module. Regardless, the zeroization requirements explained in this IG applies.

Section 8 – Electromagnetic Interference/Electromagnetic Compatibility (EMI/EMC)

Section 9 – Self-Tests

9.1 Known Answer Test for Keyed Hashing Algorithm

Applicable Levels:	*All*
Original Publishing Date:	*02/10/2004*
Effective Date:	*02/10/2004*
Last Modified Date:	*03/27/2018*
Relevant Assertions:	*AS.09.07*
Relevant Test Requirements:	*TE09.07.01*
Relevant Vendor Requirements:	*VE.09.07.01*

Background

Several keyed hashing algorithms are FIPS-approved (e.g. HMAC-SHA-1, HMAC-SHA2, HMAC-SHA3, KMAC) and have different levels of complexity that determine the power-on Know-Answer-Test (KAT) requirements.

Question/Problem

What are the KAT requirements when implementing keyed hashing algorithms in FIPS mode?

Resolution

IG 9.11 has been published on August 7, 2017. This Implementation Guidance allows the user not to perform, under the conditions stated in IG 9.11, any of the KATs for the approved algorithms implemented by the module.   When the KATs are required by IG 9.11 then, to use any HMAC in the approved mode, the module **shall** perform at least one KAT of any approved HMAC implemented in the module to self-test all other approved forms of HMAC used by that module (this includes, HMAC-SHA-1, HMAC-SHA2 and HMAC-SHA3 with all approved SHA2 and SHA3 hash functions).

Even if the module is not taking advantage of the IG 9.11 provision allowing it not to perform the known-answer tests, the vendor may still consider the information in IG 9.1 when deciding if only one HMAC self-test needs to be performed. We show in the Additional Comments below that under our assumptions the HMAC known answer tests with different hash functions only help verify the correct padding in the HMAC and that the padding strings are not very different from each other, thus the additional HMAC self-tests add very little information about the health of the module.

The decision to perform only one HMAC KAT does not absolve the module from the requirement to self-test the different hash functions, as applicable. Please see IG 9.4 and the text in the Additional Comments below.

Additional Comments

1. As stated in IG 9.3, if an HMAC is used as the approved integrity technique to verify the software or firmware components as specified in **AS.06.08**, a KAT is not required for either the HMAC-*hash* or the underlying *hash*, where *hash* stands for any approved hashing algorithm.

2. The reason for requiring only one HMAC self-test when various approved forms of HMAC (that is, the HMACs using the different hash functions including SHA3) are employed by the module is that one HMAC KAT tests everything except for some details of the padding mechanisms pertaining to the HMACs using the other approved hash functions. For example, consider the module implementing both HMAC-SHA-256 and HMAC-SHA-1. Suppose the module performs an HMAC-

SHA-256(K,m) self-test. If this test passes, this assures (per IG 9.2) the health of the SHA-256 implementation. SHA-1 needs to be self-tested separately. If the module does not have an HMAC-SHA-1 self-test, then SHA-1 will need to be self-tested either directly or by means of self-testing another approved high-level algorithm that employs SHA-1. We assume that SHA-1 has been self-tested and explain why it is not necessary to further perform an HMAC-SHA-1 self-test.

The successful KAT for HMAC-SHA-256 checks the correctness of the following value: $H(K_1\|H(K_2\|m))$, where K_1 is the HMAC key K (padded with the zero bits on the right to get to the SHA-256 block size (512 bits)) XORed with the 512-bit value (written here in the hexadecimal format) Hex(5c5c…5c), K_2 is the similarly padded value of K XORed with the 512-bit value Hex(3636…36) and H is SHA-256. For HMAC-SHA-1, a KAT would have tested $H(K_1\|H(K_2\|m))$ with SHA-1 as H. As we are assuming that SHA-1 has been self-tested separately, the only error that may possibly occur when computing HMAC-SHA-1 is in the padding being different due to the different output lengths of SHA-256 and SHA-1.

The following figure shows the formats of both versions of HMAC.

SHA-256{|_____K_1_____|_SHA-256(K_2||m)_|100…..000|00…..00100000000|}
          512 bits   256 bits   191 zeros   8 zeros at the end
                                                                       64 bits total
SHA-1{|_____K_1_____|__SHA-1(K_2||m)__|100………..000|00…..00010100000|}
          512 bits   160 bits   287 zeros   5 zeros at the end
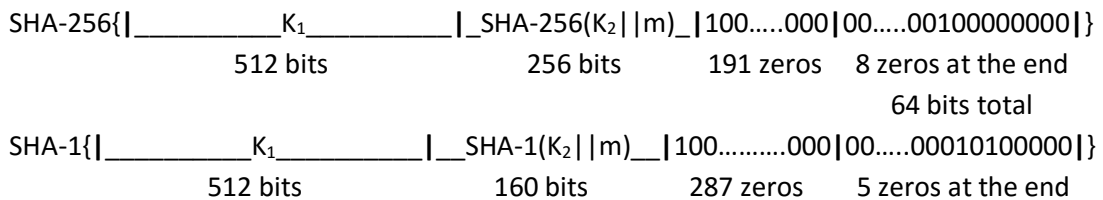
The first expression is self-tested successfully. In the second expression, K_1 and K_2 are the same as in the first one, while SHA-1 is self-tested separately and hence can be trusted to operate as designed. The paddings in the two expressions are slightly different. However, the module has been tested by a CST lab and has undergone an integrity test upon a power-on. It is safe to assume that as the module passes the applicable self-tests and correctly perform the HMAC-SHA-256 padding, it is also very likely to continue to properly perform the HMAC-SHA-1 padding. The last block in each of the above bit strings consists of 64 bits and is used to represent an output length of the hash function in the HMAC. For example, the output length of SHA-1 is $160 = 128 + 32$, thus requiring two binary 1's separated by a 0.

3. The difference in the padding mechanism would be more significant when comparing the HMAC-SHA-256 and HMAC-SHA-512, due to the different length of the Blocks (512 and 1024 bits, correspondingly). However, again, when one HMAC is self-tested, performing a KAT for the other one only results in testing the correctness of the padding mechanism.

4. The KMAC self-test requirements are addressed in IG A.15.

5. It is not required to perform a KAT for the Triple-DES MAC, as long as the underlying Triple-DES algorithm is self-tested.

9.2 Known Answer Test for Embedded Cryptographic Algorithms

Applicable Levels:	*All*
Original Publishing Date:	*02/10/2004*
Effective Date:	*02/10/2004*
Last Modified Date:	*03/27/2018*
Relevant Assertions:	*AS.09.19*
Relevant Test Requirements:	*TE09.19.01-03*
Relevant Vendor Requirements:	*VE.09.19.01-02*

Background

Core cryptographic algorithms are often embedded into other higher cryptographic algorithms for their operation in FIPS mode (e.g. SHA-1 algorithm embedded into HMAC-SHA-1 and DSA). IG 9.11 lists the conditions when a cryptographic module that implement FIPS-approved algorithms performs the Known-Answer-Tests (KATs) as part of their power-up self-tests. However, when the cryptographic module performs a KAT on the higher cryptographic algorithm, the embedded core cryptographic algorithm may also be self-tested.

Question/Problem

If an embedded core cryptographic algorithm is self-tested during the higher cryptographic algorithm KAT, is it necessary for the cryptographic module to implement a KAT for the already self-tested core cryptographic algorithm implementation?

Resolution

It is acceptable for the cryptographic module not to perform a KAT on the embedded core cryptographic algorithm implementation if;

1. the higher cryptographic algorithm uses that implementation,

2. the higher cryptographic algorithm performs a KAT at power-up and,

3. all cryptographic functions within the core cryptographic algorithm are tested (e.g. encryption and decryption for AES).

Additional Comments

If the cryptographic module contains several core cryptographic algorithm implementations (e.g., several different implementations of SHA-1 algorithm) and some are not used by other higher FIPS-approved cryptographic algorithms (and are therefore not self-tested), then the cryptographic module must perform a KAT at power-up for each of those implementations.

9.3 KAT for Algorithms used in an Integrity Test Technique

Applicable Levels:	*All*
Original Publishing Date:	*02/10/2004*
Effective Date:	*02/10/2004*
Last Modified Date:	*02/10/2004*
Relevant Assertions:	*AS.06.08 and AS.09.16*

Relevant Test Requirements:	*TE06.08.01-02 and TE09.16.01-02*
Relevant Vendor Requirements:	*VE.06.08.01 and VE.09.16.01*

Background

AS.06.08 requires that a cryptographic mechanism using an approved integrity technique **shall** be applied to all cryptographic software and firmware components within the cryptographic module. **AS.09.16** requires that a cryptographic algorithm test using a Known-Answer-Test (KAT) **shall** be conducted for all cryptographic functions of each approved cryptographic algorithm implemented by the cryptographic module and used in FIPS mode of operation.

Question/Problem

Must a cryptographic module implement a separate KAT for the underlying cryptographic algorithm used in the approved integrity technique?

Resolution

A cryptographic module may not implement a separate KAT for the underlying cryptographic algorithm used for the approved integrity technique if all the cryptographic functions of the underlying cryptographic algorithm are tested (e.g. encryption and decryption for Triple-DES).

Rationale

The software/firmware integrity check using an approved integrity technique is considered a KAT since the cryptographic module uses itself as an input to the algorithm and a known answer as the expected output.

EX: If HMAC-SHA-1 is used as the approved integrity technique to verify the software or firmware components, a KAT is not required for either the HMAC-SHA-1 or the underlying SHA-1 algorithm.

EX: If Triple-DES MAC is used as the approved integrity technique to verify the software or firmware components, a KAT is still required for the underlying Triple-DES as the integrity checking may not use both the Triple-DES encrypt and decrypt functions.

EX: If RSA is used to verify the signature of the software or firmware components, a KAT is still required for the underlying RSA as the integrity checking would not use the RSA signature generation function. However, a KAT for the underlying SHA-1 hashing function is not required.

9.4 Known Answer Tests for Cryptographic Algorithms

Applicable Levels:	*All*
Original Publishing Date:	*08/19/2004*
Effective Date:	*08/19/2004*
Note4: Transition End Date	***12/31/2012***
Last Modified Date:	*06/20/2012*
Relevant Assertions:	*AS.09.08-09, 12-13, 16 and 18*
Relevant Test Requirements:	
Relevant Vendor Requirements:	

Background

The cryptographic module **shall** perform the following power-up tests: cryptographic algorithm test, software/firmware integrity test, and critical functions test.

Cryptographic algorithm test. A cryptographic algorithm test using a known answer **shall** be conducted for all cryptographic functions (e.g., encryption, decryption, authentication, and random number generation) of each approved cryptographic algorithm implemented by a cryptographic module. A known-answer test involves operating the cryptographic algorithm on data for which the correct output is already known and comparing the calculated output with the previously generated output (the known answer). If the calculated output does not equal the known answer, the known-answer test **shall** fail.

Cryptographic algorithms whose outputs vary for a given set of inputs (e.g., the Digital Signature Algorithm) **shall** be tested using a known-answer test or **shall** be tested using a pair-wise consistency test (specified below).

Each approved cryptographic function implementation to be used in a FIPS approved mode of operation **shall** implement a cryptographic algorithm test. The cryptographic algorithm test is a *health check* of the algorithm implementation performed at power-up or on demand.

Question/Problem

What Known Answer Tests (KATs) are required for the symmetric-key algorithms that perform an invertible (encryption / decryption) operation? What are the minimum requirements placed on KATs for SHS algorithms and higher cryptographic algorithms implementing SHS algorithms so that they can be used in FIPS approved mode of operation? What qualifies as a KAT for an asymmetric-key algorithm whose output does not vary for a given set of inputs? Which approved algorithms allow the use of a pair-wise consistency test in lieu of a KAT? What are the minimum requirements placed on a pair-wise consistency test (for public and private keys) if performed at power-up or on demand?

Resolution

Following is a subset of algorithm KAT specific implementation guidance:

- for symmetric-key algorithms, such as SKIPJACK, Triple-DES or AES,

  – if the module implements the encryption function, the module **shall** have an encrypted value pre-computed, perform the encryption using known data and key, and then compare the result to the pre-computed value;

  – if the module implements the decryption function, the module **shall** have a decrypted value pre-computed, perform the decryption using known data and key (the data could be the encrypted value computed during the encryption test), and then compare the result to a value that was pre-computed value.

Note1: The SKIPJACK algorithm can only be used for decryption in FIPS approved mode so only a known-answer for decryption is required.

- if the module implements a SHS function, the following **shall** be the minimal requirements for SHS algorithms:

  - a KAT for SHA-1 is required;
  - a KAT for SHA-256 is required;
  - a KAT for SHA-224 is required if SHA-224 is implemented without SHA-256;
  - a KAT for SHA-512 is required; and,
  - a KAT for SHA-384 is required if SHA-384 is implemented without SHA-512.

- if the module implements a HMAC function, a KAT for HMAC is required and **shall** be performed with the HMAC function using at least one of the implemented underlying SHS algorithms.

- if the module implements an approved RNG or a DRBG algorithm then a KAT is required and **shall** be performed for each implemented algorithm. The values, such as seed and seed key that normally contribute to the "randomness" of an RNG or DRBG, **shall** be preset and used in the calculation of an output of the RNG or DRBG, which **shall** then be compared to the pre-computed result.

- for each public key digital signature algorithm (RSA, DSA and ECDSA), a KAT **shall** be performed using at least one of the schemes approved for use in the FIPS mode. For example, if an RSA signature algorithm is self-tested using an X9.31-compliant scheme, it is not necessary to perform any additional known-answer tests for the implementations of the digital signature compliant with RSASSA-PSS or RSASSA-PKCS1-v1_5, even if these schemes are also supported by the module.

- for the RSA algorithm,

  - if the module implements digital signature generation, the module **shall** have an RSA digital signature pre-computed, generate an RSA digital signature using known data and key, and then compare the result to the pre-computed value;

  - if the module implements digital signature verification, the module **shall** have an RSA digital signature pre-computed (which could be the output of the RSA digital signature generate test), and using a known key, verify the signature by comparing the recovered message with its target value.

The only exception to the above requirement is when the module implements the RSA Probabilistic Signature Scheme (PSS) *only*. In this case, per the provision of Section 4.9.1 of FIPS 140-2, the RSA digital signature algorithm may be tested using a pair-wise consistency test, since the algorithm's output may vary for a given set of inputs. If the module implements at least one approved RSA digital signature algorithm that has a fixed output value for a given input, an RSA KAT using the pre-computed values **shall** be performed.

Note2: an RSA KAT **shall** be performed using both the public and private exponents (*e* and *d*) and the two exponents **shall** correspond [that is, $d * e = 1 \pmod{\text{LCM}(p\text{-}1, q\text{-}1)}$]. The public exponent *e* used in this RSA KAT **shall** be chosen from the public exponent values supported by the module.

Note3: an RSA KAT **shall** be performed at a minimum on any one approved modulus size that is supported by the module.

Note4: The CMVP will not validate RSA digital signature algorithms as approved in modules that implement a pair-wise consistency test in lieu of a KAT at power-up (other than the above exception for PSS only) as represented in new test reports received after the **Transition End Date** of **December 31, 2012**.

- for algorithms whose output vary for a given set of inputs such as DSA and ECDSA, they **shall** be tested either,

   - as a KAT similar to RSA for signature generation or verification if the randomization parameter is fixed, or

   - as a *pair-wise consistency test*. This test does not require the comparison of the intermediate result (the generated signature) to a known value.

   Note5: a KAT or pair-wise consistency test for DSA **shall** be performed at a minimum on any one approved modulus size that is supported by the module.

   Note6: a KAT or pair-wise consistency for ECDSA **shall** be performed at a minimum, on any one of the implemented curves in each of the implemented two types of fields (i.e., prime field where *GF(p)*, and binary field where *GF(2^m)*).

Additional Comments

This IG is consistent with IG 9.1 *Known Answer Test for Keyed Hashing Algorithm*.

IG 9.2 *Known Answer Test for Embedded Crypto Algorithms* applies.

Self-tests for **SP 800-56A** schemes are addressed in the IG 9.6.

If the module implements asymmetric key generation, the conditional (FIPS 140-2 Section 4.9.2) *pair-wise consistency test* is applicable. No power-up test is required to test the key generation function even if such function is implemented by the DSA, ECDSA, or the RSA Digital Signature algorithm(s) supported by the module.

   Rationale: The purpose of a KAT is to perform a health-check of the cryptographic module to identify catastrophic failures or alterations of the module between power cycles and not that the implementation is correct. The implementation verification is performed during the cryptographic algorithmic testing and validation.

9.5 Module Initialization during Power-Up

Applicable Levels:	*ALL*
Original Publishing Date:	*04/01/2009*
Effective Date:	*04/01/2009*
Last Modified Date:	*04/01/2009*
Relevant Assertions:	*AS.09.08, AS.09.09, AS.09.10, AS.09.11*
Relevant Test Requirements:	
Relevant Vendor Requirements:	

Background

Power-up tests **shall** be performed by a cryptographic module when the module is powered up. All data output via the data output interface **shall** be inhibited when the power-up tests are performed.

Question/Problem

What is the *initialization period* and what module activities are allowed to occur during that period?

Resolution

The *initialization period* is the period between the time power is applied to the module (after being powered off, reset, rebooted, instantiated, etc.), and the time the module completes the power-up tests and outputs status (success or failure) indicating that the module is ready or not to perform operational cryptographic functions and services. The module may perform many activities during this period (i.e. before, during or after the power-up tests are performed) prior to the output of status and the module becoming operational. The cryptographic module is not considered to be in a FIPS approved mode of operation during the *initialization period*.

During the initialization period, the module:

- **shall** perform all the power-up tests required by FIPS 140-2 Section 4.9 including **AS.06.08** in FIPS 140-2 Section 4.6.1 (if applicable). When completed, the results (i.e. indications of success or failure) **shall** be output via the "status output" interface; (status output may be implicit or explicit);

- **shall** perform all the necessary internal services required to properly initialize or instantiate the module in conjunction with performing the power up self-tests;

- may receive data and control input via the *data input interface* or *control input interface* (e.g. may receive data and control requests for approved services that the module may act upon once the initialization period is completed);

- **shall** inhibit all data output via the data output interface *except*:

  - the module is allowed to output, when requested, non-security relevant module identification information, or module identification information. The module **shall** prevent the output of any plaintext secret and private cryptographic keys or CSPs that are contained within the module.

If applicable, the security policy **shall** describe the outputted information and the services performed during the *initialization period*.

Once the initialization period is completed (which includes the power-up tests), the module would transition to the operational state and may start providing approved cryptographic functions and services (if operating in an approved mode of operation).

Additional Comments

Rationale: One can consider the services performed to properly initialize or instantiate the module and the exchange of non-security relevant information in conjunction with the power-up tests to be part of the power-up initialization sequence (e.g. a modules handshake during the powering sequence).

9.6 Self-Tests When Implementing the SP 800-56A Schemes

Applicable Levels:	*All*
Original Publishing Date:	*10/21/2009*
Effective Date:	*10/21/2009*
Last Modified Date:	*04/23/2012*
Relevant Assertions:	*AS.09.01*
Relevant Test Requirements:	*AS.09.27*
Relevant Vendor Requirements:	*AS.09.27*

Background

FIPS 140-2 Section 4.9 states that "… *A cryptographic module* **shall** *perform power-up self-tests and conditional self-tests to ensure that the module is functioning properly. Power-up self-tests* **shall** *be performed*

*when the cryptographic module is powered up. Conditional self-tests **shall** be performed when an applicable security function or operation is invoked (i.e., security functions for which self-tests are required).*"

SP 800-56A is different from other cryptographic algorithm standards in regard to the cryptographic algorithm test because the standard does not describe an algorithm but instead describes a scheme consisting of steps utilizing existing algorithms (i.e., DSA, ECDSA, SHA, RNG, etc.). Therefore, defining the self-test requirement is different. The self-test requirement does not directly address the correct implementation of the scheme as this is addressed by CAVP validation testing. The self-test defined in **SP 800-56A** instead addresses the major underlying mathematical functions.

Question/Problem

What power-up or conditional self-tests are required when a cryptographic module implements an approved **SP 800-56A**-compliant scheme?

Resolution

The following **SP 800-56A** power-up self-tests **shall** be performed:

1. ***Primitive "Z" Computation KAT***. A Known Answer Test (KAT) **shall** be performed on the underlying mathematical function(s) which use modular exponentiation for an FFC-based key establishment protocol (per **SP 800-56A**, Section 5.7.1.1) or point multiplication for ECC-based protocol (per **SP 800-56A**, Section 5.7.1.2).

   The mathematical function can be either the computation of $g^x \pmod{p}$ for an FFC scheme or the point multiplication hxP on an elliptic curve in the usual notation. The value of p used in a self-test **shall** be in the range supported by the module. The elliptic curve point multiplication **shall** be performed on one of the NIST-recommended curves supported by the module.

   The value of x in the self-test **shall** be chosen to make the computations non-trivial. In the FFC case, x **shall** be chosen such that $g^x > p$. In the case of the elliptic-curve-based computations, the value of x **shall** be greater than 1.

   The self-tests **shall** consist of performing the calculations and comparing their result to a pre-computed value. In the case of an elliptic curve self-test, it is sufficient to compare the x-coordinate of the computed point to its expected value.

   The actual computation of a Z value is not required.

2. ***Key Derivation Function (KDF) KAT***. A KAT **shall** be performed on the SHS function which is used for the KDF function(s) used (per **SP 800-56A**, Sections 5.8.1 and/or 5.8.2).

3. ***KATs on Prerequisite Algorithms***. KATs **shall** be performed on all underlying prerequisite algorithms used in a given **SP 800-56A** scheme. Depending on which **SP 800-56A** scheme, this may include DSA, ECDSA, SHS, and/or RNG/DRBG. If these KATs are already performed as required by their underlying prerequisite algorithms, they should not need to be repeated for **SP 800-56A** if the same implementation is used.

The following **SP 800-56A** conditional self-tests **shall** be performed:

1. ***Conditional Tests for Assurances***. Necessary conditional tests **shall** be performed on Assurances used in a given **SP 800-56A** scheme. Assurances are specified in **SP 800-56A** Sections 5.5.2, 5.6.2 and 5.6.3 and will vary based on the implementation.

2. ***Conditional Tests on Prerequisite Algorithms***. A pair-wise conditional test **shall** be performed for every key pair generated by the module for use in an **SP 800-56A**-compliant protocol. If a key pair already passed a pair-wise consistency test because the pair could be used in another algorithm

implemented by the cryptographic module, the key pair does not have to be retested for the purposes of being used in the **SP 800-56A** protocols.

If the module's validation certificate claims, by referencing a CVL algorithm certificate on the approved algorithms line, a partial compliance with the requirements of **SP 800-56A** by only implementing either one or more of the **SP 800-56A** primitive(s) or by computing a shared secret Z, then only a power-up test that is named above '*Primitive "Z" Computation KAT*' is required. No conditional self-tests are necessary

Additional Comments

Separate guidance may be provided in the future for implementations that do not claim conformance to **SP 800-56A**.

Test Requirements

The vendor and tester evidence **shall** be provided under **AS.09.27**.

9.7 Software/Firmware Load Test

Applicable Levels:	*All*
Original Publishing Date:	*12/23/2010*
Effective Date:	
Last Modified Date:	*12/23/2010*
Relevant Assertions:	*AS.09.34 and AS.09.35*
Relevant Test Requirements:	
Relevant Vendor Requirements:	

Background – FIPS 140-2

FIPS 140-2 DTR

AS.09.34: (Levels 1, 2, 3, and 4) If software or firmware components can be externally loaded into the cryptographic module, then the following software/firmware load tests shall be performed.

AS.09.35: (Levels 1, 2, 3, and 4) An Approved authentication technique (e.g., an Approved message authentication code, digital signature algorithm, or HMAC) shall be applied to all validated software and firmware components when the components are externally loaded into the cryptographic module.

Question/Problem

How is this conditional test applicable for a hardware, software or firmware module?

Resolution

- For a hardware module, this requirement is applicable if software or firmware can be loaded within the defined physical boundary of the module.

The logical boundary of a software or firmware module includes all software and/or firmware that is associated, bound, modifies or is an executable requisite of the validated software or firmware module.

- For a software module, this requirement is applicable if software can be loaded within the defined logical boundary of the module.

- For a firmware module, this requirement is applicable if
  - firmware can be loaded within the defined logical boundary of the module (FIPS 140-2 Section 4.5 Level 1 only) or,
  - firmware can be loaded within the defined physical boundary of the module (FIPS 140-2 Section 4.5 Levels 2, 3, or 4.)

For a software module or a firmware module where FIPS 140-2 Section 4.5 physical security is Level 1, if the loaded software or firmware image is a complete replacement or overlay of the validated module image, this requirement is not applicable (NA) as the replacement or overlay constitutes a new module. The new module requires validation for conformance to FIPS 140-2 and is addressed as a **3SUB** (IG G.8 (3)) or **5SUB** (IG G.8 (5)) validation.

Note: The operator should zeroize the validated modules CSPs prior to the complete replacement or overlay of the validated module image.

The loading of non-security relevant software or firmware is addressed as a **1SUB** (IG G.8 (1)) validation submission and the loading of security relevant software or firmware is addressed as a **2SUB**, **3SUB** or **5SUB** (IG G.8) for validation. At a minimum, FIPS 140-2 Sections 4.10.1, 4.10.2 and Appendix C **shall** be addressed.

Additional Comments

Procedural or policy methods or statements can-not substitute for the FIPS 140-2 requirement for a software/firmware load test if the module has the capability to load software or firmware whether in an approved or non-approved mode of operation.

This requirement is not applicable if a module

1. has the capability to only load software or firmware during the pre-operational initialization (IG 9.5) of the module,

2. is configured to operate only in a FIPS approved mode of operation when operational,

3. the loading of software or firmware is inhibited while operational. (i.e. non-functional without transitioning through a new power-off, power-on re-initialization cycle), and

4. all CSPs are zeroized prior to loading the software or firmware during the pre-operational initialization.

9.8 Continuous Random Number Generator Tests

Applicable Levels:	*All*
Original Publishing Date:	*05/02/2012*
Effective Date:	*12/22/2015*
Last Modified Date:	*12/22/2015*
Relevant Assertions:	*AS09.41*
Relevant Test Requirements:	*TE09.41.01*
Relevant Vendor Requirements:	*VE09.41.01*

Background

AS.07.04: (Levels 1, 2, 3, and 4) If a cryptographic module employs Approved or non-Approved RNGs in an Approved mode of operation, the data output from the RNG shall pass the continuous random number generator test as specified in Section 4.9.2.

AS.09.29: (Levels 1, 2, 3, and 4) Conditional tests shall be performed by the cryptographic module when the conditions specified for the following tests occur: pair-wise consistency test, software/firmware load test, manual key entry test, continuous random number generator test, and bypass test.

AS.09.41: (Levels 1, 2, 3, and 4) If a cryptographic module employs Approved or non-Approved RNGs in an Approved mode of operation, the module shall perform the following continuous random number generator test on each RNG that tests for failure to a constant value.

Question/Problem

The FIPS 140-2 standard includes a requirement that all random number generators **shall** pass the continuous random number generator test (CRNGT). This requirement needs to be further explained in view of the existence of new technology such as the **SP 800-90A** Deterministic Random Bit Generators (DRBGs). In particular, the following questions need to be answered.

Does an **SP 800-90A**-compliant DRBG have to satisfy the Continuous Random Number Generator Test (CRNGT) requirement specified in **AS.07.04** and **AS.09.41**?

How should the CRNGT requirement be interpreted for all other RNGs?

Resolution

While **AS.07.04** and **AS.09.41** state that RNGs employed in an approved mode of operation **shall** perform a CRNGT as specified in Section 4.9.2 of FIPS 140-2, there are alternative ways in which this requirement to "perform a CRNGT" can be interpreted.

The simplest way to meet this requirement is to implement, for each RNG used in an approved mode, an explicit test as described in Section 4.9.2 of FIPS 140-2, AS.09.42 and AS.09.43. However, the same or the generally equivalent assurance of the RNG "not being stuck" can be achieved when performing different tests and taking into account other evidence. There are several considerations that lead to an alternative interpretation.

1. Before any random numbers are generated the module must successfully pass an integrity test. While this requirement was always present and had to be supplemented by the CRNGT, the advancement in the reliability of modern technology makes it extremely unlikely that an RNG/DRBG, tested by an accredited lab, will fail (get stuck on one output value); especially, after an integrity test was passed at module's power-up.

2. The DRBGs are particularly safe as they must pass certain health tests independent of the FIPS 140-2 CRNGT requirement. While it is possible to construct a scenario when a DRBG passes its health tests but fails a CRNGT, the **SP 800-90A** health test requirements provide an additional, strong assurance that the DRBG does not have a catastrophic failure. Conversely, just because a DRBG passes the CRNGT it is not a guarantee this generator is working properly – see item 3 below. At the time of the publication of FIPS 140-2, there was no **SP 800-90A** standard and no DRBG mechanisms available, so the assurances these advancements offer could not have benefited the introduction of a different CRNGT in the standard. The state of technology is different now.

3. It can be argued that if the CRNGT on a tested DRBG is performed according to the *method* described in Section 4.9.2 of FIPS 140-2, this test would do more harm than good. Indeed, a true random number generator will, with certain probability, generate two identical outputs one after another. FIPS 140-2 treats this as an error condition. When correcting this condition, the module

introduces an unnecessary bias among the DBRG outputs. At the same time, the CRNGT does not guard against other possible symptoms of a random number/bit generator getting stuck: for example, the DRBG can be generating the string ABABABABAB…, and this condition would not be identified by the CRNGT.

4. Similar concerns do apply if the CRNGT from Section 4.9.2 of FIPS 140-2 is performed on the output from non-deterministic RNG (NDRNG). The statistical properties of the NDRNG output depend on the amount of entropy in the noise sources incorporated by the NDRNG. Therefore, a test that takes into account this dependency can adapt better to the characteristics of a particular NDRNG and avoid altering the bias in the output than the inflexible CRNGT. The Repetition Count Test (RCT) has the capability to take into account the statistical properties of the NDRNG while protecting against catastrophic failures that cause the NDRNG to become "stuck" on single output value for a significant period. The RCT is defined as follows:

Given assessed min-entropy H of a noise source, the probability that the source generating n identical samples consecutively is at most $2^{-H(n-1)}$. (See a proof in the Additional Comments area below.) The test raises a trigger, if a sample is repeated more than the cutoff value C, which is determined by the acceptable false-positive probability $\alpha > 0$ (that is, the probability that the entropy source is functioning normally, but at a certain time would produce a string of C consecutive identical samples) and the min-entropy estimate H. The cutoff value of the repetition count test is calculated as:

$$C = \left\lceil 1 + \frac{-\log_2 \alpha}{H} \right\rceil \qquad (1).$$

This value of C is the smallest integer satisfying the inequality $\alpha \geq 2^{-H(C-1)}$, which ensures that the probability of obtaining a sequence of identical values from C consecutive noise source samples is no greater than α. For example, for $\alpha = 2^{-30}$, an entropy source with $H = 7.3$ bits per sample would have a repetition count test cutoff value of $\lceil 1+30/7.3 \rceil = 6$.

The *recommended* value of α is 2^{-30}.

Given a dataset of noise source observations, and the cutoff value C, the test is performed as follows:
1. Let A be the current sample value.
2. Initialize the counter B to 1.
3. If the next sample value is A, increment B by one.
   a. If B is equal to C, return error.
   Else:
   a. Let A be the next sample value.
   b. Initialize the counter B to 1.
   c. Repeat Step 3.

Running the repetition count test requires enough memory to store:

A: the most recently observed sample value,
B: the number of consecutive times that the sample A has been observed, and
C: the cutoff value at which the test fails.

This test's cutoff values can be applied to any min-entropy estimate, H, including very small and very large estimates.

In view of the considerations stated above, the requirement to pass the CRNGT is interpreted as follows.

The SP-800-90A-compliant DRBGs are not required to perform the test as described in AS.09.42 and AS.09.43.

The NDRNGs **shall** perform either the RCT as described above or the CRNGT as described in AS.09.42 and AS.09.43.

All other random number generators approved for use in an approved mode **shall** perform the CRNGT precisely as described in AS.09.42 and AS.09.43.

The cryptographic module's Security Policy **shall** say, for each random number/bit generator that can be used in an approved mode whether the CRNGT, as described in Section 4.9.2 of FIPS 140-2, is performed.

Additional Comments

1. If the design of the cryptographic module is such that the approved RNG or DRBG is only seeded (seed and seed key) once from an NDRNG after cryptographic module power-on and never re-seeded (seed and seed key) until the module is powered-off, and the NDRNG is not used for any other function or purposes, then the module does not need to implement the CRNGT on the output of the NDRNG.

2. The RCT test may only be used if a min-entropy estimate exists for the output from the NDRNG.

3. One may think of the RCT as a generalization of the CRNGT, which can be viewed as a RCT with a cutoff value $C=2$.

    From formula (1) it is clear that $C=2$ if and only if $\dfrac{-\log_2 \alpha}{H} \leq 1$, or $\alpha \geq 2^{-H} = p_1$, where p_1 is the highest probability (given the entropy estimate H) of an occurrence of a given sample value amount the outputs of the NDRNG. If a vendor decides to require a smaller false positive probability α than the test will not report an error until there are more than 2 consecutive sample repetitions (6 in the example in this IG.)

    To further demonstrate that the RCT test is compliant with the requirements of Section 4.9.2 of FIPS 140-2, one can note that the requirements in AS.09.42 and AS.09.43 do not address the maximum sample size generated by a NDRNG. A vendor who chooses to implement the RCT test may, after selecting the parameter α and estimating the rate of min-entropy per bit, demonstrate the module's compliance with FIPS 140-2 by combining the outputs from that generator into longer samples that become the effective sample size of the outputs from the new "NDRNG". The resulting new "NDRNG" would have the larger samples, say 64 bits or larger, and the vendor may claim that this "NDRNG" produces more than 30 bits or entropy per sample. Then, taking into account the recommended value of $\alpha = 2^{-30}$, formula (1) yields $C=2$, which is precisely equivalent to performing a CRNGT. A vendor who would want to pursue this approach for complying with Section 4.9.2 of FIPS 140-2 would just need to determine the parameters α and H and define the new NDRNG with the corresponding sample size.

4. To establish formula (1) and show how it relates to the characteristics of a particular NDRNG one may consider a noise source with assessed min-entropy H and prove that the probability of generating C identical samples consecutively by such a source is at most $2^{-H(C-1)}$. Let $p_1, p_2, \ldots, p_k$ be the probabilities of the possible outcomes from sampling the NDRNG, with $p_1 \geq p_2 \geq \ldots \geq p_k$. Note that the probability of producing C consecutive samples of outcome i is p_i^C. Then the probability P of producing any C consecutive identical samples is

$$P = p_1^C + p_2^C + \ldots + p_k^C = p_1 p_1^{C-1} + p_2 p_2^{C-1} + \ldots + p_k p_k^{C-1}$$

$$\leq p_1 p_1^{C-1} + p_2 p_1^{C-1} + \ldots p_k p_1^{C-1} = \left(p_1 + p_2 + \ldots p_k \right) p_1^{C-1} = p_1^{C-1} =$$

$$\left(2^{-H} \right)^{C-1} = 2^{-H(C-1)}, \text{ by the definition of min-entropy.}$$

5.   It is very important to use the same entropy estimate H in determining the cutoff value C for the RCT as the estimated entropy for determining the size of output buffer from the NDRNG to use in seeding a DRBG with sufficient strength for generating cryptographic keys. In other words, it is not acceptable to use a low entropy estimate H_1 in order to compute a large C for the RCT applied to the NDRNG but use a different larger estimate H_2 to determine how many samples of the NDRNG output would be needed to derive a seed for a DRBG.

9.9 Pair-Wise Consistency Self-Test When Generating a Key Pair

Applicable Levels:	*All*
Original Publishing Date:	*07/25/2013*
Effective Date:	
Last Modified Date:	*06/13/2017*
Relevant Assertions:	*AS09.30-33*
Relevant Test Requirements:	*TE09.31.01 and TE09.33.01*
Relevant Vendor Requirements:	*VE09.31.01 and VE09.33.01*

Background

AS09.30: (Levels 1, 2, 3 and 4) If a cryptographic module generates public or private keys, then the following pair-wise consistency tests for public and private keys shall be performed.

AS09.31: (Levels 1, 2, 3 and 4) If the keys are used to perform an approved key transport method, then the public key shall encrypt a plaintext value. The resulting ciphertext value shall be compared to the original plaintext value. If the two values are equal, then the test shall fail. If the two values differ, then the private key shall be used to decrypt the ciphertext and the resulting value shall be compared to the original plaintext value. If the two values are not equal, the test shall fail.

AS09.33: (Levels 1, 2, 3 and 4) If the keys are used to perform the calculation and verification of digital signatures, then the consistency of the keys shall be tested by the calculation and verification of a digital signature. If the digital signature cannot be verified, the test shall fail.

Question/Problem

1.   When does the pair-wise consistency self-test need to be performed upon the generation of a (private, public) key pair?

2.   Which of the two self-tests listed in the **AS09.31** and **AS09.33** needs to be implemented if, at the time of a key pair generation, it is not yet known if the newly-generated keys will be used in the digital signature or key establishment applications?

Resolution

There is no explicit requirement in FIPS 140-2 for the pair-wise consistency tests for keys used in the key-agreement schemes. AS.09.30 talks about performing the pair-wise consistency tests if the module generates public or private keys, but when this requirement is spelled out in detail, in AS.09.31 and AS.09.33, the

requirement applies only to the key transport schemes and to the signature algorithms, not to the key agreement schemes. The versions of **SP 800-56A** include various tests for the generated keys. Section 5.6.2.1.4 of **SP 800-56A Rev2** shows the applicable pair-wise consistency tests. No additional tests (that is, not listed in the **SP 800-56A** documents) are required by FIPS 140-2 for the key pairs used in the discrete-logarithm-based key agreement schemes.

The provisions of this Implementation Guidance apply only to the RSA-based key schemes, where the requirements of AS.09.31 and AS.09.33 are applicable. These include the RSA signature and the **SP 800-56B**-compliant key transport (encapsulation) schemes.

1. **Timing of the Pair-Wise Consistency Self-Test**

   The pair-wise consistency self-test **shall** be performed after the generation of a pair (private and public) of keys and before the intended use in the RSA-based asymmetric-key cryptography.

2. **Choice of the Pair-Wise Consistency Self-Test**

   If it is known at the time when the (private, public) RSA key pair is generated whether the keys will be used in a digital signature algorithm or to perform a key transport, then the choice of a pair-wise consistency test **shall** be consistent with the intended use of the keys. That is, if a key pair has been generated for use in the computation and the verification of a digital signature then the pair-wise consistency of the keys **shall** be tested by the calculation and verification of a digital signature on a message as described in **AS09.33**. If a key pair has been generated for use in an approved RSA-based key transport scheme, then the test **shall** be performed as described in **AS09.31**.

   If at the time when a key pair is generated, it is not known whether this pair of keys will be used in the RSA digital signature or the key establishment applications then a pair-wise consistency self-test described either in **AS09.31** or **AS09.33** may be performed on this key pair.

Additional Comments

This Implementation Guidance does not introduce any new requirements not already present in FIPS 140-2. It allows a more relaxed interpretation of the self-test requirement by saying that in one specific case there is a choice how a pair-wise consistency test can be performed.

9.10 Power-Up Tests for Software Module Libraries

Applicable Levels:	*All*
Original Publishing Date:	*07/25/2013*
Effective Date:	
Last Modified Date:	*04/25/2014*
Relevant Assertions:	*AS.09.08, AS.09.09, AS.09.13*
Relevant Test Requirements:	*TE.09.09.01, TE.09.09.02*
Relevant Vendor Requirements:	*VE.09.09.01, VE.09.13.01*

Background

FIPS 140-2 sets the following power-up test requirements for cryptographic modules:

AS09.08: (Levels 1, 2, 3 and 4) Power-up tests shall be performed by a cryptographic module when the module is powered up (after being powered off, reset, rebooted, etc.).

AS09.09: (Levels 1, 2, 3 and 4) The power-up tests shall be initiated automatically and shall not require operator intervention.

TE.09.09.02: The tester shall power-up the module and verify that the module performs the power-up self-tests without requiring any operator intervention.

Software modules may be implemented as applications or libraries. Applications and libraries are fundamentally different from each other with respect to the way the Operating System (OS) loader manages the operational control once the corresponding software module is loaded into memory. The OS loader automatically transfers control over to the application but does not do this in the case of a library unless the library is specifically instrumented to request a transfer. Therefore, an application naturally starts to execute its instructions automatically and without intervention after it is loaded but a library cannot unless it is specifically designed for this. This means that applications may easily satisfy the power-up test requirements for cryptographic modules but libraries need special care.

There are different types of software libraries with respect to the way they are intended to be linked and used by an application: static, shared and dynamically loaded (dynamic). This guidance is applicable to all library types.

Question/Problem

How can modules implemented as software libraries meet the power-up test requirements in **AS09.09**?

Resolution

FIPS 140-2 treats software applications used by an operator on a computing platform as acting on behalf of that operator. An application that links a software module library is considered a user of the module. As a result, any direct *run-time* action taken by the application on that module is considered to be an operator action.

A software module library **shall** be designed with a mechanism that forces the OS loader to transfer control over to the library immediately after loading it. The designed mechanism **shall** ensure that the transfer results into an automatic execution of a library function or a designated library code block without any intervention from an application and before the control is returned back to an application initiating the load. The power-on self-tests of the module **shall** be triggered from within that library function or code block. This execution paradigm satisfies **AS09.08** and **AS09.09** for a validated module.

Note1: While under control of the library function or code block, the determination of whether the module is a *validated* module may be performed. This may require the examination of data parameters indicating the module configuration. These parameters **shall** be set by the Crypto Officer during the module setup and initialization procedure. The Security Policy **shall** contain the detailed setup procedure with the specific instructions for establishing these parameter values. If the determination performed by the module while under the control of the library function or code block invoked by the OS loader is that the module is validated, then the power-on self-tests **shall** be initiated.

Note2: In modern operating systems, libraries may execute in user-space or in the OS kernel. A module implemented as a kernel library/extension **shall** utilize a mechanism that forces the kernel to transfer control over to the library/extension immediately after loading it. Most operating systems allow kernel extensions and provide specific mechanisms for implementing them, including the definition of a *default entry point* (DEP). If a DEP mechanism is provided, it is recommended that the kernel extension **shall** use it to initiate the power-on tests. This applies also to libraries used by other OS services (daemons) that are executing on behalf of the OS.

Note3: If a cryptographic software module is implemented as a static library with a DEP to satisfy the power-up self-test requirements, it **shall** also perform its runtime integrity check in memory by identifying and verifying the library's object code data and text segments in order to comply with **AS.09.13** without including the application into the module boundary. This also applies to shared or dynamic libraries that are loaded when it is impossible to verify the integrity of the library file image.

Note4: The module library **shall** be compiled appropriately so that the execution of constructor/destructor routines is *not* suppressed. If the operating system provides mechanisms to change the OS loader behavior and prevent the automatic invocation of the DEP to meet **AS09.09**, the tester **shall** verify that such mechanisms are

specifically disallowed in the crypto officer and operator guidance subject to **AS10.23** and **AS10.25**.

Note5: This guidance also applies to software-hybrid modules when the software part is implemented as a library.

Additional Comments

Dynamically loaded (dynamic) libraries are loaded at times other than during the startup of an application using them. Shared libraries are loaded by the application when it starts. In contrast, static libraries are embedded into the executable of the application at link time. Most compilers allow the definition of a DEP for software libraries, even for static ones. The presence of a library DEP forces the OS loader to call the DEP when it loads the library on behalf of the application linking it. The DEP is executed automatically and independently of the application code before the OS loader hands control back to the application. The OS loader utilizes a standard mechanism for invoking the DEP, which is agnostic of the library programming interface and completely independent of the application code. These nuances are important because while a cryptographic module is allowed to intercept calls to a service when the power-up tests are running and suppress any output of results until the tests complete, a module is not allowed to initiate the tests from within that service. In other words, a software module implemented as a library **shall not** rely on calls inside any function exported as a supported service to initiate the power-up tests. Only a function or a block of code, e.g. static blocks in Java, that is automatically invoked by the operating environment, e.g. the OS Loader, may initiate them.

Here are some examples for how a default entry point in a software module implemented as a library may be defined:

On Unix, Linux, Mac OS X:

```
void __attribute__((constructor)) runModulePOST() {
  /*… perform module self-tests…*/
}
```

On Windows for Win32 API:

```
BOOL WINAPI DllMain(
    HINSTANCE hinstDLL,  // handle to DLL module
    DWORD fdwReason,     // reason for calling function
    LPVOID lpReserved )  // reserved
{
    // Perform actions based on the reason for calling.
    switch( fdwReason )
    {
        case DLL_PROCESS_ATTACH:
         // Initialize once for each new process.
         // Here is where the module POST should be invoked
         // Return FALSE to fail DLL load in case POST fails
            break;

        case DLL_THREAD_ATTACH:
         // Do thread-specific initialization.
            break;

        case DLL_THREAD_DETACH:
         // Do thread-specific cleanup.
            break;

        case DLL_PROCESS_DETACH:
         // Perform any necessary cleanup.
            break;
    }
    return TRUE;  // Successful DLL_PROCESS_ATTACH.
}
```

Note6: Static constructors, i.e. DEP-equivalent mechanisms, exist also for C++ and .NET libraries and libraries implemented in other object oriented languages. To substitute for DllMain or to define a DEP in a .NET library, one could employ a static constructor on the exported class to invoke the initialization code. The default constructors of static C++ objects are executed automatically upon loading the library containing them. Similarly, Java provides static code blocks that are executed automatically when the Java Virtual Machine class loader loads the class.

Note7: The OS loaders of some operating systems do not provide a DEP mechanism for software libraries. In such cases the module **shall** utilize the available programming language capabilities to implement a DEP-like initialization as described in **Note6** above. If the module is written in a procedural language, such as the C programming language, to avoid a complete rewrite in an object-oriented language, a judicious switch to a different compiler should be considered. For example, switching to a C++ compiler and placing the original C code inside extern "C"{} brackets, would enable static objects with the desired properties for the module.

Test Requirements

The vendor and tester evidence **shall** be provided under **VE.09.09.01, TE.09.09.01, TE.09.09.02** and **TE.09.22.01-TE.09.22.07**.

9.11 Reducing the Number of Known Answer Tests

Applicable Levels:	*All*
Original Publishing Date:	*08/07/2017*
Effective Date:	*08/07/2017*
Last Modified Date:	*05/25/2018*
Relevant Assertions:	*AS09.07 and AS09.16*
Relevant Test Requirements:	*TE09.07.01 and TE09.16.01*
Relevant Vendor Requirements:	

Background

AS.09.16 requires that a cryptographic algorithm test using a Known-Answer-Test (KAT) **shall** be conducted for all cryptographic functions of each approved cryptographic algorithm implemented by the cryptographic module and used in FIPS mode of operation.

Question/Problem

How should the "all" in **AS.09.16** be interpreted? One possible interpretation is that each time the module is powered-up, a separate known answer test is performed for each approved algorithm supported by the module. However, given the serious burden this requirement imposes on the module in terms of the amount of time spent before any data can be output by the module, it will be desirable to reduce the number of these tests if some other indication of the health of the module could be presented at the time of the module's power-up.

A certain reduction in the number of KATs has already been implemented. **IG 1.7** allows the module to perform only those KATs that apply to algorithms in the Approved mode of operations chosen at power-up. Various IGs in Section 9 of the **FIPS 140-2** Implementation Guidance show that if the module successfully passes some other tests then performing a KAT for a particular algorithm may be unnecessary.

Can the number of KATs be further reduced given the knowledge that the module passed its software/hardware integrity test which is anyway required when the module contains any software or firmware?

Resolution

The module **shall** pass a complete set of power-on self-tests (an integrity test and the KAT for all approved algorithms in the chosen approved mode of operation per **IG 1.7**) upon the module's installation/configuration in each operational environment. Once the module has passed this initial set of power-on self-tests, in any

subsequent restarts if the module passed a software/firmware integrity test, then the module need not perform a KAT for any approved algorithm supported by this module; however, no approved algorithm can be executed by the module without first having performed a self-test prescribed for this algorithm. The software/firmware integrity test is mandatory for every restart of the module.

The module **shall** still provide an operator with the capability to request the execution of the KATs for all approved algorithms supported by the module at any stage of the module's lifecycle.

Rationale

If the module and its algorithm implementations have been tested by an accredited testing lab, this already provides strong guarantees that all algorithms produce the correct results. If, further, the module's firmware/software integrity has been tested, and all KAT succeed at the module's first start-up in the same operational environment, then it is all but impossible that a particular algorithm implementation running in the same operational environment has been corrupted. Thus, performing the KATs at each subsequent start-up, at the same time when the module's integrity test is performed, does not add to the assurance of the module's ability to correctly perform the cryptographic operations. While the **FIPS 140-2** standard does require that all algorithms are tested by a KAT at power-up, it can be argued that this requirement is met by the combination of lab testing, the testing upon the initial installation of the module on the platform of use which includes the power-up integrity test and the full set of KATs, and the software/firmware integrity test at each power-on of the module.

This IG does not apply to Hardware Modules with no firmware present. Any such module is required to perform the power-on self-tests for all approved algorithms defined for an approved mode selected for a given power-on.

Additional Comments

1. This Implementation Guidance does not change the requirements of the **FIPS 140-2** standard. The **FIPS 140-2** requirement **AS09.16** continues to apply. The current guidance gives the vendor a choice of how a known answer test may be conducted. The tests may be performed either by a direct computation of a cryptographic function and comparing the result of said computation to a pre-computed value, or by claiming one of the exceptions from this rule already specified in several other IGs, or – an option introduced in this Guidance – by observing and claiming that the successful passing of the known answer tests is already guaranteed by the module being tested initially and executing a software/firmware integrity test at each power-on.

2. If a pair-wise consistency test may be used in lieu of a known answer test, this guidance applies and the use of this test is not needed when the guidance says that an execution of the corresponding known answer test is not required.

3. The module is required to have the capability to perform the applicable power-up tests on demand for the approved algorithms implemented in the module.

4. To take advantage of the provisions of this IG, the module **shall** know whether a particular power-up is performed when the module is first installed or configured in a new operational environment. If the module needs to use a parameter that "remembers" that the module has already been installed in a given environment and the algorithms have been self-tested, then this parameter's value **shall** be treated the same as a public key, in which case the integrity of this parameter is assured by the module.

    In the absence of a parameter that keeps track of the earlier execution of the power-up self-tests, the module's operator may make the determination if the power-up algorithm self-tests need to be performed, based on the operator's knowledge of the history of performing these tests in each environment. If neither the module itself nor the operator is able to determine if this is the first time the module is instantiated in a given environment, the module **shall** perform all of the applicable algorithmic power-up self-tests for the approved algorithms implemented in the module.

9.12 Integrity Test Using Sampling

Applicable Levels:	*All*
Original Publishing Date:	*12/04/2017*
Effective Date:	*12/04/2017*
Last Modified Date:	*12/04/2017*
Relevant Assertions:	*AS.06.08 and AS.09.22*
Relevant Test Requirements:	*TE06.08.01-02 and TE09.22.01-07*
Relevant Vendor Requirements:	*VE.06.08.01 and VE.09.22.01*

Background

AS.09.22 requires that a software/firmware integrity test using an error detection code (EDC) or approved authentication technique (e.g., and approved message authentication code or digital signature algorithm) **shall** be applied to all validated software and firmware components within the cryptographic module when the module is powered up.

Question/Problem

Some cryptographic modules running in a limited Operational Environment include an operating system and hardware drivers together with applications, which results in a large installation binary image (e.g. a few Gigabytes installation image for a network appliance). Some other modules may not have a large (in terms of its bit size) firmware image, but they run on a resource-constrained platform (e.g. smart cards with a low processor speed). In both cases, an integrity check, even if it is using an EDC, takes such a long time to complete that it significantly slows the power-up process and renders the devices with such FIPS validated modules useless. This raises a problem of how the integrity test required in FIPS 140-2 should be performed in a reasonably short time for modules containing firmware.

How should the "all" in **AS.09.22** be interpreted? One possible interpretation is that each time the module is powered-up a firmware integrity test is performed on the *entire* image of the firmware. This technique is unacceptably slow for some devices, as indicated above. It will be desirable to reduce the amount of firmware image subject to the integrity check while still maintaining the evidence of the integrity of the module's

firmware. Sampling is the method commonly used in various industries to reduce the testing space while maintaining an acceptable level of assurance.

Will sampling methods be acceptable in a firmware integrity test?

Resolution

The module may choose to perform the firmware integrity test either on the whole firmware image or it may use a sampling method.

The following assumptions are made:

1. The firmware image can be viewed as a bitstring or a set of files.

2. The bitstring or files consisting of the firmware image can be divided into up to 20 portions, each of which has the total size no less than 100,000 bits.

3. The probability of any bit taking an erroneous value in the firmware image bitstring is very low.

The first time an integrity self-test is performed upon the installation or reconfiguration of the module or upon a factory reset, the integrity test **shall** be performed on the entire firmware image. Then for a firmware integrity test at each subsequent power-on, the integrity test is performed on a portion of the firmware that will be chosen by a sampling method. This speeds up the operations significantly. Furthermore, by using a sampling method to determine which portion of the firmware is tested on each application of an integrity test, all parts of firmware will get tested after the integrity tests are performed sufficiently many times. Therefore, it can be claimed that the test does apply to all cryptographic firmware components within the module, thus literally meeting the requirements of **AS.09.22**.

Suppose that the firmware has been divided into n portions labeled as P_1, P_2, …, P_n, where $1 \leq n \leq 20$ and bitsize(P_i) $\geq$ 100,000 for $1 \leq i \leq n$. Each time when an execution of an integrity test is required, the module samples, either **deterministically** or **randomly,** the integers between 1 and n to select the portion of the firmware which will be integrity-tested.

Deterministic Sampling Method

This method predefines the order in which a portion of the firmware for integrity test will be selected. Without the loss of generality, suppose that the predefined order is P_1, P_2, …, P_n. When the deterministic integrity test is performed for the first time, it is performed on portion P_1. The module saves the current index counter: 1. Each subsequent time the integrity test is performed, the index counter is incremented by 1 and the portion P_j, where j is the newly incremented index, is tested. The new index j is now saved and **shall** be treated as the module's public key: the module shall prevent any unauthorized modification of this parameter. When the last index n needs to be incremented, the counter is reset to 1 and the process continues. If this deterministic method is used, after n invocations of the integrity test the entire firmware image will have been tested.

Random Sampling Method

To use the random method, the module **shall** implement a random number generator Υ. This random number generator, which can generate integers between 1 and n, each of them at the same rate $1/n$, does not have to be approved for use in cryptographic applications; however, it **shall** generate its own entropy. The amount of the generated entropy **shall** be sufficient to make possible the choice of any integer number between 1 and n. The

module **shall** be able to generate this small amount of entropy at the time of each invocation of the integrity check, including at power-up.

When a firmware integrity test is performed, the random number generator is called to select an index j between 1 and n and then the module performs an integrity test on the portion P_j.

Additional Comments

1. A firmware integrity test is a health test only. It is not designed nor is it intended to guard against the targeted attacks. The cryptographic module will have other means of defense – commensurate with the module's Security Level – to protect against the deliberate attacks.

2. In the event of an integrity test failure, recovery from the error state **shall** require that the module perform an integrity check of the entire firmware as was done in the initial installation of the module. If the integrity check of the entire firmware is successful, the module may return to running an integrity check using sampling either via the deterministic or the random method described in this IG.

3. The module **shall** be designed to perform, at the request of an operator, an integrity test on its entire firmware image.

4. A natural partition of the module's firmware may contain more than 20 files or bitstrings. The vendor may group some of these portions together to meet the requirement of having no more than 20 P_i's for the purpose of performing the integrity test using sampling.

5. If the random sampling method is used, the testing lab **shall** provide to the CMVP the design of the random number generator ϒ along with the rationale for why this random number generator is suitable for its purpose and how its entropy is generated from the environment; especially, when ϒ is invoked at the power-up. The vendor's analysis of the collected entropy **shall** to be clear and logical; however, it does not need to meet the scrutiny of the analysis of the module's NDRNG that generates the entropy used for key generation.

6. This is an example of a random number generator that may be used to provide the entropy for the random sampling. It can be assumed that the module has a built-in clock and can measure time. At each power-up, the module will record the time and compute a number based on the last three decimal digits of the time's reading. That is, if the time is 13 hours, 51 minutes and 6.533146 seconds, the RNG will output the number m=146. If only the millisecond precision is available and the recorded time is 13 hours, 51 minutes and 6.533 seconds, the output is 533. The power-ups should not occur very frequently, so the outputs can be considered independent.

   Take the output from the previous step (e.g., 533) and mod it by the number of the portions of the firmware (e.g., 20) and increase it by 1: (533 mod 20) + 1 = 14. Then perform the integrity test on the portion P_{14}.

7. The upper bound (20) on n and the lower bound (100,000) on the bit size of each portion P_i in the subdivision of the module's firmware image have been chosen to assure that a significant part of the module's firmware is integrity-tested at each power-up of the module.

8. The tester **shall** document how the module meets each shall statement in this IG in the Assessment for TE.09.22.01.

9. The sampling method presented in this IG applies only to the cryptographic modules with a limited operational environment. The software modules, typically operating on a GPC with a modifiable operational environment are expected to operate with a sufficiently fast processor that will assure the execution of the full integrity test within an acceptable time limit.

9.13 Non-Reconfigurable Memory Integrity Test

Applicable Levels:	*All*
Original Publishing Date:	*12/04/2017*
Effective Date:	*12/04/2017*
Last Modified Date:	*12/04/2017*
Relevant Assertions:	*AS09.22*
Relevant Test Requirements:	*TE09.22.01*
Relevant Vendor Requirements:	*VE09.22.01, VE09.22.02*

Background

Software/firmware integrity test. A software/firmware integrity test using an error detection code (EDC) or Approved authentication technique (e.g., an Approved message authentication code or digital signature algorithm) **shall** (per AS.09.22) be applied to all validated software and firmware components within a cryptographic module when the module is powered up. The software/firmware integrity test is not required for any software and firmware components excluded from the security requirements of this standard (refer to Section 4.1 of FIPS 140-2).

Question/Problem

What is the definition of "non-reconfigurable memory"? Is software or firmware located in non-reconfigurable memory subject to the AS09.22 software/firmware integrity test?

Resolution

Non-reconfigurable memory **shall** be defined as a memory technology that stores data using a mechanical means (e.g. masked ROM, CD-ROM) that will never change once manufactured.

The software or firmware integrity test is not required for executable code stored in non-reconfigurable memory. This code is considered hardware. Therefore, the provisions of *IG 9.11 Reducing the Number of Known Answer Tests* do not apply in that if an integrity test is not performed on the non-reconfigurable memory area of the module, and if any of the approved cryptographic algorithms are implemented in this area, then the power-on self-tests **shall** be performed on these algorithms.

Additional Comments

The reason for the above definition on what constitutes non-reconfigurable memory is that most common read-only memory technologies (e.g. OTP, PROM, WORM, CD-R) store data using a chemical change or electrical charge that is subject to degradation over time. This IG does not apply to such cases where degradation is a factor, and the memory would still be subject to the integrity test. However, IG 9.12 Integrity Test Using Sampling provides strategies for reducing the burden of the integrity test.

Section 10 – Design Assurance

Section 11 – Mitigation of Other Attacks

11.1 Mitigation of Other Attacks

Applicable Levels:	*All*
Original Publishing Date:	*07/15/2011*
Effective Date:	
Last Modified Date:	*07/15/2011*
Relevant Assertions:	*AS.11.01*
Relevant Test Requirements:	*TE11.01.01-02*
Relevant Vendor Requirements:	*VE.11.01.01-02*

Background

AS.11.01: (Levels 1, 2, 3, and 4) If the cryptographic module is designed to mitigate one or more specific attacks, then the module's security policy shall specify the security mechanisms employed by the module to mitigate the attack(s).

Question/Problem

When is this section applicable?

Resolution

If a cryptographic module has been *purposely* designed, built and publicly documented to mitigate one or more specific attacks, this section is applicable and **AS.11.01 shall** be addressed regardless if the vendor of the module wishes to address the claim or not. Mitigation mechanisms may address both invasive (physical) or non-invasive mechanisms. The testing laboratory, upon inspection of the modules design and documentation (both proprietary and public), **shall** verify the implemented mitigation mechanisms and/or mitigations claimed by the vendor as specified in **AS.11.01**.

Example: FIPS 140-2 Section 4.5 Level 2 is claimed. However, the vendor states that module design includes a switch that will cause zeroization of CSPs if some part of the module is opened or penetrated. Since this is not required at Level 2, for the vendor to claim this feature, it **shall** be addressed in FIPS 140-2 Section 4.11 as an additional mitigation mechanism.

Section 12 – Appendix A: Summary of Documentation Requirements

Section 13 – Appendix B: Recommended Software Development Practices

Section 14 – Appendix C: Cryptographic Module Security Policy

14.1 Level of Detail When Reporting Cryptographic Services

Applicable Levels:	*All*
Original Publishing Date:	*11/15/2001*
Effective Date:	*11/15/2001*
Last Modified Date:	*08/07/2017*
Relevant Assertions:	*AS.01.02, AS.01.03, AS.01.12, AS.01.16, AS.03.14, AS.10.06, AS.14.02, AS.14.03, AS.14.04, AS.14.06, AS.14.07*
Relevant Test Requirements:	*TE01.03.01, TE01.03.02, TE01.16.01, TE03.14.01, TE10.06.01, TE14.07.01, TE14.07.02*
Relevant Vendor Requirements:	*VE.01.03.01, VE.01.03.02, VE.01.16.01, VE.03.14.01, VE.03.14.02, VE.10.06.01, VE.14.07.01, VE.14.07.02, VE.14.07.03*

Question/Problem

What is the level of detail that the non-proprietary security policy must contain in order to describe the cryptographic service(s) implemented by a cryptographic module?

Resolution

When presenting information in the non-proprietary security policy regarding the cryptographic services that are included in the module validation, the security policy **shall** include, at a minimum, the following information **for each service:**

- The service name

- A concise description of the service purpose and/or use (the service name alone may, in some instances, provide this information)

- A list of approved security functions (algorithm(s), key management technique(s) or authentication technique) used by, or implemented through, the invocation of the service.

- A list of the cryptographic keys and/or CSPs associated with the service or with the approved security function(s) it uses.

- For each operator role authorized to use the service:
  - Information describing the individual access rights to all keys and/or CSPs
  - Information describing the method used to authenticate each role.

The presentation style of the documentation is left to the vendor. FIPS 140-2, Appendix C, contains tabular templates that provide non-exhaustive samples and illustrations as to the kind of information to be included in meeting the documentation requirements of the Standard.

Additional Comments

FIPS 140-2 requires information to be included in the module security policy which:

- Allows a user (operator) to determine when an approved mode of operation is selected (**AS.01.16**).

- Lists all security services, operations or functions, both approved and non-approved, that are provided by the cryptographic module and available to operators (**AS.01.12, AS.03.07, AS.03.14, AS.14.03**). The only exception is services that belong to "group 4" defined in IG 3.5.

- Provides a correspondence between the module hardware, software, and firmware components (**AS.10.06**)

- Provides a specification of the security rules under which the module **shall** operate, including the security rules derived from the requirements of FIPS 140-2. (**AS.14.02**)

- For each service documented in the module Security Policy, specifies a detailed specification of the service inputs, corresponding service outputs, and the authorized roles in which the service can be performed. (**AS.03.14, AS.14.03**)

See also the definitions of *Approved mode of operation* and *Approved security function* in FIPS 140-2.

14.2 Level of Detail When Reporting Mitigation of Other Attacks

Applicable Levels:	*All*
Original Publishing Date:	*11/15/2001*
Effective Date:	*11/15/2001*
Last Modified Date:	*11/15/2001*
Relevant Assertions:	*AS.14.09*
Relevant Test Requirements:	*TE14.09.01*
Relevant Vendor Requirements:	*VE.14.09.01*

Question/Problem

What is the level of detail that the non-proprietary security policy must contain that describes the security mechanism(s) implemented by the cryptographic module to mitigate other attacks?

Resolution

The level of detail describing the security mechanism(s) implemented by the cryptographic module to mitigate other attacks required to be contained in the security policy must be similar to what is found on advertisement documentation (product glossies).

14.3 Logical Diagram for Software, Firmware and Hybrid Modules

Applicable Levels:	*All*
Original Publishing Date:	*07/03/2007*
Effective Date:	*07/03/2007*
Last Modified Date:	*07/03/2007*
Relevant Assertions:	*AS.14.01*
Relevant Test Requirements:	*TE14.01.01*
Relevant Vendor Requirements:	*VE.14.01.01*

Background

VE.14.01.01 specifies the requirement for the vendor to provide in the security policy a diagram or image of the physical cryptographic module.

While the requirement is vague when applied to a software, firmware or hybrid cryptographic module, it is intended as well to clearly illustrate the logical boundary of the module as well as the other logical objects and the operating environment with which the module executes with.

Question/Problem

For a software, firmware or hybrid cryptographic module, what are the requirements of the *logical diagram* contained in the security policy as specified in **VE.14.01.01**?

Resolution

The *logical diagram* must illustrate:

- the logical relationship of the software, firmware or hybrid module with respect to the operating environment. This **shall** include, as applicable, references to any operating system, hardware components (i.e. hybrid) other supporting applications, and illustrate the physical boundary of the platform. All the logical and physical layers between the logical object and the physical boundary **shall** be clearly defined.

Additional Comments

The *logical diagram* must convey basic information to the operator of the cryptographic module about its relationship respective to the operating environment.

The *logical diagram* could be a subset of the block diagram specified in **AS.01.13**.

14.4 Operator Applied Security Appliances

Applicable Levels:	*Level 2, 3 or 4*
Original Publishing Date:	*01/27/2010*
Effective Date:	
Last Modified Date:	*08/07/2017*
Relevant Assertions:	*AS.05.15, AS.05.26, AS.05.35, AS.05.49, AS.10.04, AS.10.22, AS.14.01 and AS.14.08*
Relevant Test Requirements:	
Relevant Vendor Requirements:	

Background

FIPS 140-2 Section 4.5, *Physical Security*, addresses specific requirements at Level 2. This IG addresses the following two requirements:

1. a module **shall** be constructed in a manner to provide tamper evidence, and

2. a module **shall** have an opaque tamper evident coating or enclosure.

IG 5.1 provides guidance on opacity and IG 5.2 on testing of tamper evident seals. Many module implementations are constructed in a manner where the operator of the module is required to install or affix items such as tamper evident seals or security appliances (e.g. baffles, screens, etc.) to configure the module to operate in the approved mode of operation. In addition, the operator may over the life-cycle of the module, modify some of the non-security relevant aspects of the module that would require the removal and replacement of tamper evident seals or security appliances.

Question/Problem

What specific information **shall** be included in the test report, certificate and Security Policy when a module validated at Physical Security Level 2 has tamper evident seals or security appliances that the operator will apply or modify over the lifecycle of the module?

Resolution

The following specific information **shall** be included in the test report, certificate and Security Policy to meet the relevant assertions:

1. If the module is shipped unassembled, then **AS.14.03 shall** be addressed with appropriate detail.

2. In addition to other applicable caveats, the certificate caveat **shall** include as applicable the following:

    (The <tamper evident seals> and <security devices> installed as indicated in the Security Policy)

3. The Security Policy **shall** include the following:

    a. The reference photo/illustration required in **AS.14.01 shall** reflect the validated module configured or constructed as specified on the validation certificate. Additional photos/illustrations may be provided to reflect other configurations that may include parts that are not included in the validation.

    b. If filler panels are needed to cover unpopulated slots or openings to meet the opacity requirements, they **shall** be included in the photo/illustration with tamper seals affixed as needed. The filler panels **shall** be included in the list of parts in **AS.01.08**.

    c. There **shall** be unambiguous photos/illustrations on the precise placement of any tamper evident seal or security appliance needed to meet the physical security requirements.

    d. The total number of tamper evident seals or security appliances that are needed **shall** be indicated (e.g. 5 tamper evident seals and 2 opacity screens). The photos/illustrations which provide instruction on the precise placement **shall** have each item numbered in the photo/illustration and will equal the total number indicated (the actual tamper evident seals or security appliances are not required to be numbered).

    e. If the tamper evident seals or security appliances are parts that can be reordered from the module vendor, the Security Policy **shall** indicate the vendor name and part number of the seal, security appliance or applicable security kit.

        Note: After reconfiguring, the operator of the module may be required to remove and introduce new tamper evident seals or security appliances.

    f. There **shall** be a statement in the Security Policy stating:

        The <tamper evident seals> and <security devices> **shall** be installed for the module to operate in the approved mode of operation.

    g. The Security Policy **shall** identify the operator role responsible for:

        ▪   securing and having control at all times of any unused seals, and

        ▪   the direct control and observation of any changes to the module such as reconfigurations where the tamper evident seals or security appliances are removed or installed to ensure the security of the module is maintained during such changes and the module is returned to a FIPS approved state.

    h. If tamper evident seals or security appliances can be removed or installed, clear instructions **shall** be included regarding how the surface or device **shall** be prepared to apply a new tamper evident seal or security appliance.

Additional Comments

If a cryptographic module requires more than one tamper evident seal to be applied, the Physical Security Test report that is submitted to the CMVP for review **shall** address the testing of each tamper evident seal individually if the surface topography or surface material is different between different sets of seals.

14.5 Critical Security Parameters for the SP 800-90 DRBGs

Applicable Levels:	*ALL*
Original Publishing Date:	*12/23/2010*
Effective Date:	

Last Modified Date:	*08/07/2017*
Relevant Assertions:	*AS.01.15, AS.07.09, AS.07.13, AS.07.14, AS.07.23, AS.14.04, AS.14.06*
Relevant Test Requirements:	*TE01.15.01, TE07.13.01*
Relevant Vendor Requirements:	

Background

The **FIPS 140-2** cryptographic module Security Policy **shall** specify all cryptographic keys and CSPs employed by the cryptographic module.

Question/Problem

Which are the critical security parameters that determine the security of the **SP 800-90A** DRBG mechanisms?

Resolution

The entropy input string and the seed **shall** be considered CSPs for all the DRBG mechanisms.

During the instantiation of a DRBG the initial state is derived from the seed. The internal state contains administrative information and the working state. Some values of the working state are considered secret values of the internal state. These values are listed below:

1. Hash_DRBG mechanism

    The values of V and C are the "secret" values of the internal state.

2. HMAC_DRBG mechanism

    The values of V and Key are the "secret values" of the internal state.

3. CTR_DRBG mechanism

    The values of V and Key are the "secret values" of the internal state.

Additional Requirements

1. The **SP 800-90A** requires that the internal state is protected at least as well as the intended use of the pseudorandom output bits requested by the consuming application. **SP 800-90A** further requires that the DRBG internal state is contained within the DRBG mechanism boundary and **shall not** be accessed by non-DRBG functions or other instantiations of that or other DRBGs.

    TE.01.15.01 **shall** specify how the above requirements are met.

2. In the case of the CTR_DRBG the test report **shall** indicate if a derivation function is used during the instantiation and reseeding. If the derivation function is not used, the test report **shall** explain how the module meets the **SP 800-90A** requirement of having another approved DRBG or an entropy source provide full entropy to the CTR_DRBG.

3. The following assessments: **AS07.09**, **AS07.14** and **AS07.23** are no longer applicable, as the only approved random number generators (not otherwise approved for classified applications) are the **SP 800-90A**-compliant DRBGs. The DRBGs do not use the *seed key* parameter.

FIPS 140-2 Annex A – *Approved Security Functions*

A.1 Validation Testing of SHS Algorithms and Higher Cryptographic Algorithm Using SHS Algorithms

Applicable Levels:	*All*
Original Publishing Date:	*08/19/2004*
Effective Date:	*08/19/2004*
Last Modified Date:	*08/19/2004*
Relevant Assertions:	*AS.01.12*
Relevant Test Requirements:	*TE01.12.01*
Relevant Vendor Requirements:	*VE.01.12.01*

Background

The Cryptographic Algorithm Validation Program (CAVP) validates every SHS algorithm implementation: SHA-1, SHA-224, SHA-256, SHA-384 and SHA-512. Several higher cryptographic algorithms use those SHS hashing algorithms in their operation.

Question/Problem

What are validation testing requirements for the SHS algorithms and higher cryptographic algorithms implementing SHS algorithms for their use in FIPS approved mode of operation?

Resolution

To be used in a FIPS approved mode of operation:

- every SHS algorithm implementation must be tested and validated on the appropriate OS.

- for DSA, RSA, ECDSA and HMAC, every implemented combination must be tested and validated on the appropriate OS.

The algorithmic validation certificate annotates all the tested implementations that may be used in a FIPS approved mode of operation.

Any algorithm implementation incorporated within a FIPS 140-2 cryptographic module that is not tested may not be used in a FIPS approved mode of operation. If there is an untested subset of a FIPS approved algorithm, it would be listed as non-approved and non-compliant on the FIPS 140-2 validation certificate.

A.2 Use of non-NIST-Recommended Asymmetric Key Sizes and Elliptic Curves

Applicable Levels:	*All*
Original Publishing Date:	*09/12/2005*
Effective Date:	*09/12/2005*
Last Modified Date:	*08/07/2017*
Relevant Assertions:	*AS.01.12*
Relevant Test Requirements:	*TE01.12.01*

Relevant Vendor Requirements:	*VE.01.12.01*

Background

The NIST **FIPS 186-4** standard allows (in Section 6.1.1) the use of the non-NIST-Recommended curves in the ECDSA algorithm in the approved mode. Similarly, the NIST **SP 800-56A Rev2** standard allows (in Section 5.5.1.2) the use of the non-NIST-Recommended curves in the elliptic-curve-based key agreement methods, EC Diffie-Hellman and ECMQV. **IG D.8** further allows the use in the approved mode of non-approved methods (that is, not compliant with **SP 800-56A** or **SP 800-56A Rev2**) in the key agreement schemes[1]; these non-approved but allowed methods may utilize the non-NIST-Recommended curves.

Question/Problem

What are the rules for using the non-NIST-Recommended elliptic curves in the ECDSA signature algorithm and the ECC-based key agreement schemes the approved mode of operation?

Resolution

The CMVP allows the use of non-approved elliptic curves in the approved mode of operation providing:

– the algorithm implementation **shall** use approved underlying algorithms, such as the message digests,

– the security policy **shall** list all approved and non-approved curves that are implemented,

– the security policy **shall** indicate the associated security strength for all non-approved curves that are implemented. The vendor **shall** check that the curve is not singular; provide to the CMVP the information about the curve's underlying field (which **shall** be either of a prime order or of the order 2^m, where m is prime) and about the number of points on the curve; present the factorization of the number of points on the curve into a large prime n and a co-factor h as shown in FIPS 186-4; verify that h is within the limits established in Table 1 of Section 6.1.1 of FIPS 186-4; check that the curve is non-anomalous (the number of points on the curve is not equal to the size of the field), and that the MOV condition is met for all B ≤ 100. See **ANS X9.62** or the ECC textbooks for details, and

– if an algorithm (ECDSA or KAS) is listed on the certificate's approved line, the algorithm implementation **shall** have been CAVS-tested and validated for at least one NIST-Recommended curve.

A.3 Vendor Affirmation of Cryptographic Security Methods

Applicable Levels:	*All*
Original Publishing Date:	*01/25/2007*
Effective Date:	*01/25/2007*
Last Modified Date:	*04/23/2012*
Relevant Assertions:	*AS.01.12*
Relevant Test Requirements:	*TE01.12.01*
Relevant Vendor Requirements:	*VE.01.12.01*

Background

A cryptographic module **shall** implement at least one approved security function used in an approved mode of operation. Non-approved security functions may also be included for use in non-approved modes of operation or allowed for use in an approved mode of operation. Documentation **shall** list all security functions, both

---

[1] This provision stays in effect until the 2017 transition documented in SP 800-131A Rev1 is fully implemented.

approved and non-approved, that are employed by the cryptographic module and **shall** specify all modes of operation, both approved and non-approved. The vendor **shall** provide a validation certificate for all approved cryptographic algorithms. The tester **shall** verify that the vendor has provided validated certificate(s) as described above.

Questions/Problems

For approved security functions, approved random number generators or approved key establishment techniques specified in FIPS 140-2 Annexes A, C, and D, if CAVP testing is not available, can the approved methods be used in FIPS mode, and if so, how **shall** it be tested and annotated on the module validation certificate and security policy?

Resolution

As new methods are published and approved, they will be added to the relevant FIPS 140-2 Annexes. The annexes may reference FIPS 140-2 Implementation Guidance for methods allowed in lieu of approved methods.

1. If new approved methods (e.g. NIST FIPS, Special Publication, etc.) are added to the Annexes which provides a new method that did not exist before (e.g. key establishment), until such time that CAVP testing is available for the new method, the CMVP would continue to:

    – allow methods as provided by guidance (untested and listed as non-approved but *allowed* in FIPS mode); and

    – allow the vendor to implement the new approved method (untested, listed as approved and allowed in FIPS mode with the caveat *vendor affirmed*).

    Once testing is deployed by the CAVP to the testing laboratories:

    a. a transition period (e.g. n months) would be provided for new test reports received by the CMVP:

        ▪ during the transition period, a new approved method would either be listed as approved with a reference to a CAVP validation certificate, or as *vendor affirmed* if testing was not performed; and

        ▪ allow continued implementation of methods as provided by guidance (untested and listed as non-approved but *allowed* in FIPS mode).

    b. when the transition period ends, for newly received test reports:

        ▪ only approved methods that have been tested and received a CAVP validation certificate would be allowed. All other methods would be listed as non-approved and not allowed in an approved FIPS mode of operation.

    c. the vendor could optionally follow up with testing of un-tested vendor affirmed methods and if so, the reference to *vendor affirmed* would be removed and replaced by reference to the algorithm certificate. If there are no changes to the module, this change can be submitted under IG G.8 Scenario 1. If the module is changed, this change can be submitted under IG G.8 Scenarios 1, 3 or 5 as applicable.[2]

2. If new approved methods (e.g. NIST FIPS, Special Publication, etc.) are added to Annexes which provides a new method commensurate with those that currently exist (e.g. a new symmetric key algorithm, RNG, DRBG, hash, digital signature, etc.), until such time that CAVP testing is available for the new method, the CMVP would:

    – allow prior approved methods (tested and listed as approved); and

- allow the vendor to implement the new approved method (untested, listed as approved and allowed in FIPS mode with the caveat *vendor affirmed*)

Once testing is deployed by the CAVP to the testing laboratories:

a. a transition period (e.g. n months) would be provided for new test reports received by the CMVP:

- during the transition period, a new approved method would either be listed as approved with a reference to a CAVP validation certificate, or as *vendor affirmed* if testing was not performed.

b. when the transition period ends, for newly received test reports:

- only approved methods that have been tested and received a CAVP validation certificate would be allowed. All other methods would be listed as non-approved and not allowed in an approved FIPS mode of operation.

c. the vendor could optionally follow up with testing of prior un-tested vendor affirmed methods and if so, the reference to *vendor affirmed* removed and replaced by reference to the algorithm certificate. If there are no changes to the module, this change can be submitted under IG G.8 Scenario 1. If the module is changed, this change can be submitted under IG G.8 Scenarios 1, 3 or 5 as applicable .[1]

3. The Cryptographic Technology Group at NIST may determine that prior methods may be retroactively disallowed and moved to non-approved and not allowed in a FIPS mode of operation (e.g. DES). A Federal Register notice would be published with a transition period to allow migration from the no longer approved or allowed method.

4. For all approved methods, all applicable FIPS 140-2 requirements **shall** be met (e.g., key management, self-tests, etc.)

Additional Comments

Vendor Affirmed: a security method reference that is listed with this caveat has not been tested by the CAVP, and the CMVP or CAVP provide no assurance regarding its correct implementation or operation. Only the vendor of the module affirms that the method or algorithm was implemented correctly.

The users of cryptographic modules implementing vendor affirmed security functions must consider the risks associated with the use of un-tested and un-validated security functions.

Test Requirements

Until the FIPS 140-2 DTR and CRYPTIK tool are updated and released, please provide the following information under **VE.01.12.01** and **TE.01.12.01**.
Required Vendor Information

VE.01.12.03: The vendor **shall** provide a list of all vendor affirmed security methods.

VE.01.12.04: The vendor provided nonproprietary security policy **shall** include reference to all vendor affirmed security methods.

Required Test Procedures

---

[1] If the change is security relevant either to the module or the method, then IG G.8 Scenarios 3 or 5 would be applicable depending on the extent of the changes. If for example there was a non-security relevant change to the module not associated with the security method implementation, IG G.8 Scenario 1 could be applicable.

TE01.12.03: The tester **shall** verify that the vendor has provided the list of vendor affirmed security methods as described above.

TE01.12.04: The tester **shall** verify that the vendor provided documentation specifies how the implemented vendor affirmed security methods conform to the relevant standards.

Required Use of "Vendor Affirmed" Caveat

All cryptographic methods that are approved and *vendor affirmed* **shall** be specified on the certificate and in the security policy, and be annotated with, in addition to the other required caveats as applicable, the caveat (vendor affirmed: *FIPS or NIST Special Publication #*). See IG G.13 for vendor affirmation examples.

A.4 moved to W.7

A.5 Key/IV Pair Uniqueness Requirements from SP 800-38D

Applicable Levels:	*All*
Original Publishing Date:	*03/10/2009*
Effective Date:	*03/10/2009*
Last Modified Date:	*12/04/2017*
Relevant Assertions:	*AS01.12, AS07.03, AS14.02, AS14.03*
Relevant Test Requirements:	
Relevant Vendor Requirements:	

Background

SP 800-38D was added to FIPS 140-2 Annex A on December 18, 2007. **SP 800-38D** requires that "the probability that the authenticated encryption function ever will be invoked with the same IV and the same key on two (or more) distinct sets of input data **shall** be no greater than 2^{-32}".

One difficulty of testing the module's compliance with this requirement comes from the fact that each module is tested independently while **SP 800-38D** demands that the probability of the (Key, IV) pair collision between all modules at all times should be sufficiently low to ensure cryptographic strength.

Question/Problem

How shall a cryptographic module satisfy these requirements?

Resolution

An AES GCM key may either be generated internally or entered into the cryptographic module.

Techniques for generating an IV that are acceptable for the purposes of FIPS 140-2 validation are listed below.

1. Construct the IV in compliance with the provisions of a peer-to-peer industry standard protocol whose mechanism for generating the IVs for AES GCM has been reviewed and deemed acceptable by the appropriate validation authorities and subject to the additional requirements established in this guidance. The current list of acceptable protocols is shown below:

    a. TLS 1.2 GCM Cipher Suites for TLS, as described in RFCs 5116 , 5288 and 5289 provisions;

b. IPSec-v3 protocol, as described in RFCs 4106, 7296 and 5282.

c. MACsec with GCM-AES-128, GCM-AES-256, GCM-AES-XPN-128 and GCM-AES-XPN-256 Cipher Suites, as described in IEEE 802.1AE (MACsec) and its amendments.

The following are additional specific requirements for each acceptable protocol for the purposes of FIPS 140-2 validation.

TLS protocol IV generation

If an IV is constructed according to the TLS protocol, then this IV may only be used in the context of the AES GCM mode encryption within the TLS protocol.

If the vendor claims that the IV generation is in compliance with the TLS specification and only for use within the TLS protocol, then the module's Security Policy and the Validation Test Report **shall** explicitly state the module's compatibility with TLSv1.2 and the module's support for acceptable GCM ciphersuites from SP 800-52 Rev 1, Section 3.3.1.

For the purposes of this Guidance, the module may demonstrate its compliance with the rules for TLS compliance in one of the following two ways.

i) The operations of one of the two parties involved in the TLS key establishment scheme **shall** be performed *entirely within* the cryptographic boundary of the module being validated. The testing laboratory **shall** check the module implementation and verify that the keys for the client and server negotiated in the handshake process (client_write_key and server_write_key) are compared and the module aborts the session if the key values are identical; or

ii) The laboratory **shall** check the TLS protocol implementation that relies on the module being validated against an independently developed instance of TLS, such as the many TLS client test sites on the Internet, verify that a session is successfully established, which implies that the client_write_key and server_write_key values are derived correctly, and the following condition **shall** be met:

The module's implementation of AES GCM is used together with an application that runs either inside or outside the module's cryptographic boundary. This application negotiates the protocol session's keys and the 32-bit nonce value of the IV. The nonce is positioned where there is the "name" field in Scenario 3 of this Guidance.

Whether an implementation is using the i) or the ii) path to meet the compliance requirements, the counter portion of the IV **shall** be set by the module *within* its cryptographic boundary and the requirements of Scenario 3 of this Guidance for the counter field (including the IV restoration conditions) are satisfied.

The implementation of the nonce_explicit management logic inside the module **shall** ensure that when the nonce_explicit part of the IV exhausts the maximum number of possible values for a given session key (e.g., a 64-bit counter starting from 0 and increasing, when it reaches the maximum value of 2^{64} -1), either party (the client or the server) that encounters this condition triggers a handshake to establish a new encryption key – see Sections 7.4.1.1 and 7.4.1.2 in RFC 5246. A statement to that effect **shall** be included in the Security Policy and Validation Test Report.

IPSec protocol IV generation

If an IV is constructed in compliance with the IPSec protocol, then this IV may only be used in the context of the AES GCM mode encryption within the IPSec protocol.

If the vendor claims that the IV generation is in compliance with the IPSec specification and only for use within the IPSec protocol then the module's Security Policy and the Validation Test Report **shall** explicitly state the module's compliance with RFC 4106 and/or RFC 5282 (depending on the protocols supporting GCM). The Security Policy and Validation Test Report **shall** also state that the module uses RFC 7296 compliant IKEv2 to establish the shared secret SKEYSEED from which the AES GCM encryption keys are derived.

Similar to the allowances shown above for the TLS implementations, the module may demonstrate its compliance with the rules for IPSec compliance in one of the following two ways.

i) The operations of one of the two parties involved in the IKE key establishment scheme **shall** be performed entirely within the cryptographic boundary of the module being validated. The testing laboratory **shall** check the module implementation and verify that the two keys established by IKEv2 for one security association (one key for encryption in each direction between the parties) are not identical and abort the session if they are; or

ii) The laboratory **shall** check the IPSec protocol implementation that relies on the module being validated against an independently developed instance of IPSec with IKEv2, verify that a session is successfully established, which implies that the two keys established by IKEv2 are derived correctly, and the following condition **shall** be met:

The module's implementation of AES GCM is used together with an application that runs either inside or outside the module's cryptographic boundary. This application negotiates the protocol session's keys and the value in the first 32 bits of the nonce (see below).

Note that in RFC 5282 the term for what is called an IV in SP 800-38D and in this IG is "nonce", while the term "IV" in RFC 5282 refers only to the last 64 bits of the "nonce" field. In other words, IPSec requires four octets of salt followed by eight octets of deterministic nonce. Whether an implementation is using the i) or the ii) path to meet the compliance requirements, the construction of the last 64 bits of the "nonce" (the IV in RFC 5282) for the purposes of FIPS 140-2 validation **shall** be deterministic (e.g., using a counter) and satisfy one of the IV restoration conditions defined in Scenario 3 of this Implementation Guidance.

The implementation of the management logic for the last 64 bits of the "nonce" (the IV in RFC 5282) inside the module **shall** ensure that when the IV in RFC 5282 exhausts the maximum number of possible values for a given security association (e.g., a 64-bit counter starting from 0 and increasing, when it reaches the maximum value of 2^{64} -1), either party to the security association that encounters this condition triggers a rekeying with IKEv2 to establish a new encryption key for the security association – see RFC 7296. A statement to that effect **shall** be included in the Security Policy and Validation Test Report.

MACsec protocol IV generation

A typical implementation of the MACsec protocol includes the components shown in Figure 1 below. The generation and management of IV's for the GCM cipher suites in the MACsec protocol is distributed among the three components. For the purposes of a FIPS 140-2 validation, the cryptographic functionality relevant for MACsec in each component **shall** be validated as a separate module. The requirement in Section 9.1 in **SP 800-38D** to contain the IV "generation unit" within a module boundary is satisfied by the composition of the **Peer**, **Authenticator** and optional **Authentication Server** modules. All modules – **Peer**, **Authenticator**, and, if applicable, **Authentication Server**, should be validated (or re-validated) after this Implementation Guidance takes effect, so that they all comply with the applicable requirements of this IG. While all modules are validated separately by the CMVP, each module's Security Policy **shall** tell what this module's role is in the MACsec protocol, explain what the module does in support of the IV generation for the MACsec's use of AES GCM, and state that when supporting the MACsec protocol in the approved mode, the module should only be used together with the CMVP-validated modules providing the remaining <**Peer**, **Authenticator**, …> functionalities.
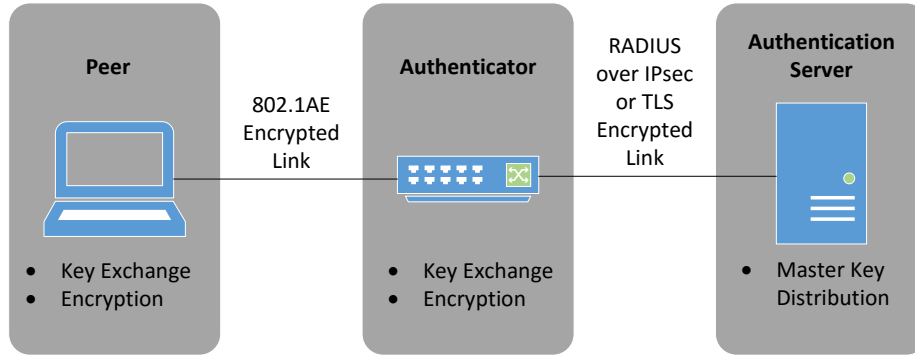
Figure 1. MACsec protocol components

The link between the **Authenticator** and the optional **Authentication Server** is typically implemented by the RADIUS protocol, which is a de-facto industry standard for communication to authentication servers. All references to RADIUS in this Guidance are applicable only to the use of this protocol in the MACsec context. To provide security the RADIUS traffic should be tunneled over an IPsec (cf. RFC 3162) or TLS channel (cf. RFC 6614). All configuration instructions for the link between the **Authenticator** and the **Authentication Server** **shall** be provided in the Security Policy of the module.

The **Peer** and the **Authenticator** Modules Security Policies **shall** state that the link between the **Peer** and the **Authenticator** should be secured to prevent the possibility for an attacker to introduce foreign equipment into the local area network – see Section 7.3 in IEEE Std 802.1X-2010.

2. The IV may be generated internally at its entirety *randomly*. In this case,

   - The generation **shall** use an Approved DRBG and

   - The DRBG seed **shall** be generated inside the module's physical boundary.

   - The IV length **shall** be at least 96 bits (per **SP 800-38D**). See Additional Comments for the discussion of why an IV of less than 96 bits may not be generated randomly and concatenated to the remaining part of an IV.

   - Since the IV is generated internally and randomly and must be at least 96 bits in length, requirements from IG 7.15 **shall** apply to the module to claim at least 96 bits of entropy strength.

   A statement to that effect **shall** be included in the Security Policy and Validation Test Report.

3. If an AES GCM Key is generated either internally or externally and the IV is constructed at its entirety internally *deterministically* then the requirement of **SP 800-38D** quoted in the Background section above will be modified. Instead of requiring that the probability of any (key, IV) collision anywhere in the Universe at all times did not exceed 2^{-32}, it will only be required that for a given key distributed to one or more cryptographic modules, the (key, IV) collision probability would not exceed 2^{-32}. This is equivalent to the requirement that for any key distributed to one or more modules the probability of a collision between the deterministically-generated IVs is no greater than 2^{-32}.

The module **shall** use at least 32 bits of the IV field as a name and use at least 32 bits as a deterministic non-repetitive counter for a combined IV length between 64 bits and 128 bits. The name field **shall** include an encoding of the module name and the name construction **shall** allow for at least 2^{32} different names. For example, if the module name is such that it consists of at least 8 hexadecimal characters then this condition is satisfied, since 16^8 is no smaller than (indeed, equal to) 2^{32}. Alternatively, if the name consists of at least 6 alphanumerical characters, each having at least 62 values, then this is also sufficient. Even though not all possible names are equally likely to be used, the fact that the modules can possibly have at least 2^{32} different names will be sufficient to meet this requirement.

The implementation of the deterministic non-repetitive counter management logic inside the module **shall** ensure that when the counter part of the IV exhausts the maximum number of possible values for a given session key (e.g., a 32-bit counter starting from 0 and increasing, when it reaches the maximum value of $2^{32}-1$) the encryptor **shall** abort the session.

Further, at least one of the IV restoration conditions **shall** be satisfied for the deterministic non-repetitive counter.

The IV restoration conditions are as follows (for additional details, see Section 9.1 of **SP 800-38D**):

(1) The module's memory **shall** be set in such way that it will reset to the last IV value used in case the module's power is lost and then restored. (This condition is enforced by the module and **shall** be tested by a testing lab.)

(2) There will be a human operator who will reset the IV to the last one used in case the module's power is lost and then restored. (This condition is not enforced but **shall** be stated in the module's Security Policy, under the "User Guide" heading.)

(3) In case the module's power is lost and then restored, a new key for use with the AES GCM encryption/decryption **shall** be established. (This condition may or may not be enforced but **shall** be stated in the module's Security Policy, under the "User Guide" heading.)

A statement explaining how the deterministic IV generation is performed and how the IV restoration conditions are met **shall** be included in the Security Policy and Validation Test Report.

NOTE. The module does not need to comply with any particular Scenario (1 – 3) shown in this section of the IG. Meeting the rules of any one of these Scenarios, whether protocol-dependent or not, when generating the IVs for AES GCM is sufficient. The Security Policy **shall** explain the rules under which the module must operate in compliance with this Implementation Guidance.

4. If an implementation does not meet the requirements of any of the Scenarios 1 through 3 explained above in this Implementation Guidance, the vendor may present their own proof of the compliance with the **SP 800-38D** requirement stated in the **Background** section of this IG. The burden of proof is on the vendor. The testing laboratory **shall** review the proof and verify its correctness. Each proof will be examined by the CMVP reviewers who will make the final determination of the proof's validity.

Additional Comments

1. This Implementation Guidance does not introduce any new requirements. On the contrary, the purpose of this Implementation Guidance is to relax some of the **SP 800-38D** requirements. This relaxation is needed because **SP 800-38D** imposes some system-wide requirements, including those that concern the probabilities of the (key, IV) pair collisions. The compliance with such system-wide requirements cannot be tested within the scope of the CMVP program where each module is validated separately.

In some Scenarios allowed by this IG, the (key, IV) collision probability is calculated as it applies to one module. To reconcile this interpretation of the **SP 800-38D** requirement with the security goals stated in **SP 800-38D**, the module needs to operate within the certain limits established by this Implementation Guidance. One possibility is for the module to comply with the specific confines of the known industry protocols. If this is the case, the reliance on the properties of the protocol allows the CMVP to modify the rules of **SP 800-38D** without introducing any security risks or exposures.

Scenarios 2 and 3 in the **Resolution** section are protocol-independent.

2. When the module uses a (non-protocol-specific) deterministic IV construction, the name field provides an assurance that the IVs are not repeated even when they are generated independently by different modules, possibly manufactured by different vendors. If this name field is only 32 bits long, then it is barely sufficient to provide for the 2^{32} different values. Operators of the modules that use the 32-bit long names need to ensure that there is no possibility of name collisions in the systems they operate. When it is not possible to control the name assignment (as in the case when a session that uses an AES GCM encryption runs over a network managed without a central control over the names of the modules) then a 32-bit field may be insufficient. In such cases, it is highly recommended to

switch to 64-bit or even 96-bit long names and limit the number of encrypted blocks under the same key to 2^{32}.

3. As stated in this Implementation Guidance, if the entire IV is generated randomly, the length of the random field **shall** be at least 96 bits. The reason for it is that with 2^{32} possible AES GCM encryptions under the same key, the probability of having at least one (key, IV) collision (i.e., an IV collision, as the key stays the same) can be estimated to be of the order of 2^{-32}. However, to maintain the same probability of collisions in the case of a 64-bit random field, one would have to reduce the maximum number of AES GCM encryptions with the same key to only 2^{16}.

4. Including the module's name in the IV field does not amount to a passphrase-based key derivation. The IV is not a key. Their cryptographic properties are different.

5. The methods presented in this IG apply to the module's generation of an IV parameter for the AES GCM encryption. When an IV is used for decryption, the responsibility for the IV generation lies with the party that performs the AES GCM encryption therefore none of the SP 800-38D requirements and none of the Scenarios presented in this Guidance are applicable to the module performing the decryption.

6. Some proprietary implementations of MACsec allow the static configuration of a pre-shared key for the Security Association Key (SAK) used by the protocol. The static configuration of a pre-shared SAK **shall not** be used in the approved mode of operation.

7. If any of the IETF or IEEE documents referenced in this IG become obsolete or get updated by another IETF RFC or an IEEE standard, then the new document number **shall** be considered the replacement of the number listed in this Guidance.

A.6 moved to W.8

A.7 moved to W.9

A.8 Use of HMAC-SHA-1-96 and Truncated HMAC

Applicable Levels:	*All*
Original Publishing Date:	*03/02/2015*
Effective Date:	*03/02/2015*
Last Modified Date:	*05/10/2017*
Relevant Assertions:	*AS.01.12*
Relevant Test Requirements:	*TE.01.12.01, TE.01.12.02*
Relevant Vendor Requirements:	

Background

The Keyed-Hash Message Authentication Code (HMAC) function is used by the message sender to produce a value, called the MAC, which is formed by condensing the secret key and the message input. The HMAC function may use any approved hash algorithm. HMAC is documented in FIPS 198-1.

Some internet protocols such as IPsec and SSH use HMAC-SHA-1 and truncate the MAC to 96 (leftmost) bits. Other implementations use HMAC-SHA-384 truncated to 192 bits. Some other truncations may also be considered by implementers.

Question/Problem

Can a truncated HMAC be used in the approved mode?

Resolution

According to SP 800-107rev1, published August 2012, the truncated forms of an approved HMAC are the approved algorithms if an HMAC output is truncated to its λ leftmost bits with $\lambda \geq 32$. In particular, HMAC-SHA-1-96 and HMAC-SHA-384-192 are approved algorithms and can be used in the approved mode of operation. This includes their use as an approved integrity technique required in Section 4.6.1 of FIPS 140-2 and as an approved authentication technique when performing the software/firmware load test described in Section 4.9.2 of FIPS 140-2.

Additional Comments

1. In compliance with the transition requirements specified in SP 800-131Arev1, any key for a full-length-output HMAC or a truncated HMAC that possesses less than 112 bits of strength **shall not** be used in approved mode.

2. When a truncated HMAC is used in the approved mode, the corresponding full-output HMAC **shall** have a CAVP algorithm certificate. The module's validation certificate needs to show this CAVP certificate and does not need to reference the truncated HMAC. The use of the latter **shall** be shown in the module's security policy.

3. The security of the truncated HMAC values is addressed in SP 800-107rev1. The permission to use in the approved mode an HMAC output truncated to (the minimum of) 32 bits does not contradict the requirement of SP 800-131Arev1 of providing at least 112 bits of equivalent encryption strength. The cryptographic strength of HMAC depends primarily on the strength of the HMAC key. See SP 800-107rev1 for details.

4. While SP 800-107rev1 allows the truncation of HMAC to its λ leftmost bits with $\lambda \geq 32$, that Special Publication discourages the HMAC truncations to less than 64 bits. For the purposes of the FIPS 140-2 validations, it is the 32-bit requirement that will be enforced.

5. HMAC-SHA-3 is subject to the same truncation rules as the other HMACs that utilize the approved hash functions.

A.9 XTS-AES Key Generation Requirements

Applicable Levels:	*All*
Original Publishing Date:	*12/28/2015*
Effective Date:	*07/01/2016*
Last Modified Date:	*12/04/2017*
Relevant Assertions:	*AS.01.12*
Relevant Test Requirements:	*TE.01.12.01*
Relevant Vendor Requirements:	

Background

XTS-AES is approved in **SP 800-38E** by reference to **IEEE Std. 1619-2007**. The IEEE standard specifies a key, denoted by *Key*, that is 256 [or 512] bits long; *Key* is then parsed as the concatenation of two AES keys, denoted by *Key_1* and *Key_2*, that are 128 [or 256] bits long. Sec. D.4.3 (pp. 31-32) explains that XTS-AES

differs from the generic XEX construction (due to Rogaway) in that $Key\_1 = Key\_2$ for XEX but not for XTS-AES. Annex D of the IEEE standard is labeled as informative, not normative and there are no other requirements on the generation of Key otherwise in that standard.

Question/Problem

Misuse of XTS-AES with a class of improper keys results in a security vulnerability. An implementation of XTS-AES that improperly generates *Key* so that $Key\_1 = Key\_2$ is vulnerable to a chosen ciphertext attack that would defeat the main security assurances that XTS-AES was designed to provide. In particular, by obtaining the decryption of only one chosen ciphertext block in a given data sector, an adversary who does not know the key may be able to manipulate the ciphertext in that sector so that one or more plaintext blocks change to any desired value. Rogaway illustrates the attack for disallowed parameterizations of XEX (without fully exploring its consequences) in Sec. 6 of his 2004 paper Efficient Instantiations of Tweakable Blockciphers and Refinements to Modes OCB and PMAC, available at http://web.cs.ucdavis.edu/~rogaway/papers/offsets.pdf.

Resolution

Key_1 and *Key_2* are intended to be distinct keys, and they must each be generated pseudo-randomly to comply with approved key generation guidelines. The module **shall** check explicitly that
$Key\_1 \neq Key\_2$, regardless of how *Key_1* and *Key_2* are obtained. See section **Additional Comments** below for further implementation considerations. In addition, the CST testing lab **shall** document in TE.01.12.01 of the Test Report how the module meets the above requirement.

Additional Comments

This interpretation of the IEEE standard is consistent with the requirements on the generation of secret keys for other NIST approved cryptographic algorithms, namely, from a cryptographically strong pseudorandom source or approved KDF and with support from a good entropy source.

The check for $Key\_1 \neq Key\_2$ **shall** be done at any place BEFORE using the keys in the XTS-AES algorithm to process data with them. This allows for choosing an appropriate place for implementing the check, anywhere from within the algorithm boundary to the module boundary.

A.10 Requirements for Vendor Affirmation of SP 800-38G

Applicable Levels:	*All*
Original Publishing Date:	*06/17/2016*
Effective Date:	*06/17/2016*
Transition End Date:	
Last Modified Date:	
Relevant Assertions:	*AS.01.12*
Relevant Test Requirements:	*TE01.12.03-04*
Relevant Vendor Requirements:	

Background

SP 800-38G was added to FIPS 140-2 Annex A on April 6, 2016. IG A.3 was added on January 25, 2007. Until CAVP testing for **SP 800-38G** is available, IG A.3 is applicable. **SP 800-38G** contains the description of two methods, FF1 and FF3, for format-preserving encryption.

Question/Problem

To claim *vendor affirmation* to **SP 800-38G**, which sections of the standard need to be addressed and what are the documentation requirements?

Resolution

Vendor affirmation for **SP 800-38G**, *Recommendation for Block Cipher Modes of Operation: Methods for Format-Preserving Encryption* includes

1) validation testing for the underlying encryption function (AES),
2) documenting in the module's Security Policy the selection of the format-preserving algorithms: FF1, FF3, or both,
3) documenting in the module's Security Policy the lengths of the following parameters from **SP 800-38G**: *radix*, *minlen*, *maxlen*, and, for the FF1 scheme, *maxTlen*. The value of *maxTlen* may not exceed $2^{32} - 1$,
4) an affirmation of the implemented algorithms' compliance to the functions (FF1, FF3) specified in **SP 800-38G**.

Additional Comments

1. **SP 800-38G** requires that the underlying encryption function used in the format-preserving algorithms is an approved block cipher operating on the 128-bit blocks of data. As of June 2016, and for the foreseeable future, the only such function is the AES algorithm. The length of an AES key may be 128, 192, or 256 bits.

2. The module's validation certificate **shall** include an entry for the AES algorithm that has been tested and is being used as an underlying encryption function in the format-preserving algorithms. The AES algorithm certificate **shall** show the validation of the mode of AES that has been used as an underlying block cipher (encryption function) in format preserving encryption.

   It is sufficient to test the underlying AES encryption function in any approved mode of AES that uses this underlying encryption engine.

3. No special acronym is required in the validation certificate to annotate the module's compliance with **SP 800-38G**. Use "AES" as with any other approved mode of AES.

4. The vendor **shall** document the **SP 800-38G**-compliant format preserving encryption in the list of the vendor affirmed security methods in TE.01.12.03 of the Test Report. In TE.01.12.04, the vendor **shall** demonstrate how their implementation complies with the appropriate provisions of **SP 800-38G**; in particular, with the requirement that the *radix*, *minlen*, *maxlen* and, if applicable, *maxTlen* parameters satisfy the constraints defined in Sections 5.1 and 5.2 of **SP 800-38G** for FF1 and FF3, correspondingly. If the FF1 scheme is used, the validation Test Report **shall** also show the possible byte lengths *t* of the tweak used by this module.

A.11 The Use and the Testing Requirements for the Family of Functions defined in FIPS 202

Applicable Levels:	*All*
Original Publishing Date:	*08/01/2016*
Effective Date:	*08/01/2016*
Last Modified Date:	*08/07/2017*
Relevant Assertions:	*AS.01.12*
Relevant Test Requirements:	*TE01.12.01*
Relevant Vendor Requirements:	*VE.01.12.01*

Background

FIPS 202 was published in August 2015 and added to **FIPS 140-2** Annex A on September 17, 2015. This standard includes the specifications for the SHA-3 family of hash functions: SHA3-224, SHA3-256, SHA3-

384 and SHA3-512, as well as for the two extendable-output functions, SHAKE128 and SHAKE256. CAVS testing for all of these functions became available on January 29, 2016.

Question/Problem

1. Are there any limitations on the use of hash functions defined in FIPS 202 in the CMVP-validated cryptographic modules?

2. What are the validation testing requirements for the FIPS 202-compliant algorithms and the higher-level cryptographic algorithms that are using the functions defined in FIPS 202? In particular, how to address the case when CAVS testing is available for a higher-level cryptographic algorithm when this algorithm uses the hash functions defined in FIPS 180-4 but there is still no testing when the same higher-level algorithm uses the FIPS 202 functions?

3. Which self-tests are required for the FIPS 202-defined functions?

Resolution

1. To be used in the FIPS approved mode of operation, the SHA-3 hash functions may be implemented either as part of an approved higher-level algorithm, for example, a digital signature algorithm, or as the standalone functions. The SHAKE128 and SHAKE256 extendable-output functions may only be used as the standalone algorithms.

2. The validation, testing and the certificate documentation requirements are as follows.

    a. Every implementation of each SHA-3 and SHAKE function **shall** be tested and validated on the appropriate OS.

    b. If any of the SHA-3 hash functions are used as part of a higher-level algorithm and the CAVS testing that supports SHA-3 is available for this higher-level algorithm, then, upon the expiration of a transitional period defined when such CAVP testing becomes available, to use the higher-level algorithm in the approved mode the vendor **shall** obtain a CAVP certificate for this algorithm. If the vendor does not obtain such a certificate, then the higher-level algorithm that uses the SHA-3 functions may not be used in the approved mode and **shall** not be listed on the module's validation certificate.

    c. If any of the SHA-3 hash functions are used as part of a higher-level approved algorithm and the CAVS testing that supports SHA-3 is NOT yet available for this higher-level algorithm or the transitional period has not yet expired, then to use this higher-level algorithm in the approved mode the vendor **shall** claim the vendor affirmation for this algorithm. This **shall** be accompanied by obtaining the CAVS certificates for the SHA-3 functions used in the higher-level algorithm. If the module implemented the same higher-level algorithm with a FIPS 180-4 hash function and there is a corresponding entry on the approved line of the module's validation certificate, then the vendor affirmation of the same algorithm using SHA-3 does not need to be shown separately on the certificate's approved line but **shall** be documented in the module's Security Policy.

    d. Until CAVS testing for a higher-level algorithm with the SHA-3 hash functions is available, IG A.3 is applicable.

3. The self-test requirements.

    a. At the minimum, the cryptographic module **shall** perform a known answer test for one of the functions defined in FIPS 202: SHA3-224, SHA3-256, SHA3-384, SHA3-512, SHAKE128 and SHAKE256, no matter how many of these functions the module may be designed to use. A known answer test for a hash function may be performed as part of the known answer test of a higher-level approved algorithm.

    b. If a SHA-3 hash function is used as part of a higher-level approved algorithm that is tested by the CAVP, then the module **shall** perform a known answer test for this higher-level algorithm. No additional power-on self-tests for any of the FIPS 202-compliant functions implemented by this module are required.

    c. If a SHA-3 hash function is used as part of a higher-level approved algorithm that is vendor affirmed but not tested by the CAVP or the transitional period has not yet expired, then the module **shall** perform a known answer test either for the higher-level algorithm or for one of the SHA-3 hash functions this higher-level algorithm uses.

    d. No conditional self-tests are required in support of the FIPS 202-compliant algorithms.

Additional Comments

1. The reason for requiring a known answer test for only one of the FIPS 202-compliant hash functions is that all of these functions, including SHAKE128 and SHAKE256, rely on the same underlying Keccak-p permutation. Note that this is different from the SHA-1 and SHA-2 self-test requirements where separate self-tests are needed for SHA-1, SHA-256 and SHA-512, if the module is designed to use these hash functions.

2. If the module implements several Keccak-p permutation engines, a self-test **shall** be performed for more than one implementation of the FIPS 202-defined functions so that each permutation engine's implementation is self-tested.

3. Examples for how to annotate the use of the FIPS 202 algorithms in the module's validation certificate.

    a. **SHA-3 (Cert. #55)**. This demonstrates that one or more of the following functions: SHA3-224, SHA3-256, SHA3-384, SHA3-512, SHAKE128, SHAKE256 is implemented in the module and tested by the CAVP. There is no separate acronym for the SHAKE functions.

    b. **RSA (SHA-3 Cert. #55, vendor affirmed)**. This is the case when the implementation of the RSA signature algorithm supported by the module uses only the SHA-3 hash functions, and no CAVP testing is available for the configuration of the RSA algorithm that uses the SHA-3 hash functions (or if such testing is available but the transitional period announced upon the introduction of this testing has not yet expired.)

    c. **DSA (Cert. #200)**. [No change from the existing notation.] The module's DSA algorithm implementation(s) may use both the FIPS 180-4 and the FIPS 202 hash functions. One entry in the module's validation certificate is sufficient. The Security Policy **shall** indicate that the use of the DSA with the SHA-3 hash functions is vendor affirmed if the CAVP testing for the DSA that uses the SHA-3 hash functions is not yet available.

4. The future updates of this Implementation Guidance will keep track of the testing availability status for the approved higher-level algorithms that might use the SHA-3 hash functions.

A.12 Requirements for Vendor Affirmation to the Addendum to SP 800-38A

Applicable Levels:	*All*
Original Publishing Date:	*11/15/2016*
Effective Date:	*11/15/2016*
Transition End Date:	
Last Modified Date:	
Relevant Assertions:	*AS.01.12, AS.09.16*
Relevant Test Requirements:	*TE01.12.03-04, TE.09.16.01*
Relevant Vendor Requirements:	*VE.01.12.01*

Background

SP 800-38A was published in December 2001. This standard defines several approved modes of AES: ECB, CBC, CFB, OFB, and CTR. In a further development, in October 2010, NIST published an **Addendum to SP 800-38A** to expand the definition of the Cipher Block Chaining mode (CBC). The reason for writing this Addendum was that the version of the AES CBC mode in the original publication of **SP 800-38A** assumed that plaintext consisted of a whole number of the 128-bit-long blocks of data. When this requirement was not met, Appendix A of **SP 800-38A** showed how to "pad" a partial block of plaintext to its full 128-bit size (the length of an AES encryption block). Unfortunately, this also resulted in an expansion of the ciphertext.

The 2010 **Addendum to SP 800-38A** introduces three different modes of making the complete 128-bit plaintext blocks when performing the AES CBC encryption. These modes are: CBC-CS1, CBC-CS2 and CBC-CS3. As explained in the **Addendum to SP 800-38A**, the "CS" notation stands for "ciphertext stealing", meaning that the bits needed to make all plaintext blocks complete are taken from one of the ciphertext blocks. These three modes do not require the ciphertext expansion.

The **Addendum to SP 800-38A** was added to FIPS 140-2 Annex A on November 24, 2010. Until CAVP testing for the three modes introduced in the **Addendum to SP 800-38A** is available, IG A.3 is applicable. The current Guidance explains the rules for vendor affirmation when any of the three AES modes, CBC-CS1, CBC-CS2 and CBC-CS3 are used in the approved mode.

Question/Problem

To claim *vendor affirmation* to the **Addendum to SP 800-38A**, which sections of the standard need to be addressed and what are the documentation requirements?

Resolution

The entire text of the **Addendum to SP 800-38A** is applicable. To simplify the notation on the module's validation certificate, if the module supports any of the three modes of AES: CBC-CS1, CBC-CS2 and CBC-CS3 and the module's validation certificate already includes an AES entry on the approved line with an AES algorithm certificate showing that testing has been performed for the CBC mode of AES, then no additional entries on the module's certificate are needed. If the module's certificate does not show an AES algorithm entry that includes the CBC mode, then a separate vendor affirmation entry AES-CBC-CS **shall** be included in the module's certificate, even if the module's certificate already shows an AES algorithm certificate on the approved line.

The Security Policy **shall** show vendor affirmation to the applicable modes: CBC-CS1, CBC-CS2 or CBC-CS3. The Test Report **shall** list these instances of vendor affirmation in the applicable assessments.

Annotation

An entry **AES-CBC-CS (vendor affirmed)** may be required on the approved line. It is not necessary to specify in the module's certificate which of the AES CBC ciphertext stealing modes (CS1, CS2, or CS3) are implemented. The details will be provided in the module's Security Policy and the Test Report.

Additional Comments

1. There are several ways that the documentation requirements for vendor affirmation to the CBC-CS1, CBC-CS2 and CBC-CS3 modes of AES could be addressed. One solution would not require any new entries in the module's certificate. However, this would misinform the user when the module implemented the AES CBC ciphertext stealing modes and the vendor affirmed the implementation's compliance with the **Addendum to SP 800-38A**. Another alternative would be to always require a vendor affirmation entry if any of the three modes in questions were supported in the approved mode.

   This Guidance shows a compromise solution that relies on the explanations in the module's Security Policy and the Test Report with an exception of an unlikely case when the module supports one of the AES CBC ciphertext stealing modes, but not the AES CBC mode defined in **SP 800-38A**. In this case, a separate vendor affirmation entry in the module's certificate is highly informative and is, therefore, required.

2. If the module supports one of the approved modes of AES in the approved mode then it is necessary to perform the AES known answer tests for both the encryption and the decryption functions, if both functions are supported. The known answer tests may be performed using any approved mode of AES shown in the module's validation certificate. This requirement applies even if the module's implementation does not include any tested AES modes and the module's only approved modes of AES are the vendor-affirmed AES CBC ciphertext stealing modes addressed in this Guidance.

3. The use of the CBC mode with or without the ciphertext stealing requires the presence of an IV parameter. The requirements on the IV for its use in the CBC-CS1, CBC-CS2 and CBC-CS3 modes of AES are the same as those when the IV is used in the AES CBC mode. Specifically, the IV does not need to be secret but must be unpredictable. See Appendix C of **SP 800-38A** for the precise meaning of the unpredictability in this context.

A.13 SP 800-67rev1 Transition

Applicable Levels:	*All*
Original Publishing Date:	*05/10/2017*
Effective Date:	*05/10/2017*
Last Modified Date:	*03/27/2018*
Relevant Assertions:	*AS.01.12*
Relevant Test Requirements:	*TE.01.12.01*
Relevant Vendor Requirements:	*VE.01.12.01*

Background

SP 800-67rev1 was published in January 2012 and added to FIPS 140-2 Annex A in February 2017. **SP 800-67rev1** added a requirement prohibiting users from performing more than 2^{32} 64-bit data block encryptions under the same three-key Triple-DES key. In an earlier version of **SP 800-67**, this requirement was a "should not" rather than a "**shall not**" statement. In compliance with SP 800-67rev1, NIST on July 11, 2017, placed on the Computer Security Division's Website a document that explained the rationale for this restriction on the number of the Triple-DES encryptions using the same key.

Then, in November 2017, NIST published SP 800-67rev2, which further tightened the restriction on the number of the Triple-DES encryptions with the same key. Per SP 800-67rev2, one key **shall not** be used to encrypt more than 2^{20} 64-bit data blocks. This version of the SP 800-67 standard was added to FIPS 140-2 Annex A in January 2018.

Question/Problem

How **shall** the aforementioned evolving requirement on the limit of the number of the Triple-DES encryptions with the same key be enforced? In particular, how can a user of a validated cryptographic module be confident that other modules are not performing the Triple-DES encryptions with the same key thus, possibly, exceeding the overall limit on the number of such encryptions?

Resolution

Each validated module **shall** have a limit of either 2^{20} or 2^{16} 64-bit data block encryptions with the same Triple-DES key.

The limit of 2^{20} encryptions with the same Triple-DES key applies when keys are generated as part of one of the recognized IETF protocols. To use this provision, the Security Policy **shall** say which of the IETF protocols governs the generation of the Triple-DES keys and list the IETF RFC(s) where the details of this protocol, relevant to the generation of the Triple-DES encryption keys, are documented. A proof of the implementation's compliance with the referenced protocol is not required.

If a key is not generated as part of a recognized IETF protocol (or, at least, no such claim is made by the vendor) then a further restriction on the number of encryptions with the same Triple-DES key is necessary, to avoid a "system-wide" violation of the overall limit on the number of encryptions with the same key. This limit is 2^{16} encryptions by each validated module.

For modules validated at Security Levels 1 or 2 in Section 4.1 of FIPS 140-2, the requirement limiting the number of encryptions with the same key may be enforced by policy. If this approach is taken, the module's Security Policy **shall** state that the user is responsible for ensuring the module's compliance with this requirement.

For modules validated at security levels 3 or 4 in Section 4.1 of FIPS 140-2, the module **shall** enforce this requirement. See an Additional Comment below for an example of how this *may* be done. The Security Policy **shall** explain how the module performs the enforcement.

Additional Comments

1. If an encryption key is generated as part of an IETF protocol implementation, there is a strong reason to believe (even though the module's compliance to the protocol is not tested by the CMVP) that the same key will not be used by any entity except for the two parties that are involved in the encryption session that required the generation of this key. Therefore, if each module performs no more than 2^{20} encryptions with the same key, then system-wide the number of the Triple-DES encryptions with that key will usually be limited to 2^{20} or, *in the worst possible case*, if the module's partner in this session can also perform the encryptions with the same key, to 2^{21}. While 2^{21} exceeds that stated limit, for the purposes of compliance with this IG, this solution is acceptable.

2. If an encryption key is not generated as part of a known protocol, then it is impossible to tell, in the general case, how many modules may use this encryption key. The limit on the number of same-key Triple-DES encryptions is set at 2^{16}. If the number of modules using this key for encryption is no greater than 16 then the overall limit of 2^{20} will not be exceeded. In the highly unusual scenario when more than 16 modules share the same key, it is likely that at least some of these modules will not perform the number of encryptions that is close to the allocated maximum (2^{16}). Even if they did and the total number of same-key encryptions exceeded 2^{20}, it would be difficult for the attacker to increase the chances of success when the encryptions are performed by the unrelated modules.

3. Here is an example of how the module may enforce this requirement when having Section 4.1 validated at Security Levels 3 or 4. The module will have a counter associated with each Triple-DES encryption key. When the counter reaches a certain value, the key can only be used for the decryption operations.

   An easier solution would be for the module to have only one counter that will be increased by one every time the module performs a Triple-DES encryption. When the counter reading reaches the prescribed threshold, the module blocks all Triple-DES keys stored in the module at that time from being used to perform encryption. Of course, in this case the new Triple-DES encryption keys would need to be generated more often.

   If the encryption counters are used in an implementation, then, in the case when the module's power is lost, the module may have a mechanism to restore the counters, or it may establish *all* new Triple-DES keys upon the restoration of power.

4. An earlier version of this Implementation Guidance contained a relaxed form of this requirement: the module was required to comply with the limit of 2^{32} Triple-DES encryptions established in SP 800-67*rev1*. This earlier version of the IG has established a transition period for modules being validated at Security Levels 3 and 4 in Section 4.1 of FIPS 140-2 to have this requirement enforced by the module. The new, tighter requirement of SP 800-67rev2 takes effect during this period, so here is an updated transition schedule. The effective date refers to the date of a CST laboratory's original submission of the module's test report.

   - Effective immediately, the Security Policy of the module **shall** state the limit (2^{20} or 2^{16}, as explained in this Implementation Guidance) on the number of the Triple-DES encryptions with the same key.

- If a vendor is claiming a Security Level 3 or 4 for Section 4.1 of FIPS 140-2, the module **shall** enforce the following limits on the number of the Triple-DES encryptions with the same key:
  - Effective May 10, 2018, the 2^{32} limit, if the key is used within an IETF protocol; the 2^{28} limit, otherwise.
  - Effective May 10, 2019, the 2^{20} limit, if the key is used within an IETF protocol; the 2^{16} limit, otherwise.

  The Security Policy **shall** explain the method of the module's enforcement of the appropriate limits on the number of the Triple-DES encryptions.

5. The provisions of this IG apply to the Triple-DES key wrapping the same way as to the data encryption.

6. The provisions of this IG apply only to the three-key Triple-DES encryption. The use of the two-key Triple-DES encryption for the protection of sensitive data is no longer allowed.

A.14 Approved Modulus Sizes for RSA Digital Signature and Other Approved Public Key Algorithms

Applicable Levels:	*All*
Original Publishing Date:	*08/07/2017*
Effective Date:	*08/07/2017*
Transition End Date:	
Last Modified Date:	
Relevant Assertions:	*AS.01.12*
Relevant Test Requirements:	*TE01.12.01-04*
Relevant Vendor Requirements:	

Background

The **FIPS 186-4** digital signature standard was published in July, 2013. This standard specifies three possible RSA modulus sizes for signature generation and verification: 1024, 2048 and 3072 bits. Because of the transition to the stronger algorithms and key sizes, as documented in **SP 800-131A Rev1**, the 1024-bit RSA modulus may now be used in the approved mode for signature verification, but not for signature generation.

Question/Problem

SP 800-131A Rev1 provides only the lower bound, 2048 bits, for the RSA modulus size used in signature generation. Does this imply that the RSA modulus sizes other than 2048 and 3072 may be used to generate the RSA signatures in the approved mode? In particular, is the use of the 4096-bit modulus approved and, if so, what are the testing requirements for the RSA key generation if the key pair used in the RSA signature algorithm is generated by the module?

Resolution

When performing an RSA signature generation, a module may use any modulus size greater than or equal to 2048 bits. At least one of the RSA modulus lengths supported by the module for RSA signature generation **shall** be 2048, 3072, or 4096 bits. The RSA signature algorithm implementations **shall** be tested by a CST lab for all implemented RSA modulus lengths where CAVS testing is available. If there is no CAVS testing for the generation of RSA keys of a particular size, then the requirement of IG 7.12 to obtain a CAVS certificate for the RSA key-generation algorithm does not apply to this key size.

Some of the RSA key generation methods described in Appendix B.3 of FIPS 186-4 rely on the use of the auxiliary primes p_1, p_2, q_1 and q_2 that must be generated before the module generates the RSA primes p and q. Table B.1 in FIPS 186-4 specifies, for RSA modulus lengths of 2048 and 3072 bits only, the minimum bit lengths and the maximum total length of the auxiliary primes. When implementing the RSA signature generation algorithm with other approved RSA modulus sizes, the vendor **shall** use the limitations from Table B.1 that apply to the longest RSA modulus shown in Table B.1 of FIPS 186-4 whose length does not exceed that of the implementation's RSA modulus. The minimum number of the Miller-Rabin tests used in primality testing **shall** be consistent (based on the entries in Tables C.2 and C.3 in FIPS 186-4) with the bit sizes of p, q, p_1, p_2, q_1 and q_2 taken from Table B.1, as described above. Hence, for the 4096-bit RSA modulus, the bit size of each auxiliary prime p_1, p_2, q_1 and q_2 is greater than 170 and, if the target prime-testing error probability is 2^{-100}, then the minimum number of the Miller-Rabin tests when generating and testing each of these primes is 27.

For example, when generating primes for the 4096-bit RSA modulus, the p and q primes **shall** be of 2048 bits each and the auxiliary primes **shall** be longer than 170 bits. The required minimum numbers of the Miller-Rabin tests are obtained from the bottom lines (those showing the 3072 modulus) of either Table C.2 or Table C.3 (for the target primality-testing error probability of 2^{-128} or 2^{-100}, respectively) of FIPS 186-4.

The use of the approved hash functions in digital signatures is documented in **SP 800-131A Rev1**, Table 9. The choice of a hash function may affect the security strength of the RSA signature algorithm.

Per IG 9.4, Note3, an RSA known-answer test for signature generation may be implemented for any approved RSA modulus size supported by the cryptographic module; that is, any implemented RSA modulus size of at least 2048 bits.

When performing an RSA signature verification, a module may use a 1024-bit modulus, in addition to each RSA modulus size approved for use in signature generation.

Additional Comments

1. This Implementation Guidance is concerned with a description of the approved modulus sizes (and, therefore, the approved key sizes) for the RSA digital signature algorithm only. For completeness, here is the status of the approved and allowed key sizes in other asymmetric-key-based algorithms and schemes. For the key agreement and key transport schemes, the status is expected to change upon the implementation of the transition announced in **SP 800-131A Rev1**.

    a. RSA-based key transport (encapsulation) schemes. The modulus sizes of 2048 and 3072 bits, are approved per **SP 800-56B**. All modulus sizes of 2048 bits and higher are allowed. It is strongly recommended that if an allowed scheme uses the auxiliary primes p1, p2, q1 and q2, the limits on the minimum sizes of these primes and on their maximum total size are the same as those stated in the main body of this IG for the auxiliary primes used in the RSA digital signature algorithm. The required numbers of the Miller-Rabin tests when testing the candidate primes are also the same as what is stated in the RSA digital signature case.

    b. DSA signatures. The approved bit sizes of the primes p and q used in the DSA algorithm are given in Section 4.2 of FIPS 186-4. Of these sizes, only the following pairs (2048, 224), (2048, 256) and (3072, 256) can be used in the approved mode for signature generation. For signature verification, these primes' sizes and the (1024, 160) pair are approved. There are no 'non-approved but allowed' DSA versions. While **SP 800-131A Rev1** says that any size of p no smaller than 2048 bits and any size of q no smaller than 224 bits are acceptable for DSA, no sizes other than those listed in FIPS 186-4 for DSA may be used.

    c. FFC-based key agreement schemes. Table 1 in **SP 800-56A** shows the following three sets of the sizes of the p and q primes used in the FFC-based key-agreement schemes: FA: (1024, 160), FB: (2048, 224) and FC: (2048, 256). Of the three, only FB and FC are currently approved, each resulting in the 112-bit strength in the established symmetric keys. In addition, any FFC-based scheme with the bit size of p no smaller than 2048 and the size of q no smaller than 224 is allowed in the approved mode.

d. ECDSA signatures. Digital signature generation and verification algorithms shown in FIPS 186-4 are approved, as long as the chosen elliptic curves are acceptable for their specific use. That is, all NIST-recommended curves are approved for signature verification. All but the following three NIST-recommended curves that provide less than 112 bits of encryption strength: P-192, B-163 and K-163, are also approved for signature generation. In addition, the use of any curve with the domain parameters generated in compliance with the rules specified in Section 6.1.1 of FIPS 186-4 is approved for signature verification. The use of these curves is also approved for signature generation, if the bit length of n, as shown in Table 1 in FIPS 186-4, is least 224. It is vendor's responsibility to demonstrate that a non-NIST-recommended curve meets these requirements.

e. There are no 'non-approved but allowed' ECDSA versions.

f. ECC-based key agreement schemes. Approved, if the scheme is compliant **SP 800-56A** or **SP 800-56A Rev2** (see IG D.1-rev2) and the elliptic curve used would meet the requirements for the ECDSA signature generation as shown above. Without the verified (through CAVS testing or vendor affirmation, per IG D.1-rev2) compliance to one of the revisions of **SP 800-56A** a scheme is allowed to be used in the approved mode if the vendor demonstrates (per **FIPS 186-4** or **ANSI X9.62**, as applicable) that the elliptic curve used in a key-agreement scheme and the associated domain parameters provide at least 112 bits of security.

g. RSA-based key agreement and DLC-based key transport schemes. These schemes are rarely used. There exist only the approved versions, no 'non-approved but allowed' ones. See SP **800-56BRev1** and **SP 800-56ARev2** for the detail explanations of these schemes and the security strength considerations.

2. When implementing a key agreement scheme (or a shared secret computation as part of a key agreement scheme), the vendor **shall** indicate in the module's Security Policy whether the scheme is of the Diffie-Hellman or the MQV variety. If a key agreement scheme (FFC or ECC-based) is documented on the module's certificate's non-approved line, the vendor is encouraged to state there if this is a Diffie-Hellman or an MQV scheme.

A.15 Vendor Affirmation for the SP 800-185 Algorithms

Applicable Levels:	*All*
Original Publishing Date:	*12/04/2017*
Effective Date:	*12/04/2017*
Last Modified Date:	*12/04/2017*
Relevant Assertions:	*AS.01.12*
Relevant Test Requirements:	*TE01.12.03, TE01.12.04*
Relevant Vendor Requirements:	*VE.01.12.03, VE01.12.04*

Background

SP 800-185 was published in December 2016. This standard includes specifications for several algorithms that are based on the SHAKE and KECCAK[c] constructions defined in **FIPS 202**. The functions are:

- cSHAKE128
- cSHAKE256
- KMAC128
- KMAC256

- KMACXOF128
- KMACXOF256
- TupleHash128
- TupleHash256
- TupleHashXOF128
- TupleHashXOF256
- ParallelHash128
- ParallelHash256
- ParallelHashXOF128 and
- ParallelHashXOF256.

Of these, cSHAKE128 and cSHAKE256 are the modifications of SHAKE128 and SHAKE256 defined in **FIPS 202.** These modifications allow the use of the customization strings, one of which is a function-name bit string with all of the function names defined by NIST. Another is the user-defined customization string (the "S" parameter in the **SP 800-185** notation.)

TupleHash and ParallelHash are the new hash function families based on the cSHAKE construction. The KMACs are the versions of the keyed message authentication mechanism also defined in terms of the cSHAKE use. The security strength of each algorithm (assuming that a KMAC key is sufficiently strong) is 128 or 256 bits, as indicated in the algorithm's name. The KMACs and all of the hash functions defined here have a variable output length. The output length parameter may either be built into the computational procedures itself (in the KMAC, TupleHash or ParallelHash functions without XOF in the name) or the computations may be independent of the desired output length but the output string would end once the desired length is reached (if XOF is present in the name of the function).

Question/Problem

To claim the vendor affirmation of the correct implementation of all or some of the functions defined in **SP 800-185**, which sections of the standard need to be addressed and what are the self-test and the documentation requirements?

Resolution

IG A.11 specifies the KECCAK family of sponge functions and defines the SHA-3 and SHAKE hash functions that are based on KECCAK. CAVS testing is available for the SHA-3 algorithms with the 224, 256, 384 and 512-bit output lengths and for the SHAKE128 and SHAKE256 hash functions. **IG A.11** further addresses the anticipated use of the SHA-3 by the existing high-level algorithms such as HMAC, RSA, etc.

SP 800-185, on the other hand, introduces the new families of algorithms, which are based on the cSHAKE functions also defined in **SP 800-185**. Some of these algorithms – the KMACs - can be viewed as "high-level", while the others are the new hash functions. The roles and the uses of these algorithms will likely be different and hence the algorithms will require the separate certificate annotation.

To claim the vendor affirmation to the **SP 800-185** standard for any subset of the cSHAKE, TupleHash and ParallelHash functions listed above, the vendor **shall**

1. Obtain a CAVS certificate for the applicable underlying SHAKE128 and/or SHAKE256 functions,
2. Verify that the cSHAKE functions are implemented in compliance with **SP 800-185**,
3. If the vendor affirmation is claimed for any of the TupleHash and ParallelHash functions, verify their implementations' compliance with **SP 800-185**,
4. If applicable, explain in the Security Policy the rules for setting the user-customized strings "S" in the hash functions for which the vendor affirmation is claimed,
5. Place the following entry into the module's validation certificate: **SHA-3-Customized (SHA-3 Cert. #nnn, vendor affirmed)**, and

6. Individually list in the Security Policy all algorithms named in the Background section of this Implementation Guidance for which the vendor affirmation is claimed. The vendor is responsible for making security claims commensurate with the choice of the **SP 800-185**-compliant hash functions (128 or 256 bits of security.)

To claim the vendor affirmation to the **SP 800-185** standard for any subset of the KMAC functions listed above, the vendor **shall**

1. Obtain a CAVS certificate for the underlying SHAKE128 and/or SHAKE256 functions,
2. Verify that the cSHAKE functions are implemented in compliance with **SP 800-185**,
3. Verify the KMAC implementations' compliance with **SP 800-185** for all versions of KMAC where the vendor affirmation is claimed,
4. If applicable, explain in the Security Policy the rules for setting the user-customized strings "S" in the hash functions for which the vendor affirmation is claimed,
5. Place the following entry into the module's validation certificate: **KMAC (SHA-3 Cert. #nnn, vendor affirmed)**, and
6. Individually list in the Security Policy all functions named in the Background section of this Implementation Guidance for which the vendor affirmation is claimed. The vendor is responsible for making security claims commensurate with the choice of KMACs. If the KMAC's key strength is smaller than the security strength indicated in the KMAC function's name (128 or 256), document in the Security Policy the reduction in the strength of this message authentication algorithm.

No self-tests are required to claim the vendor affirmation for the algorithms defined in **SP 800-185**. A self-test is required to the underlying SHAKE128 and/or SHAKE256 algorithms defined in **FIPS 202**, as their implementation and testing is a pre-requisite for the vendor affirmation of the **SP 800-185**-compliant algorithms and, therefore, the self-test requirements of **IG A.11** apply.

Additional Comments

1. At this time, there are no approved or allowed high-level algorithms using the **SP 800-185**-compliant functions.

FIPS 140-2 Annex B – *Approved Protection Profiles*

FIPS 140-2 Annex C – *Approved Random Number Generators*

C.1 moved to W.3

C.2 moved to W.4

FIPS 140-2 Annex D – *Approved Key Establishment Techniques*

D.1 moved to W.10

D.1-rev2 CAVP Requirements for Vendor Affirmation of SP 800-56A-rev2

Applicable Levels:	*All*
Original Publishing Date:	*07/08/2015*
Effective Date:	*07/08/2015*
Transition End Date:	
Transition End Date:	
Last Modified Date:	*07/08/2015*
Relevant Assertions:	*AS.01.12*
Relevant Test Requirements:	*TE01.12.01*
Relevant Vendor Requirements:	*VE.01.12.01*

Background

SP 800-56A was originally added to FIPS 140-2 Annex D on January 24, 2007. Its publication was followed by a release, on December 24, 2008, of a CAVS version containing tests for (most of the features of) the algorithms in that version of **SP 800-56A**. As of March 24, 2009, all newly submitted modules claiming to contain the **SP 800-56A**-compliant key agreement schemes were required to be tested.

The second revision of **SP 800-56A**, denoted **SP 800-56A-rev2**, was published in May 2013. There are several important differences between the two versions of **SP 800-56A**, the main one being that **SP 800-56A-rev2** incorporates the use of many additional key derivation functions (KDF's) into the key agreement schemes. These are the KDF's documented in **SP 800-135rev1** and **SP 800-56C**.

Question/Problem

Some vendors would like to claim vendor affirmation to **SP 800-56A-rev2** while others may continue testing their modules to the original version of **SP 800-56A** and receiving the KAS certificates not yet available for the modules' compliant with **SP 800-56A-rev2**. How does this co-existence of two different versions of **SP 800-56A** along with two different methods of claiming the module's compliance get administered by the CMVP?

Further, to claim *vendor affirmation* to **SP 800-56A-rev2**, what sections of the publication need to be addressed?

Resolution

Vendors may continue testing their modules' implementations of key agreement schemes to the original version of **SP 800-56A**, for which a CAVS test is currently available. While it is possible that an implementation compliant with the first release of **SP 800-56A** will also be compliant with **SP 800-56A-rev2**, no testing to **SP 800-56A-rev2** is available and therefore no claims can be made of the tested compliance to **SP**

800-56A-rev2. The KAS certificates showing compliance with the original version of **SP 800-56A** will continue to be issued until a further notice.

To claim *vendor affirmation* to **SP 800-56A-rev2**, information contained in the following sections in that standard that are supported by the implementation under test (IUT) **shall** be implemented:

Section 5.6.2.3.1	Finite Field Cryptography (FFC) Full Public Key Validation Routine (if FFC is implemented)
Section 5.6.2.3.2	Elliptic Curve Cryptography (ECC) Full Public Key Validation Routine (if ECC is implemented)
Section 5.7	DLC Primitives (the Diffie-Hellman or the various ECC MQV primitives)
Section 5.8	Key Derivation Functions for Key Agreement Schemes (any function from the newly expanded list of key derivation functions)
Section 5.9	Key Conformation, if the implementation supports it (see also the scheme-specific key confirmation information in various parts of **Section 6**)
Section 6	Key Agreement (any of the schemes presented in this section)

Note the change in section numbering from the original publication of **SP 800-56A**.

Additional Comments

1. The requirements specified in **SP 800-56A-rev2** depend on several NIST approved security functions. To claim vendor affirmation to **SP 800-56A-rev2**, the underlying security functions used by an IUT **shall** be tested and validated prior to claiming vendor affirmation. These include:

   - Approved hash algorithms (SHA1, SHA224, SHA256, SHA384, and/or SHA512)
   - Approved Message Authentication Code (MAC) algorithms (CMAC, CCM, GMAC and/or HMAC)
   - Approved Random Number Generators (RNG and DRBG)
   - If FFC is supported,
     - If the IUT generates domain parameters, the DSA PQG generation and/or verification tests from FIPS 186-4.
     - If the IUT generates key pairs, the DSA key pair generation tests from FIPS 186-4.
   - If ECC is supported,
     - If the IUT generates key pairs, the ECDSA key pair generation test and/or the Public Key Validation (PKV) test from FIPS 186-4.

2. **SP 800-56A-rev2** self-tests required in cryptographic module implementations must consist of the known answer tests that validate the correctness of the implemented DLC primitives and the known answer tests for all key derivation functions implemented in the key agreement schemes. See Implementation Guidance 9.6 for details.

3. There is no guidance provided to claim vendor affirmation to the DLC-based Key Transport schemes from **SP 800-56A-rev2**.

4. To claim vendor affirmation with **SP 800-56A-rev2**, the implementation **shall** comply with the requirements of **SP 800-131A**. That is, the key lengths and the Mac Tag and Mac Key lengths (if key confirmation is supported) must be appropriate to guarantee at least the 112-bit security strength. Thus the FA and EA domain parameters **shall** not be used in an approved mode. See Tables 8 and 9 in **SP 800-56A-rev2** for the minimum lengths of the key confirmation parameters.

5. Once CAVS testing to **SP 800-56A-rev2** becomes available, and after passing of a suitable transition period, there will be no need to issue any more KAS certificates with testing performed to the original version of **SP 800-56A**.

Furthermore, there will be no more CVL certificates issued to show that the shared secret computation was performed as required in **SP 800-56A** (whether the original version of the standard or **SP 800-56A-rev2**.) The reason being is that with the addition of all of the key derivation functions from **SP 800-135rev1** and **SP 800-56C** to **SP 800-56A-rev2**, there is no good reason to implement the shared secret computation as in **SP 800-56A-rev2**, but to not implement at least one of the **SP 800-56A-rev2** key agreement schemes in its entirety. Therefore, in order to receive the credit for the **SP 800-56A-rev2**-compliant shared secret computation the vendor will have to obtain a KAS certificate.

Until testing to **SP 800-56A-rev2** is available, vendors may claim vendor affirmation to **SP 800-56A-rev2** along with the applicable key derivation function CVL certificates.

Annotation

Refer to IG G.13 for annotation examples.

Derived Test Requirements

Upon the following successful review, the CST Lab **shall** affirm by annotating the algorithm entry per the IG G.13 annotation requirements.

Required Vendor Information

The vendor **shall** provide evidence that their module implements the sections outlined above completely and accurately. This **shall** be accomplished by documentation and code review.

Required Test Procedures

The tester **shall** review the vendor's evidence demonstrating that their implementation conforms to the specifications specified above. This **shall** be accomplished by documentation and code review. The tester **shall** verify the rationale provided by the vendor.

D.2 Acceptable Key Establishment Protocols

Applicable Levels:	*All*
Original Publishing Date:	*02/10/2004*
Effective Date:	*02/10/2004*
Last Modified Date:	*08/07/2017*
Relevant Assertions:	*AS.07.21*
Relevant Test Requirements:	*TE07.21.01*
Relevant Vendor Requirements:	*VE.07.21.01-02*

Background

Cryptographic modules may use various methods for establishing keys within a cryptographic module. These methods include the use of symmetric and asymmetric key establishment schemes within protocols to establish and maintain secure communication links between modules. FIPS 140-2 Annex D provides a list of approved key establishment techniques for establishing keying material that are applicable to FIPS 140-2.

Question/Problem

What are all the types of key establishment within a cryptographic module, and what are the approved and allowed methods for each type that may be used in the approved mode of operation?

Resolution

Key establishment is the process by which secret keying material is securely established either within the module or between two or more entities. This IG lists all types of methods for key establishment that may be performed in an approved mode of operation. The specifics of each type of key establishment are addressed in the corresponding IGs that this IG references. Therefore, this IG serves as an umbrella IG for the approved and allowed key establishment methods.

The following are the six types of methods that may be used in the approved mode for the establishment of keys within a cryptographic module.

Key agreement is a method of electronic key establishment where the resulting keying material is a function of information contributed by two or more participants, so that no party can predetermine the value of the secret keying material independently from the contribution of any other party. Key agreement is performed using key agreement schemes. The approved schemes for key agreement that may be implemented within a cryptographic module are referenced in Annex D of FIPS 140-2 and further discussed in IG D.8, which also lists the Allowed key agreement schemes.

Key transport is a method of electronic key establishment whereby one party (the sender) selects a value for the secret keying material and then securely distributes that value to another party (the receiver). Key transport is performed using key transport schemes. The approved schemes for key transport that may be implemented within a cryptographic module are referenced in Annex D of FIPS 140-2 and further discussed in IG D.9, which also lists the allowed key transport schemes.

Key generation is the process for generating cryptographic keys within a particular cryptographic module. The approved methods for key generation are listed in IG 7.8.

Key entry is a method for key establishment where the key is manually or electronically entered into the module. It does not include the key transport schemes described earlier in this IG. IG 7.7 provides further information about mapping key entry and output states to the FIPS 140-2 requirements.

Key derivation is a method for deriving keys from the certain parameters using the approved key derivation functions. One possibility is to derive a key from an already existing related key as described in **SP 800-108**. Another is to derive a key for storage applications only, in compliance with **SP 800-132**.

Pre-loading of a key is a method by which a manufacturer of the module can establish a key within the module. A key pre-loaded by the manufacturer is available when the module is first powered-on.

Additional Comments

This IG does not address key establishment for use in authentication techniques.

The key establishment method(s) that involve key agreement or key transport used by the cryptographic module **shall** be listed under **AS.07.21**.

While some IGs referenced from this IG list various Key Agreement and Key Transport methods as either approved or allowed, it is important to keep in mind that the strength of these methods may be weaker than the strength of the transported or agreed-upon key. In this case, the resulting strength of the key should be properly documented. See IG 7.5 for ways to calculate the strength of the established key, and IG G.13 for the proper way to caveat the possible loss of the established key's cryptographic strength in the module's certificate.

D.3 Assurance of the Validity of a Public Key for Key Establishment

Applicable Levels:	*All*
Original Publishing Date:	*10/21/2009*
Effective Date:	*10/21/2009*
Last Modified Date:	*10/21/2009*
Relevant Assertions:	*AS.07.17*
Relevant Test Requirements:	*TE07.17.01-02*
Relevant Vendor Requirements:	*VE.07.17.01*

Background

The correct functioning of public key algorithms depends, in part, on the arithmetic validity of the public key.

Both the owner and the recipient of a public key need to obtain assurance of public key validity before using the key for operational purposes after key establishment. Public key algorithms for key establishment are specified in **SP 800-56A** and **SP 800-56B**. Methods for obtaining assurance of public key validity are provided in Section 5.6.2 of **SP 800-56A**, and in Section 6.4 of **SP 800-56B**.

The key establishment schemes in **SP 800-56A** are specified using either static (long term, multi-use) keys or ephemeral (short term, single use) keys or both. The keys used in the **SP 800-56B** schemes are generally long term (i.e., static) keys.

Since a static key is normally used for a relatively long period of time, and a number of methods are provided for obtaining assurance of public key validity either by the owner or recipient directly, or by using a trusted third party, the process of obtaining the assurance is not too onerous. However, methods for obtaining this assurance for ephemeral keys are more limited, since a trusted third party is normally not available for obtaining the required assurance. The owner of an ephemeral public key generates that key, and obtains assurance of ephemeral public key validity by virtue of generating the key as specified in **SP 800-56**A (see Section 5.6.2.1; Note that this section applies to the owner assurances of both Static and Ephemeral public key validity). However, the recipient of an ephemeral public key must obtain the assurance by performing an explicit public key validation process.

Question/Problem

Public key validation requires a certain amount of time to perform, which can significantly affect communication performance. Can this process be omitted if at least some of the security goals (i.e., authentication of the public key owner and the integrity of the ephemeral key) are fulfilled by other means?

Resolution

The owner or a recipient of a static public key **shall** obtain assurance of the validity of that public key using one or more of the methods specified in **SP 800-56A** or **SP 800-56B**, as appropriate. The owner of an ephemeral public key **shall** obtain assurance of the validity of that key as specified in **SP 800-56A**. Explicit public key validation of an ephemeral public key is required as specified in **SP 800-56A** by a recipient, except in the following situation; in this case, explicit public key validation of the ephemeral public key by the recipient is optional:

1. The ephemeral public key was generated for use in an FFC dhEphem key agreement scheme or an ECC Ephemeral Unified Model key agreement scheme, and

2. The key agreement scheme is being conducted using a protocol that authenticates the source and the integrity of each received ephemeral public key by means of an approved security technique (e.g., a digital signature or an HMAC).

Protocols that satisfy #2 above and, therefore, may omit the explicit ephemeral public key validation process include:

- Internet Key Exchange (IKE) protocol,

- Internet Key Exchange protocol, version 2 (IKEv2),
- Transport Layer Security (TLS) protocol, versions 1.0, and
- Datagram Transport Layer Security (DTLS) protocol, version 1.0.

In this case, when explicit public key validation is not performed on the ephemeral public key by an implementation in the manner specified in **SP 800-56A** (and therefore is not tested by the CAVS), the cryptographic algorithm's validation will indicate that the capability to provide assurance of ephemeral public key validity is not required for algorithm validation, based on this IG. However, the cryptographic algorithm validation and the cryptographic module validation may still claim that the algorithm and module are otherwise compliant with **SP 800-56A**.

Additional Comments

CAVP

Example of the Description/Notes field of a **SP800-56A** algorithm validation entry where the explicit public key validation of an ephemeral public key is not required for algorithm validation based on this IG (and therefore is not tested by the CAVS):

ECC: (ASSURANCES <5.5.2 #3>

ASSURANCE 5.6.2.3: requirement is not required for algorithm validation, based on IG 7.10)
SCHEMES [EphemeralUnified (KARole(s): Responder)
(EC: P-256 SHA256)]
SHS Val#650 DRBG Val#1

CMVP

If a cryptographic module includes a key agreement scheme whereby the recipient of an ephemeral public key omits the explicit public key validation, the modules Security Policy **shall** indicate the appropriate protocol listed above that allows the omission of the validation in order to claim conformance to this IG.

D.4 Requirements for Vendor Affirmation of SP 800-56B

Applicable Levels:	*All*
Original Publishing Date:	*07/15/2011*
Effective Date:	*07/15/2011*
Transition End Date:	
Last Modified Date:	*05/10/2016*
Relevant Assertions:	*AS.07.17*
Relevant Test Requirements:	*TE07.17.01 and TE07.17.02*
Relevant Vendor Requirements:	*VE.07.17.01*

Background

SP 800-56B was published August 2009 and added to FIPS 140-2 Annex D on October 08, 2009. Until CAVP testing for SP **800-56B** is available, IG A.3 is applicable. **SP 800-56B** includes information beyond the specifications of the key establishment algorithm itself; i.e. instructions to the implementer to aid in the implementation of the algorithm.

Question/Problem

To claim *vendor affirmation* to **SP 800-56B**, what sections of the publication need to be addressed?

Resolution

To claim *vendor affirmation* to **SP 800-56B**, the vendor first needs to specify what parts of **SP 800-56B** are supported by the subject implementation. These parts include some (at least one) or all of the following: RSA key pair generation, public key validation, a key agreement scheme, and a key transport scheme.

Information contained in the following sections that are supported by the implementation under test (IUT) **shall** be implemented:

Section 6.3	Either **Section 6.3.1** or **Section 6.3.2** or both (if RSA key pair generation is claimed). This further requires the implementation of information contained in **Section 5.4** (Prime Number Generators).
Section 6.4	Either **Section 6.4.1** or **Section 6.4.2** or both (if public key validation is claimed)
Section 8	If a key agreement scheme is claimed
Section 9	If a key transport key is claimed
Section 5.9	Approved key derivation functions. This section applies if a vendor affirmation of either a key agreement or a key transport scheme is claimed

Annotation

Refer to IG G.13 for annotation examples (**KAS** or **KTS**).

Derived Test Requirements

Upon the following successful review, the CST Lab **shall** affirm by annotating the algorithm entry per the IG G.13 annotation requirements.

Required Vendor Information

The vendor **shall** provide evidence that their implementation implements the sections outlined above completely and accurately. This **shall** be accomplished by documentation and code review.

Required Test Procedures

The tester **shall** review the vendor's evidence demonstrating that their implementation conforms to the specifications specified above. This **shall** be accomplished by documentation and code review. The tester **shall** verify the rationale provided by the vendor.

Additional Comments

1. The components in **SP 800-56B** **shall** only be used within the **SP 800-56B** protocol.

2. The requirements specified in **SP 800-56B** depend on several NIST approved security functions, such as, SHA, DRBG, and the FIPS 186-4 key pair generation for RSA. While validation testing for **SP 800-56B** concentrates largely on testing the algorithm unique to **SP 800-56B**, other supporting security functions may not be thoroughly tested by the testing in **SP 800-56B**, when such testing becomes available. The validation tests for these supporting security functions are found in the validation test suite for this specific function. Therefore, these supporting security functions **shall** be validated as a prerequisite to **SP 800-56B** vendor affirmation.

   To claim vendor affirmation to **SP 800-56B**, the underlying security functions used by this IUT **shall** be tested and validated prior to claiming vendor affirmation. These include:

   - Hash algorithms as applicable
   - If vendor affirmation is claimed for RSA key pair generation (Section 6.3), or a key agreement scheme (Section 8) or a key transport scheme (Section 9)
     - Supported Random Bit Generators (DRBG)
   - If vendor affirmation is claimed for RSA key pair generation (Section 6.3)
     - An RSA key pair generation algorithm in FIPS 186-4

3. The **SP 800-56B** Self-Tests:

    The RSA algorithms used in the key wrapping and key agreement schemes described in **SP 800-56B** require the known-answer tests. The module **shall** have an RSA encryption pre-computed and then, while performing a power-up self-test, the module **shall** perform the RSA encryption again and compare the newly-generated result to the pre-computed value.

    The module **shall** also have a separate known answer for the RSA decryption by starting with a given value representing an RSA encryption (which could be either pre-computed or generated during a power-up test described earlier in this paragraph) and decrypting this value using the RSA algorithm. The result of said decryption operation is compared to a pre-computed result. If the module performs only one of the RSA encryption operations, say, either the wrapping or the unwrapping of a cryptographic key, then only the self-test that is attributable to this operation is required. If a key-agreement scheme from **SP 800-56B** is implemented then again only the RSA operations required by that scheme need to be tested using a known answer test.

    While it may appear that the requirements for the RSA encryption and the RSA decryption (corresponding to the key wrapping and key unwrapping schemes) known answer tests are identical, they are not. The RSA encryption known answer test consists of checking the value of $M^e (\mathrm{mod}\ N)$, while the RSA decryption known answer test consists of checking the value of $M^d (\mathrm{mod}\ N)$, where (e, N) is the RSA public key with *e* taking one of the values allowed in **SP 800-56B**, and *d* is the RSA private key consistent with the public key (e, N). The *e* and *d* values **shall** be valid public and private key exponents, correspondingly.

D.5 moved to [W.11](W.11)

D.6 Requirements for Vendor Affirmation of SP 800-132

Applicable Levels:	*All*
Original Publishing Date:	*07/15/2011*
Effective Date:	*07/15/2011*
Transition End Date:	
Last Modified Date:	*04/23/2012*
Relevant Assertions:	*AS.07.11 and AS.07.16*
Relevant Test Requirements:	*TE07.11.01-02 and TE07.16.01-02*
Relevant Vendor Requirements:	*VE.07.11.01 and VE.07.16.01*

Background

SP 800-132 was published December 2010 and added to FIPS 140-2 Annex D on January 04, 2011. Until CAVP testing for **SP 800-132** is available, [IG A.3](IG A.3) is applicable. This Special Publication defines the methods and the applicability for password-based key derivation.

Question/Problem

To claim *vendor affirmation* to **SP 800-132**, what sections of the publication need to be addressed and what are the applicable documentation requirements?

Resolution

The entire text of **SP 800-132** is applicable. In Section 5.4 of that Special Publication, two options are given for deriving a Data Protection Key from the Master Key. The vendor **shall** specify in the cryptographic module's Security Policy which option is used by the module. If the module is designed to support both options, then this **shall** be stated in the Security Policy.

The strength of the Data Protection Key is based on the strength of the Password and/or Passphrase used in key derivation. **SP 800-132** does not impose any strictly defined requirements on the strength of a password. It says that "passwords **should** be strong enough so that it is infeasible for attackers to get access by guessing a password." Therefore, the vendor **shall** document in the module's Security Policy the length of a password/passphrase used in key derivation and establish an upper bound for the probability of having this parameter guessed at random. This probability **shall** take into account not only the length of the password/passphrase, but also the difficulty of guessing it. The decision on the minimum length of a password used for key derivation is the vendor's, but the vendor **shall** at a minimum informally justify the decision.

The vendor **shall** also document the acceptable values of other parameters used in key derivation, see Section 5.3 of **SP 800-132**.

Further, the vendor **shall** indicate in the module's Security Policy that keys derived from passwords, as shown in **SP 800-132**, may only be used in storage applications.

The vendor **shall** comply with all "**shall**" statements in **SP 800-132**. These refer to but are not limited to the length of the salt parameter and the use of the approved hash functions, encryption algorithms and random number generators.

Annotation

Refer to IG G.13 for annotation examples (**PBKDF**).

Derived Test Requirements

Upon the following successful review, the CST Lab **shall** affirm by annotating the algorithm entry per the IG G.13 annotation requirements.

Required Vendor Information

The vendor **shall** provide evidence that their implementation complies with the requirements of **SP 800-132** and of the documentation requirements of this Implementation Guidance. This **shall** be accomplished by documentation and code review.

Required Test Procedures

The tester **shall** review the vendor's evidence demonstrating that their implementation conforms to the specifications specified above. This **shall** be accomplished by documentation and code review.

Additional Comments

1. While the wording in IG D.9 specifically prohibits using password-based key establishment methods in FIPS mode, this does not contradict the statements in **SP 800-132** and in this IG, since **SP 800-132** allows the derived keys to be used only for storage applications. The key establishment addressed in IG D.9 shows how to establish a key used for protecting sensitive data that may leave the cryptographic module.

2. No self-tests are required to claim *vendor affirmation* to **SP 800-132**.

D.7 moved to W.12

D.8 Key Agreement Methods

Applicable Levels:	*All*
Original Publishing Date:	*04/23/2012*
Effective Date:	
Last Modified Date:	*05/10/2017*
Relevant Assertions:	*AS07.21*
Relevant Test Requirements:	*TE07.21.01*
Relevant Vendor Requirements:	*VE07.21.01-02*

Background

Cryptographic modules may implement various key establishment schemes to establish and maintain secure communication links between modules. Key establishment includes the processes by which secret keying material is securely established between two or more entities. Keying material is data that is necessary to establish and maintain a cryptographic keying relationship. These schemes are classified into key agreement schemes and key transport. Key transport is addressed in IG D.9; this IG addresses key agreement.

Key agreement is a method of key establishment where the resulting keying material is a function of information contributed by two or more participants, so that no party can predetermine the value of the secret keying material independently from the contribution of any other party. Key agreement is performed using key agreement schemes.

Question/Problem

What are the approved and allowed key agreement techniques that can be used in an approved mode of operation?

Resolution

There are currently *six scenarios* for the full or partial **key agreement schemes** that are allowed in a FIPS approved mode of operation. Of the following six scenarios, the first four scenarios apply when a key is established (i.e. key agreement) and last two scenarios when only the primitive is implemented (e.g. in a software toolkit):

1. CAVP KAS Certificate

2. approved **SP 800-56B**-compliant RSA-based key agreement scheme

3. non-approved but *allowed* per this IG (a primitive as defined in **SP 800-56A** with a KDF specified in this IG or a SP)

4. non-approved but *allowed* legacy implementation

5. approved (compliant with **SP 800-56A)** primitive only

6. non-approved (not compliant with **SP 800-56A**) primitive only

NIST has specified key agreement schemes in **SP 800-56A** using Discrete Logarithm Cryptography (DLC) and in **SP 800-56B** using Integer Factorization Cryptography (RSA). *The latter is a new technology and at this time it is only allowed in the approved mode of operation if fully compliant with the key agreement provisions of SP 800-56B.* The remaining part of this IG deals with the details of the DLC-based key agreement schemes.

Scenario 1 requires compliance with **SP 800-56A**, as demonstrated by a **KAS** certificate. Each scheme in SP **800-56A** consists of several elements:

- A primitive (i.e., an algorithm) that is used to generate a shared secret from the public and/or private keys of the initiator and responder in a key agreement transaction. The shared secret is an intermediate value that is used as input to a key derivation function.

- A key derivation function (KDF) that uses the shared secret and other information to derive keying material.

- An optional message authentication code (MAC) that is used for key confirmation or implementation validation. Key confirmation is a procedure that provides assurance to one party (the key confirmation recipient) that another party (the key confirmation provider) actually possesses the correct secret keying material and/or shared secret.

- The rules for using the scheme securely. The rules specified in **SP 800-56A** include criteria for generating the domain parameters and asymmetric key pairs used during key agreement, methods for obtaining the required assurances, and specifications for performing key confirmation.

Scenario 2 is addressed in **SP 800-56B**.

Scenario 3 addresses any implementations of DLC key agreement schemes which do not completely conform to **SP 800-56A** but claim conformance to **SP 800-56A** primitives. The module's DLC key agreement scheme **shall** include:

One or more of the primitives specified in **SP 800-56A**. Domain parameters and key sizes **shall** conform to **SP 800-56A**. A **CVL** algorithm validation certificate for a DLC primitive is required, and the KDFs **shall** conform to any of the following:

- One of the key derivation methods shown in **SP 800-56C**, or

- The KDFs specified in **SP 800-135rev1**. Each KDF **shall** have a **CVL** certificate and may be used only in a protocol where it is specifically allowed.

If key confirmation is claimed for a key agreement scheme, one or more of the key confirmation methods in **SP 800-56A shall** be used.

An implementation **shall** conform to the key agreement rules specified in **SP 800-56A**, with the possible exception of the format of the KDF (see above).

Scenario 4 applies when the module implements a key agreement scheme which is not compliant with either the **SP 800-56A** primitives or the KDF standards listed in Scenario 3 or both. Such implementations are allowed in the approved mode subject to the requirements of IG D.11.

Scenario 5 requires compliance with one or more of the key agreement primitives specified in **SP 800-56A**. Domain parameters and key sizes **shall** conform to **SP 800-56A**. A **CVL** algorithm validation certificate for a DLC primitive is required.

Scenario 6 is self-explanatory.

Note that all implementations are subject to the transition rules of **SP 800-131A**. After the transition period, the use of the non-compliant algorithms **shall not** be allowed in the approved mode.

Additional Comments

This IG does not address key establishment techniques other than those used for key agreement.

The key establishment method(s) used by the cryptographic module **shall** be listed under **AS.07.21**.

FIPS 140-2 certificate annotation examples for the key agreement schemes can be found in IG G.13.

D.9 Key Transport Methods

Applicable Levels:	*All*
Original Publishing Date:	*04/23/2012*
Effective Date:	*07/08/2015*
Last Modified Date:	*01/19/2018*
Relevant Assertions:	*AS07.21*
Relevant Test Requirements:	*TE07.21.01*
Relevant Vendor Requirements:	*VE07.21.01-02*

Background

Cryptographic modules may implement various key establishment schemes to establish and maintain secure communication links between modules. Key establishment includes the processes by which secret keying material is securely established between two or more entities. Keying material is data that is necessary to establish and maintain a cryptographic keying relationship[1]. These schemes are classified into key agreement schemes and key transport schemes. Key agreement is addressed in IG D.8; this IG addresses key transport.

Key transport is a method of key establishment whereby one party (the sender) selects a value for the secret keying material and then securely distributes that value to another party (the receiver). Key transport can be provided using either symmetric or asymmetric techniques.

Question/Problem

What are the approved and allowed key transport techniques that can be used in an approved mode of operation?

Resolution

Symmetric and asymmetric algorithms are used to provide confidentiality and integrity protection of the keying material to be transported. Key transport includes some means of key encapsulation or key wrapping for the keying material to be transported. Key transport **shall** be performed using the appropriate key lengths classified as acceptable, deprecated or legacy-use as specified in SP 800-131Arev1.

> **Key Encapsulation** is a class of techniques whereby keying material is encrypted using asymmetric (public key) algorithms; integrity protection is also commonly provided. The amount of keying material is usually limited by the practicality of performing the encryption operation. The key used for key encapsulation is called a key encapsulation key, which is a public key for which the associated private key is known by the receiver.

> **Key Wrapping** is a class of techniques whereby keying material is encrypted using symmetric algorithms; integrity protection is also commonly provided. The key used by the key wrapping algorithm to wrap the key to be transported is called a key wrapping key, which is a key that must be known by both the sender and the receiver.

Approved methods for key transport.

- Employing an approved RSA-based key transport scheme, as specified in SP 800-56B.

- Employing a key wrapping key, shared by the sender and receiver, together with an approved symmetric key-wrapping algorithm to wrap the keying material to be transported. Approved key wrapping algorithms are specified in SP 800-38F. One method is to use AES in either the KW or KWP mode, or Triple-DES in the TKW mode. Another is to use a previously approved authenticated symmetric encryption mode, such as, AES GCM, for key wrapping. Yet another approved key-wrapping technique is a "combination" method: use any approved symmetric encryption mode, such

---

[1] The state existing between two entities when they share at least one cryptographic key.

as AES ECB, AES CBC, Triple-DES TECB, etc. together with an approved authentication method (for example, HMAC or AES CMAC). The entire wrapped message **shall** be authenticated.

The symmetric key encryption algorithm, and, if applicable, the authentication algorithm, used for key wrapping **shall** be tested and validated by the CAVP, and the algorithms' certificate numbers **shall** be shown on the module's certificate. If the security strength of the key wrapping algorithm and the wrapping algorithm's key can be lower than that of the (potential) security strength of the wrapped key, then the resulting security strength of the wrapped key is the security strength of the key wrapping key and algorithm, and **shall** be shown on the module's certificate in accordance with IG G.13.

Allowed methods for key transport.

The SP 800-131A transition document, published in November 2015, has stipulated that the non-approved key encapsulation and key wrapping would be disallowed after December 31, 2017. However, since an updated version of SP 800-56B has not been published in time for this transition, the use of the non-SP-800-56B-compliant methods for key encapsulation continues to be allowed (as long as the RSA keys are at least 2048 bit long), until a new transition date for this method of key transport is announced.

The transition to the mandatory use of the SP 800-38F-compliant methods for key wrapping has taken place on December 31, 2017, as it has been announced in the November 2015 version of SP 800-131A. Key wrapping is not allowed, if the algorithm does not meet the requirements of SP 800-38F.

The following key un-encapsulation/unwrapping methods for key transport are allowed, for use in the approved mode.

- Any RSA-based key decryption algorithm that uses an RSA modulus that is at least 2048 bits long.

- A key unwrapping using any approved mode of AES or two-key or three-key Triple-DES.

The use of these algorithms by the module for key transport **shall** be annotated on the certificate's allowed algorithm line as shown in IG G.13.

Additional Comments

1. This IG does not address key establishment mechanisms other than those used for key transport.

2. The key transport method(s) used by the cryptographic module **shall** be listed under **AS.07.21**.

3. The module's compliance with the approved key transport techniques **shall** be annotated on the validation certificate's FIPS approved Cryptographic Algorithms line as KTS.

4. As the SP-800-38F-compliant schemes are comprised of the algorithms that have been tested and issued the CAVS validation certificates, no vendor affirmation of this key transport scheme in the module's validation certificate is required. See IG G.13 for details.

   It is the tester's responsibility to verify that when using the "combination" method described above the entire message gets authenticated.

5. As of January, 2018, there is no CAVP algorithm test for a full SP-800-56B-compliant key transport scheme; until such test exists, this compliance will be indicated in the module's validation certificate as vendor affirmed for KTS.

6. SP 800-131A is in the process of being revised. This revision will address the new transition guidance and the new transition dates for the non-approved key encapsulation methods in light of the upcoming publication of SP 800-56B revision 2.

7. The key wrapping used in many industry protocols, such as TLS and SSH are likely to be compliant with one of the provisions of SP 800-38F. If a module implements such a protocol then the key transport **shall** be documented as a KTS.

D.10 Requirements for Vendor Affirmation of SP 800-56C

Applicable Levels:	*All*
Original Publishing Date:	*04/23/2012*
Effective Date:	*11/30/2011*
Transition End Date:	
Last Modified Date:	*04/17/2012*
Relevant Assertions:	*AS07.11 and AS07.16*
Relevant Test Requirements:	*TE07.11.01-02 and TE07.16.01-02*
Relevant Vendor Requirements:	*VE07.11.01 and VE07.16.01*

Background

SP 800-56C was published November 2011 and added to FIPS 140-2 Annex D on December 20, 2011. Until CAVP testing for **SP 800-56C** is available, IG A.3 is applicable. This Special Publication defines the methods and the applicability for password-based key derivation.

Question/Problem

To claim *vendor affirmation* to **SP 800-56C**, what sections of the publication need to be addressed and what are the applicable documentation requirements?

Resolution

The entire text of **SP 800-56C** is applicable. The Randomness Extraction **shall** be performed as shown in Section 5 of **SP 800-56C** and this step **shall** be followed by Key Expansion using the key derivation functions defined in **SP 800-108**, as it is shown in Section 6 of **SP 800-56C**.

The vendor **shall** comply with all "**shall**" statements in **SP 800-56C**.

Annotation

Refer to IG G.13 for annotation examples.

Derived Test Requirements

Upon the following successful review, the CST Lab **shall** affirm by annotating the algorithm entry per the IG G.13 annotation requirements.

Required Vendor Information

The vendor **shall** provide evidence that their implementation complies with the requirements of **SP 800-56C** and of the documentation requirements of this Implementation Guidance. This **shall** be accomplished by documentation and code review.

Required Test Procedures

The tester **shall** review the vendor's evidence demonstrating that their implementation conforms to the specifications specified above. This **shall** be accomplished by documentation and code review.

The tester **shall** verify that the vendor chose a MAC function appropriately for the targeted security strength, as shown in Tables 1, 2, and 3 of **SP 800-56C**.

Additional Comments

No self-tests are required to claim *vendor affirmation* to **SP 800-56C**.

D.11 References to the Support of Industry Protocols

Applicable Levels:	*All*
Original Publishing Date:	*07/25/2013*
Effective Date:	
Last Modified Date:	*05/10/2017*
Relevant Assertions:	
Relevant Test Requirements:	
Relevant Vendor Requirements:	

Background

The cryptographic modules may implement various protocols known in the security industry. The examples of such protocols are IKE, TLS, SSH, SRTP, SNMP and TPM, listed in **SP 800-135rev1**. These protocols usually include a complete or partial key establishment scheme and, sometimes, an encrypted session that uses the newly-established key to protect sensitive data.

In the past, the Security Policy may have made references to modules' support of such protocols. The CMVP did not provide guidance to how and under what conditions the protocol references should be documented. Therefore, the supporting claims may be inconsistent from module to module and may not reflect the relevance of these protocols to the algorithms approved for or allowed for use in the FIPS 140-2 approved mode of operations. Nor did these claims reflect the level of the CAVP algorithm testing performed. In some cases, a test was available for a portion of the protocol, such as a key derivation function (KDF). However, the testing was not performed and the module's support for the corresponding protocol in the approved mode would still be claimed. In other cases, the generic description of a protocol contained several distinct key establishment schemes, such as in the TLS v1.1 protocol that could utilize either the RSA key transport (key encapsulation) scheme or the Diffie-Hellman key agreement scheme to establish a secret value known to the two parties in the protocol. By simply claiming the module's protocol support it may not be clear which components are compliant to FIPS 140-2.

Question/Problem

What are the module documentation requirements to show support for the protocols which have their key derivation functions listed in **SP 800-135rev1**?

Resolution

FIPS 140-2 and its Annexes do not address protocols. Only the cryptographic *algorithms* (such as, for example, AES or DSA) and *schemes* (such as the key agreement schemes from **SP 800-56A** or the RSA-based key encapsulation schemes) that are approved and allowed may be used in the approved mode of operations. These algorithms and schemes are referenced in the FIPS 140-2 Annexes.

The protocols' KDFs described in **SP 800-135rev1** are well-defined and are viewed as algorithms, not protocols within the scope of a FIPS 140-2 validation. The CAVP testing for such KDFs is available. The testing laboratories **shall** determine if any of the KDFs implemented in the module are the same those described in **SP 800-135rev1**.

There are four possible implementation and documentation cases as follows:

1. If the module implements a KDF from **SP 800-135rev1** and this KDF has not been validated by the CAVP, then the module's certificate **shall not** list this function. The module's Security Policy **shall** make it clear that the corresponding protocol **shall not** be used in an approved mode of operation. In particular, none of the keys derived using this key derivation function can be used in the approved mode.

2. If the module implements a KDF from **SP 800-135rev1** and this KDF has been validated by the CAVP, then the module's certificate **shall** list the KDF on the FIPS-approved algorithm line as a **CVL** entry. If the module's Security Policy claims that the module supports or uses the corresponding

protocol, then the Security Policy **shall** state that no parts of this protocol, other than the KDF, have been tested by the CAVP and CMVP.

3. If the module does not implement any KDFs from **SP 800-135rev1** but the module's Security Policy claims that the module supports or uses parts of the corresponding protocol(s) then no entry on the certificate's approved or allowed algorithms lines is required. As in the case considered above (2), the Security Policy **shall** state that this protocol has not been reviewed or tested by the CAVP and CMVP.

    This situation may occur when a module implements a portion of a protocol, e.g. not including the KDF, and it is the calling application's responsibility to perform the entire protocol.

4. If the module does not implement a KDF from **SP 800-135rev1** and the module's Security Policy makes no claims that the module supports or uses any of the protocols named in **SP 800-135rev1** then the rules explained in this IG do not apply. The module may implement a (non-SP **800-135rev1**) key establishment scheme if it meets the applicable requirements of IG D.8 and IG D.9.

Additional Comments

The use of KDFs described in **NIST SP 800-108** and **NIST SP 800-56C** are out scope for the purposes of this **IG**.

D.12 Requirements for Vendor Affirmation to SP 800-133

Applicable Levels:	*All*
Original Publishing Date:	*08/07/2015*
Effective Date:	*08/07/2015*
Last Modified Date:	*04/25/2017*
Relevant Assertions:	*AS.07.11 and AS.07.16*
Relevant Test Requirements:	
Relevant Vendor Requirements:	

Background

The FIPS 140-2 Implementation Guidance D.2 defines various methods for key establishment within the cryptographic module. These methods include key agreement, key transport, key generation, key entry and key derivation from other keys.

Key generation is a process of securely deriving a cryptographic key from various input sources. These sources **shall** include an output from an approved random number generator located within the physical boundary of the cryptographic module that is deriving the key. In December 2012, NIST published **SP 800-133** that showed how to derive symmetric cryptographic keys and the seeds used when generating private keys for the asymmetric-key algorithms. **SP 800-133** was added to FIPS 140-2 Annex D on February 26, 2014.

Question/Problem

To claim the *vendor affirmation* to **SP 800-133**, what sections of the publication need to be addressed?

Resolution

To claim the vendor affirmation to **SP 800-133** the vendor **shall** generate the module's symmetric keys and seeds used for generating the asymmetric keys using methods described in Section 5 of **SP 800-133**. If a key-generating method involves an XOR, as in formula (1) of **SP 800-133**, this step and the nature of the parameters U and V **shall** be explained in detail by the vendor.

Note that the four examples in Section 5 of **SP 800-133** are informative, not normative. The vendor may either affirm that the module follows one of these examples, or demonstrate that the module uses a different method to meet the independence requirement for U and V. The requirement that U is an output of an approved Random Number/Bit Generator (updated, possibly, using a qualified post-processing method, as explained below) is normative.

The module may further generate a symmetric key or a seed used in generating the asymmetric keys as shown in Section 7.6 of **SP 800-133**, provided that

(a) At least one of the component keys $K_1, \ldots, K_n$ is generated as shown in Section 5 of **SP 800-133** with an independence requirement of Section 7.6 met, and

(b) None of the component keys $K_1, \ldots, K_n$ is generated from a password.

(c) The module may perform the qualified post-processing, explained in IG 7.8, to the output U of an approved DRBG before passing this updated value of U to the key generation process.

Vendor affirmation to **SP 800-133** is required for all methods covered by this IG; that is, when a symmetric key or a seed for asymmetric key generation is generated starting with a random bit string. The module's validation certificate **shall** have a CKG entry only if the module is generating keys for the symmetric-key algorithms. Vendor affirmation to the seed generation compliance with **SP 800-133 shall** be addressed in the Security Policy.

Only one CKG entry is required for the module's certificate, even if the module employs multiple key-generation methods. The Security Policy **shall** provide the details of each method.

Additional Comments

1. If the module directly uses an output U from an approved DRBG as a symmetric key or as a seed to be used in the asymmetric key generation, then it is not necessary to explain that this technique is equivalent to XORing of U and V where V is a string of binary zeros. The Security Policy **shall** state that the resulting symmetric key or a generated seed is an unmodified output from a DRBG.

Test Requirements

Code review, vendor documentation review, and mapping of the module's key generation procedures into the methods described in **SP 800-133**.

D.13 Elliptic Curves and the MODP Groups in Support of Industry Protocols

Applicable Levels:	*All*
Original Publishing Date:	*05/10/2017*
Effective Date:	
Last Modified Date:	
Relevant Assertions:	*AS07.17*
Relevant Test Requirements:	*TE.07.17.01-02*
Relevant Vendor Requirements:	

Background

Various industry protocols employ key establishment techniques. These techniques commonly use the Diffie-Hellman schemes, utilizing either the Finite Field or the Elliptic Curve cryptography. Over the years, the protocols developed a notation of their own to define the sets of the Diffie-Hellman domain parameters and the specific elliptic curves. This notation is often different from the corresponding terminology used in FIPS 186-4, SP 800-56A and in the FIPS 140-2 Implementation Guidance. This results in difficulties when establishing a module's compliance with the CMVP validation requirements and in understanding the key

establishment scheme's encryption strength as expressed in the terminology adopted for the FIPS 140-2 Implementation Guidance.

It is therefore necessary to establish an unambiguous correspondence between the Finite Field Diffie-Hellman domain parameters as defined in SP 800-56A and those documented as the specific Modular Exponential (MODP) Diffie-Hellman groups used in various publications such as the IETF RFCs. Similarly, a mapping has to exist between the NIST Recommended Curves defined in FIPS 186-4 (and referenced in SP 800-56A) and those commonly used in various industry protocols.

Question/Problem

1. What is the relationship of the Diffie-Hellman domain parameters used in SP 800-56A and those defined by the MODP groups?

2. To which NIST recommended curves do the curves used in various industry protocols correspond?

Resolution

The MODP Diffie-Hellman groups used in industry protocols such as the Internet Key Exchange protocol (IKE) employ the so-called "safe" primes; that is, $p = 2q + 1$. This makes them, at present, non-compliant with SP 800-56A, which defines a different relationship between the p and q primes. Future updates of SP 800-56A will address the use of these groups. Until the expiration of the transition period defined in SP 800-131Arev1, the vendor may use these groups' parameters in the allowed key agreement schemes. The values of p and g (the group's generator) are defined, for each MODP group, in the IETF RFC 3526.

IKE v2 (RFC 3526)
2048-bit MODP group (ID=14)
3072-bit MODP group (ID=15)
4096-bit MODP group (ID=16)
6144-bit MODP group (ID=17)
8192-bit MODP group (ID=18)

Table 1. The MODP groups for industry protocols.

The SSLv.2 protocol, as defined in RFC 4253, employs the "diffie-hellman-group14-sha1" key exchange method, which is based on the use of the 2048-bit MODP group.

The elliptic curves used in certain industry standards and the corresponding NIST Recommended Curves are listed in Table 2. Entries in the same row refer to the same elliptic curves under the different names. Absence of the equivalent entries is indicated by '-'.

FIPS 186-4 and SP 800-56A	TLS (RFC 4492, SP 800-52)	IKE v2 (RFC 5903)
P-224	secp224r1	-
P-256	secp256r1	secp256r1
P-384	secp384r1	secp384r1
P-521	secp521r1	secp521r1
K-233	sect233k1	-
K-283	sect283k1	-
K-409	sect409k1	-
K-571	sect571k1	-
B-233	sect233r1	-
B-283	sect283r1	-

| B-409 | sect409r1 | - |
| B-571 | sect571r1 | - |

Table 2. The correspondence between the elliptic curves defined in different standards

All curves listed in Table 2 may be used either in the approved key agreement schemes (if all other applicable requirements are met) or in the allowed schemes. See IG D.8 for more information.

Additional Comments

1. For the purposes of this Implementation Guidance, an industry protocol is any widely-recognized protocol that might be supported by the cryptographic modules being validated for compliance to FIPS 140-2. The MODP groups and the various elliptic curves shown in this Implementation Guidance as protocol-specific have been defined in the IETF RFC publications.

2. This Implementation Guidance does not discuss the Oakley Groups. The reader may refer to the IETF RFC 2409 for more information.

3. While FIPS 186-4 is a digital signature standard, some of its provisions also apply to the key establishment standards, such as SP 800-56A.

4. The **SP 800-56A** notation in this Implementation Guidance refers to the latest publication of the standard: NIST Special Publication 800-56Arev2.

Withdrawn Guidance

W.1 Cryptographic Key Strength Modified by an Entropy Estimate

Applicable Levels:	*All*
Original Publishing Date:	*05/02/2012*
Date Withdrawn:	*08/07/2015*
Last Modified Date:	*01/17/2014*
Relevant Assertions:	*AS.07.13*
Relevant Test Requirements:	*TE.07.13.01, TE.07.13.02*
Relevant Vendor Requirements:	

Background

FIPS 140-2 Section 4.7 states that **"***compromising the security of the key generation method (e.g., guessing the seed value to initialize the deterministic RNG)* **shall** *require as least as many operations as determining the value of the generated key*.**"** To comply with this requirement **TE.07.13.02** states that "*The tester* **shall** *determine the accuracy of any rationale provided by the vendor. The burden of proof is on the vendor; if there is any uncertainty or ambiguity, the tester* **shall** *require the vendor to produce additional information as needed.*"

Question/Problem

A module implements AES-256; the module generates an encryption key to be used with the AES-256 algorithm; the only entropy available to the module is a 160 bit RNG seed that is loaded into the module before the module becomes operational. The RNG does not get re-seeded during the key generation. How can this module comply with the key generation requirements of **AS.07.13** when the generated key has insufficient entropy commensurate with the key length?

Resolution

This problem is similar to the key establishment requirements of **AS.07.19**. Legacy key establishment methods would often reduce the encryption strength of the established key. This scenario was addressed by the introduction of a caveat on the module's validation and annotated in the Security Policy that would clearly state the minimum and maximum security strengths of the established keys. With this additional caveat, the module can be validated by the CMVP.

Similar, the requirement of **AS.07.13** can be met by defining the true cryptographic strength of each key established by the module (using one of the key establishment procedures addressed in IG D.2). The strength of each key generated by the module **shall** be documented in the Security Policy and reflected in the testing laboratories test report submission in addition to all other required key characteristics. If **AS.07.13** is not met by all keys generated in the module, the modules validation **shall** include the following text added to any other applicable certificate caveats as a separate sentence:

> The module generates cryptographic keys whose strengths are modified by available entropy

If a key is generated within the module's cryptographic boundary using an approved RNG, the entropy limitation would be the only constraint on the key's encryption strength. If the key is established using a key agreement or a key wrapping algorithm has a strength limitation based on the entropy estimates, then this **shall** be taken into account when documenting the encryption strength of the established key.

For example, if an AES key is established using an EC Diffie-Hellman algorithm and the strength of the EC Diffie-Hellman key establishment is known to be between 112 and 192 bits but the entropy used in the generation of an Elliptic Curve private key does not exceed 160 bits, then the Security Policy and the test report **shall** state that the encryption strength for the AES key is between 112 and 160 bits.

The precise format of how the entropy-limitation-induced strength limitations will be annotated in the Security Policy and the test report is not addressed at this time. However, both the Security Policy and test report **shall** clearly annotate the entropy-limitation-induced strength for each key.

If the amount of entropy estimated is insufficient to support at least 112 bits of security strength, then the associated algorithm and key **shall** not be used in the approved mode of operation.

Test Requirements

The vendor and tester evidence **shall** be provided under **TE.07.13.01** and **TE.07.13.02**.

W.2 Validating the Transition from FIPS 186-2 to FIPS 186-4

Applicable Levels:	*All*
Original Publishing Date:	*04/23/2012*
Date Withdrawn:	*01/06/2016*
Last Modified Date:	*01/17/2014*
Relevant Assertions:	*AS.01.12*
Relevant Test Requirements:	*TE.01.12.01-02*
Relevant Vendor Requirements:	*VE.01.12.01-02*

Background

FIPS 186-3, *Digital Signature Standard*, was approved in June, 2009 to replace FIPS 186-2. FIPS 186-3 was updated with some minor modifications and was replaced, in July, 2013 with FIPS 186-4. This IG outlines the details of a transition from FIPS 186-2 to FIPS 186-4.

FIPS 186-2 specified the Digital Signature Algorithm (DSA) for the generation and verification of digital signatures, and adopted ANSI X9.31 for the generation and verification of digital signatures using the RSA algorithm, and ANSI X9.62 for the generation and verification of digital signatures using the Elliptic Curve Digital Signature Algorithm (ECDSA). Two additional techniques for the generation and verification of digital signatures using RSA were approved in FIPS 140-2, Annex A: RSASSA-PKCS1-v1_5 and RSASSA-PSS; both are specified in Public Key Cryptography Standard (PKCS) #1, version 2.1, RSA Cryptography Standard.

FIPS 186-4 includes the DSA specification from FIPS 186-2, and adopts the RSA techniques specified in ANSI X9.31 and PKCS #1 (i.e., RSASSA-PKCS1-v1.5 and RSASSA-PSS) and ECDSA as specified in ANSI X9.62. FIPS 186-4 also increases the key lengths allowed for DSA, provides additional requirements for the use of RSA and ECDSA, and includes requirements for obtaining the assurances necessary for valid digital signatures and new methods for generating key pairs and domain parameters (see SP 800-89). While FIPS 186-2 contained specifications for random number generators (RNGs), FIPS 186-4 does not include such specifications, but requires the use of an approved random bit generator, such as one specified in SP 800-90A, for obtaining random bits.

Question/Problem

Transitioning from the validation of implementations of FIPS 186-2 to the validation of implementations on FIPS 186-4 is complicated by a planned transition to the use of key lengths for digital signature generation that provide higher security strengths. The transition schedule is provided in SP 800-131A. IG G.14 addresses the

validation and revalidation issues associated with this transition, as well as the status of the validation of already-validated implementations.

This IG contains the transition rules specific to the validation to the FIPS 186-2 and FIPS 186-4 standards. These transition rules apply to both the cryptographic algorithm validations and the cryptographic module validations that are conducted by the CAVP and CMVP, respectively.

Resolution

1. **CAVP Validation Testing**

   Cryptographic algorithm and key lengths that are categorized as acceptable, deprecated, restricted or legacy-use in **SP 800-131A** **shall** be validated for Federal government use.

   The CAVP is currently testing the following digital signature-specific functions for FIPS186-4; the validation of auxiliary functions (e.g., hash functions and RNGs) is discussed in IG G.14, with reference to **SP 800-131A**.

   - DSA: domain parameter generation and validation, key pair generation, public key validation, and digital signature generation and validation.
   - ECDSA: key pair generation, public key validation, and digital signature generation and verification; only the NIST-recommended curves are used as domain parameters for testing ECDSA.
   - RSA: key pair generation, public key validation, and digital signature generation and verification; RSA has no domain parameters.

   For FIPS 186-2, the set of current CAVP tests is different:

   - DSA: domain parameter validation, public key validation and digital signature verification.
   - ECDSA: public key validation and digital signature verification; only the NIST-recommended curves are used as domain parameters for testing ECDSA.
   - RSA: public key validation and digital signature verification; RSA has no domain parameters.

   The parameter sets that can be tested for DSA, ECDSA and RSA are presented in Table 1 below, along with an indication of the applicable standard (FIPS 186-2 or FIPS 186-4). For DSA, the key length is commonly considered to be the value of L. For ECDSA, the key length is considered to be the bit length of n. For RSA, the key length is considered to be nlen, which is the bit length of the modulus n. Note that the following testable parameter sets are subject to the transitions provided in **SP 800-131A**:

   - DSA: L = 1024, N = 160,
   - ECDSA: the B-163, K-163 and P-192 elliptic curves
   - RSA: nlen = 1024 and 1536.

   Also note that in FIPS 186-4, e = 3 and 17, and nlen ≠ 1024, 2048 or 3072 are not specified, so these values will not be tested during FIPS 186-4 validation. The value of nlen = 1024 will be tested for public key validation and signature verification purposes only.

   See FIPS 186-4 for the precise meanings of L, N, nlen and e for the specific digital signature algorithm.

Table 1. CAVP-testable parameter sets for DSA, ECDSA and RSA

DSA (L, N)	ECDSA	RSA	
		Modulus length (*nlen*)	**Public exponent value** (*e*)
$L = 1024, N = 160$ Both FIPS 186-2 and FIPS 186-4	All NIST-recommended curves	$nlen = 1024$ Both FIPS 186-2 and FIPS 186-4	FIPS 186-2: $e = 3, 17, 2^{16} + 1$
$L = 2048, N = 224$ FIPS 186-4 only		$nlen = 1536$ FIPS 186-2 only	

$L = 2048, N = 256$ FIPS 186-4 only	Both FIPS 186-2 and FIPS 186-4	$nlen = 2048$ Both FIPS 186-2 and FIPS 186-4	FIPS 186-4: $2^{16}+1 \leq e < 2^{256}$, where e is odd
$L = 3072, N = 256$ FIPS 186-4 only		$nlen = 3072$ Both FIPS 186-2 and FIPS 186-4	
		$nlen = 4096$ FIPS 186-2 only	

2. **FIPS 186-2 to FIPS 186-4 Validation Transition Rules**

The validation transition rules are as follows:

1. Conformance to FIPS 186-4:

    a. Cryptographic algorithm and module implementations may be tested by the CST labs for conformance to FIPS 186-4 (or parts of FIPS 186-4) and submitted for validation. An example of an algorithm or module implementation that conforms to only part of FIPS 186-4 might be an implementation that performs key pair generation, but does not perform domain parameter generation or validation, or an implementation that performs signature verification, but not signature generation.

    CAVP: The CAVP will accept from the CST labs test results of cryptographic algorithm implementations of FIPS 186-4 (or parts of FIPS 186-4) that contain testable parameter sets, with key lengths that are categorized as either acceptable, deprecated or legacy-use as specified in **SP 800-131A**. The testable parameter sets are listed in Table 1 above. Only implementations of those testable parameter sets whose key lengths are classified as either acceptable or deprecated in **SP 800-131A** may be validated for domain parameter generation, key pair generation and digital signature generation. Implementations of domain parameter validation, public key validation and digital signature verification may be validated at any testable key length. Further information about the validation of implementations containing key lengths categorized as deprecated or legacy-use is provided in IG G.14.

    CMVP: The CMVP will accept from the CST labs test reports of cryptographic modules containing implementations of FIPS 186-4 (or parts of FIPS 186-4) for which the cryptographic algorithms and testable parameter sets have been validated by the CAVP. Further information about the validation and revalidation of modules containing key lengths categorized as deprecated or legacy-use is provided in IG G.14.

2. Conformance to FIPS 186-2:

    a. After **December 31, 2013,** implementations of domain parameter generation, key pair generation and digital signature generation as specified in FIPS 186-2 are *no longer validated* by the CAVP or CMVP. Already-validated implementations remain valid, subject to the key length usage restrictions specified as disallowed in **SP 800-131A**.

    As time and resources permit, the following actions will be taken by the CAVP or CMVP for already-validated implementations of these functions:

CAVP: Algorithm validation listings for already-validated implementations that contain one or more testable key lengths permitted by FIPS 186-2 that are disallowed will be annotated to indicate the key lengths that are disallowed. If an already-validated implementation only supports testable key lengths permitted by FIPS 186-2 that are disallowed, the algorithm validation will be revoked. Complete validations and parts of validations using testable key lengths that are categorized as acceptable will remain valid.

CMVP: For already-validated modules:

- If an algorithm validation listing has been annotated to disallow some, but not all, of the testable key lengths (i.e., only part of a validation is disallowed), the module's CMVP validation certificate will not be changed.

- If an algorithm validation is revoked by the CAVP, the module's CMVP validation certificate will be updated to remove the algorithm's listing from the "FIPS-approved algorithms" line of the certificate and placed on the "Other algorithms" line.

- For further information about the CMVP validation of a module containing transitioning algorithms and key lengths, see IG G.14.

b. Cryptographic algorithm and module implementations that perform domain parameter validation, public key validation and digital signature verification may be tested by the CST labs for conformance to FIPS 186-2 (or parts of FIPS 186-2) and submitted for validation, subject to the following conditions.

CAVP: The CAVP will accept test results from the CST labs of cryptographic algorithm implementations of FIPS 186-2 (or parts of FIPS 186-2) that contain testable key lengths permitted by FIPS 186-2 that are categorized as either acceptable or legacy-use as specified in **SP 800-131A**.

CMVP: New modules (3SUB and 5SUB submissions) and already-validated modules containing digital signature processes conforming to FIPS 186-2 that have algorithm validations issued by the CAVP may be validated or revalidated, as appropriate.

W.3 CAVP Requirements for Vendor Affirmation of SP 800-90

Applicable Levels:	*All*
Original Publishing Date:	*06/21/2007*
Effective Date:	*06/21/2007*
Transition End Date:	*02/15/2008*
Last Modified Date:	*06/21/2007*
Relevant Assertions:	*AS.01.12*
Relevant Test Requirements:	*TE01.12.01*
Relevant Vendor Requirements:	*VE.01.12.01*

Background

SP 800-90 was added to FIPS 140-2 Annex C on January 24, 2007. FIPS 140-2 Implementation Guidance, IG A.3, was added January 25, 2007. Until CAVP testing for SP 800-90 is available, IG A.3 is applicable. SP 800-90 includes information beyond the specifications of the deterministic random bit generation (DRBG) algorithms themselves, e.g., stricter entropy requirements, and assurance.

Question/Problem

To claim *vendor affirmation* to SP 800-90, what sections of the publication need to be addressed?

Resolution

To claim *vendor affirmation*, the vendor shall affirm compliance with the following three sections of SP 800-90, *Recommendation for Random Number Generation Using Deterministic Random Bit Generators*:

Section 9	DRBG Mechanism Functions
Section 10	DRBG Algorithm Specifications
Section 11	Assurance

The vendor is not required to meet the requirements in Section 8, including the entropy requirements in Section 8.6. Entropy requirements are addressed in AS.07.13.

Additional Comments

The requirements of SP 800-90 depend on several NIST approved security functions, for example, SHA, AES, and three-key Triple-DES. The validation testing for these supporting security functions is found in their corresponding validation test suites and, therefore, they shall be validated as a prerequisite to SP 800-90 vendor affirmation.

To claim vendor affirmation to SP 800-90, the following supporting security functions, if used, shall be tested and validated:

- Supported hash algorithms (SHA224, SHA256, SHA384, and/or SHA512)
- Supported Message Authentication Code (MAC) algorithm (HMAC)
- Advanced Encryption Standard (AES)
- Three key Triple-DES

Derived Test Requirements

Upon the following successful review, the CST Lab shall affirm by annotating the algorithm entry per the IG G.13 annotation requirements

Required Vendor Information

The vendor shall provide evidence that their implementation implements the sections outlined above completely and accurately. This shall be accomplished by documentation and code review.

Required Test Procedures

The tester shall review the vendor's evidence demonstrating that their implementation conforms to the specifications specified above. This shall be accomplished by documentation and code review. The tester shall verify the rationale provided by the vendor.

W.4 Use of other Core Symmetric Algorithms in ANSI X9.31 RNG

Applicable Levels:	*All*
Original Publishing Date:	*01/21/2005*
Effective Date:	*01/21/2005*
Last Modified Date:	*01/21/2005*
Relevant Assertions:	*AS.07.10*
Relevant Test Requirements:	*TE07.10.01*
Relevant Vendor Requirements:	*VE.07.10.01*

Background

ANSI X9.31 Appendix A.2.4 specifies 2-key Triple-DES as the core symmetric algorithm in its deterministic random number generator.

Question/Problem

Is it acceptable to use other FIPS approved symmetric algorithms as the ANSI X9.31 Appendix A.2.4 RNG core algorithm?

Resolution

In addition to 2-key Triple-DES, it is acceptable to use the following FIPS approved symmetric algorithms as the ANSI X9.31 RNG core algorithm:

- AES
- 3-key Triple-DES
- SKIPJACK

CAVS testing is available for the 2-key Triple-DES, 3-key Triple-DES and AES. Until CAVS testing is available for RNG testing using SKIPJACK, for module testing purposes, the core cryptographic algorithm SKIPJACK **shall** be validated and the RNG implementation will be marked as "vendor affirmed".

Additional Comments

FIPS 140-2 Annex C has been updated to include reference to the NIST RNG specification for implementing 3-key Triple-DES and AES with ANSI X9.31 Appendix A.2.4.

W.5 RNGs: Seeds, Seed Keys and Date/Time Vectors

Applicable Levels:	*All*
Original Publishing Date:	*11/16/2007*
Date Withdrawn:	*05/10/2016*
Last Modified Date:	*11/16/2007*
Relevant Assertions:	*AS.07.09*
Relevant Test Requirements:	*TE07.09.01*
Relevant Vendor Requirements:	*VE.07.09.01*

Background

An RNG may employ a seed and seed key and a Date/Time vector for its operation. FIPS 140-1 IG 8.7 provides a basis for the requirements related to the ANSI X9.31 RNG **seed**, **seed key** and Date/Time vector. The document titled *NIST Recommended Random Number Generator based on ANSI X9.31 Appendix A.2.4 using the 3-Key Triple DES and AES Algorithms* allows for the use of Triple-DES and AES.

Questions/Problems

1. In the case where an RNG employs a **seed** and **seed key**, how does **AS.07.09** apply?

2. In the case where an RNG employs a Date/Time vector, what, if any, additional attributes apply?

Resolution

1. **AS.07.09** specifies that the seed and seed key **shall** not have the same value.

    During initialization of the **seed** or **seed key**, the initialization data provided for one, **shall** not be provided as initialization data to the other. The **seed** or **seed key** or both may be re-initialized prior to each call for a random data value.

2. The Date/Time vector **shall** be updated on each iteration or call to the RNG. In lieu of a Date/Time vector, an incrementer may be used. The Date/Time vector or incrementer **shall** be a non-repeating value during each instance of the module's power-on state.

Additional Comments

ANSI X9.31 specifies that the **seed shall** also be kept secret. As such, the **seed** is considered a CSP and **shall** meet all the requirements pertaining to CSPs.

AS.07.14 and **AS.07.23** are applicable to the **seed key**.

The seed key is sometimes referred as the RNG key; the key used by the underlining encryption algorithm(s) implemented by the RNG.

W.6 Definition of an NDRNG

Applicable Levels:	*All*
Original Publishing Date:	*05/02/2012*
Date Withdrawn:	*05/10/2016*
Last Modified Date:	*05/02/2012*
Relevant Assertions:	*AS.07.0, AS.07.07 and AS.07.10*
Relevant Test Requirements:	
Relevant Vendor Requirements:	

Background

FIPS 140-2 defines a random number generator as follows: *Random Number Generators (RNGs) used for cryptographic applications typically produce a sequence of zero and one bits that may be combined into sub-sequences or blocks of random numbers. There are two basic classes: deterministic and nondeterministic. A deterministic RNG consists of an algorithm that produces a sequence of bits from an initial value called a seed. A nondeterministic RNG produces output that is dependent on some unpredictable physical source that is outside human control.*

AS.07.07: (Levels 1, 2, 3, and 4) Nondeterministic RNGs shall comply with all applicable RNG requirements of this standard.

AS.07.10: (Levels 1, 2, 3, and 4) Documentation shall specify each RNG (Approved and non-Approved) employed by a cryptographic module.

SP 800-90A addresses deterministic RBGs and nondeterministic RBG definitions. *Draft* **SP 800-90B** addresses entropy testing.

Approved RNGs are referenced in FIPS 140-2 Annex C. At the time of this IGs publishing, the only approved RNGs referenced are deterministic. Non-approved nondeterministic RNGs (NDRNG) implemented in a cryptographic module may take various forms or implementations. The followings are examples:

- ring oscillator;
- noisy diode;
- transistor thermal noise;
- gathering or sampling of different nondeterministic machine states;
- gathering or sampling of different nondeterministic user actions (e.g. mouse movements, etc.);
- function calls to operating system provided services (e.g., /dev/rand); etc.

A NDRNG may collect entropy from one or many sources (e.g. sampling from a single physical noise source or from many sources such as the time between various operator actions and the time of day, etc.) and the number of random bits collected may be different each time.

Question/Problem

What defines a nondeterministic RNG (NDRNG) and what are the requirements that apply to it?

Resolution

Any hardware, firmware or software construct that collects or samples bits from single or multiple sources within the modules defined boundary and converts this collection into a single random stream of bits to be used as a seed input for an approved RNG or as random input bits for other processes **shall** be defined as a NDRNG within the scope of FIPS 140-2.

All the requirements of FIPS 140-2 Section 4.7.1; the self-test requirements specified in FIPS 140-2 Section 4.9; and the conditional Continuous Random Number Generator Test (CRNGT) addressed in IG 9.8 **shall** apply to an NDRNG implemented in a module. The NDRNG **shall** be identified in the security policy and fully described in the test report. The description **shall** include all entropy sources and applicable smoothing function.

W.7 CAVP Requirements for Vendor Affirmation of SP 800-38D

Applicable Levels:	*All*
Original Publishing Date:	*12/18/2007*
Date Withdrawn:	*05/10/2016*
Transition End Date:	*03/24/2009*
Last Modified Date:	*12/18/2007*
Relevant Assertions:	*AS.01.12*
Relevant Test Requirements:	*TE01.12.01*
Relevant Vendor Requirements:	*VE.01.12.01*

Background

SP 800-38D was added to FIPS 140-2 Annex A on December 18, 2007. IG A.3 was added January 25, 2007. Until CAVP testing for **SP 800-38D** is available, IG A.3 is applicable. **SP 800-38D** includes information beyond the specifications of the Galois/Counter Mode itself; i.e., uniqueness requirements on IVs and keys.

Question/Problem

To claim *vendor affirmation* to **SP 800-38D**, what sections of the standard need to be addressed?

Resolution

Validation testing for **SP 800-38D**, *Recommendation for Block Cipher Modes of Operation: Galois/Counter Mode (GCM) and GMAC* includes validation testing for the authenticated encryption function and the authenticated decryption function. To claim *vendor affirmation* to **SP 800-38D**, information contained in the following sections that are supported by the implementation under test (IUT) **shall** be implemented:

Section 5	Elements of GCM
Section 6	Mathematical Components of GCM
Section 7	GCM Specifications

Additional Comments

1. The GCM functions in **SP 800-38D** require the forward direction of an approved symmetric key block cipher with a block size of 128 bits. Currently, the only NIST-approved 128-bit block cipher is the Advanced Encryption Standard (AES) algorithm specified in Federal Information Processing Standard (FIPS) Pub. 197. The validation testing for the forward direction of this supporting algorithm, the AES Cipher (Encrypt) function, is found in its corresponding validation test suite and, therefore, **shall** be validated as a prerequisite to **SP 800-38D** vendor affirmation.

2. The **SP800-38D** Self Tests required in cryptographic module implementations **shall** consist of a known answer that validates the correctness of the GCM elements, GCM mathematical components and GCM specifications of the two GCM functions, namely, the authenticated encryption function and the authenticated decryption function.

3. Section 8, *Uniqueness Requirement on IVs and Keys*, and Section 9, *Practical Considerations for Validating Implementations*, contain requirements for module validation, which is conducted by the CMVP. Therefore, Section 8 and Section 9 are outside of the scope of algorithm validation.

Derived Test Requirements

Upon the following successful review, the CST Lab **shall** affirm by annotating the algorithm entry per the IG G.13 annotation requirements

Required Vendor Information

The vendor **shall** provide evidence that their implementation implements the sections outlined above completely and accurately. This **shall** be accomplished by documentation and code review.

Required Test Procedures

The tester **shall** review the vendor's evidence demonstrating that their implementation conforms to the specifications specified above. This **shall** be accomplished by documentation and code review. The tester **shall** verify the rationale provided by the vendor.

W.8 CAVP Requirements for Vendor Affirmation of FIPS 186-3 Digital Signature Standard

Applicable Levels:	*All*
Original Publishing Date:	*07/07/2009*
Date Withdrawn:	*05/10/2016*
Transition End Date:	***10/02/2009** – See Below*
Transition End Date:	***06/30/2010** – See Below*
Transition End Date:	***08/27/2010** – See Below*
Transition End Date:	***06/03/2011** – See Below*
Last Modified Date:	*04/09/2010*
Relevant Assertions:	*AS.01.12*
Relevant Test Requirements:	*TE01.12.01*
Relevant Vendor Requirements:	*VE.01.12.01*

Transition

The Transition End Date for those elements of FIPS 186-3 DSA which CAVP testing is currently available [if not supporting the generation and validation of provably prime domain parameters p and q and canonical generation and validation of domain parameter g] is: **October 02, 2009**.

With the March 31, 2010 CAVP release of CAVS 9.0, testing for all elements of FIPS 186-3 DSA are available. For the new and final set of elements, the transition end date is: **June 30, 2010**

With the May 27, 2010 CAVP release of CAVS 10.0, testing for all elements of FIPS 186-3 ECDSA are available. The transition end date is: **August 27, 2010**

With the March 03, 2011 CAVP release of CAVS 11.0, testing for all elements of FIPS 186-3 RSA are available. The transition end date is: **June 03, 2011**

The transition plan for migration to FIPS 186-3 is found in IG G.15.

Background

Federal Information Processing Standard (FIPS) 186-3, *Digital Signature Standard (DSS)* was added to FIPS 140-2 Annex A on June 18, 2009. FIPS 186-3 specifies a suite of algorithms that can be used to generate a digital signature. These include the DSA, ECDSA, and RSA algorithms. CAVP testing is currently available for DSA as specified in FIPS 186-3, with the exception of generation and validation of provably prime domain parameters p and q and canonical generation and validation of domain parameter g. CAVP testing is not available for ECDSA and RSA. Until CAVP testing for FIPS 186-3 is available for the above elements of DSA and for ECDSA and RSA algorithms, this IG is applicable.

Question/Problem

To claim *vendor affirmation* to the above listed domain parameter generation and validation methods of DSA, ECDSA, and RSA as specified in FIPS 186-3, what sections of the publication needs to be addressed?

Resolution

Validation testing for FIPS 186-3, *Digital Signature Standard (DSS)* is separated into the three digital signature algorithms. Validation testing is available for FIPS 186-3 DSA, with the exception of the domain parameter generation and validation method listed above. These methods, along with FIPS 186-3 ECDSA and RSA, will require *vendor affirmation* until validation testing is available in the CAVS tool.

<u>Vendor Affirmation for FIPS 186-3 DSA Domain Parameter Generation and Validation for provable primes p and q and verifiable canonical generation of the generator g</u>

To claim vendor affirmation for FIPS 186-3 DSA generation of provably primes *p* and *q*:

1. The vendor must affirm that the method of FIPS 186-3 A.1.2.1.2 is used to generate provable primes *p* and *q*.

2. 2. The vendor **shall** use the CAVP to validate the underlying SHA implementation used by this DSA implementation and report the validation number.

To claim vendor affirmation for FIPS 186-3 DSA verifiable canonical generation of the generator *g*:

1. The vendor must affirm that the method of FIPS 186-3 A.2.3 is used for verifiable canonical generation of the generator *g*.

2. 2. The vendor **shall** use the CAVP to validate the underlying SHA implementation used by this DSA implementation and report the validation number.

To claim vendor affirmation for FIPS 186-3 DSA validation of provable primes *p* and *q*:

1. The vendor must affirm that the method of FIPS 186-3 A.1.2.2 is used for validation of provable primes *p* and *q*.

2. The vendor **shall** use the CAVP to validate the underlying SHA implementation used by this DSA implementation and report the validation number.

To claim vendor affirmation for FIPS 186-2 DSA validation when the canonical generation of the generator *g* was used:

1. The vendor must affirm that the method of FIPS 186-3 A.2.4 is used for validation of *g* where the verifiable canonical generation of *g* was used.

2. The vendor **shall** use the CAVP to validate the underlying SHA implementation used by this DSA implementation and report the validation number.

Vendor Affirmation for FIPS 186-3 ECDSA

To claim vendor affirmation for FIPS 186-3 ECDSA, the following **shall** be affirmed:

1. For all ECDSA implementations, the assurances listed in FIPS 186-3, Section 3 and 3.1 **shall** be defined. If Signature Validation is implemented, Section 3.3 Assurances are also required.

2. If Key Pair Generation is implemented:

   a. The vendor **shall** affirm that at least one of the methods in FIPS 186-3 Appendix B.4 is used to generate d and Q, the private and public keys.

   b. The implementation must support at least one of the NIST curves in FIPS 186-3 Appendix D.1.

   c. The vendor **shall** use the CAVP to validate the underlying RNG or DRBG implementation used by this ECDSA implementation and report the validation number.

3. If Public Key Validation (PKV) is implemented:

   a. The vendor must run the FIPS 186-2 ECDSA PKV tests and report the validation number.

4. If Signature Generation is implemented:

   a. The vendor **shall** affirm compliance with FIPS 186-3 Section 6.4.

   b. The vendor **shall** affirm compliance with FIPS 186-3 Appendix B.5 for generation of the Per-message secret number.

   c. The vendor **shall** use the CAVP to validate the underlying SHA implementation used by this ECDSA implementation and report the validation number.

5. If Signature Validation is implemented:

   a. The vendor **shall** affirm compliance with FIPS 186-3 Section 6.4.

   b. The vendor **shall** use the CAVP to validate the underlying SHA implementation used by this ECDSA implementation and report the validation number.

Vendor Affirmation for FIPS 186-3 RSA

To claim vendor affirmation for FIPS 186-3 RSA, the following **shall** be affirmed:

1. For all RSA implementations, the assurances listed in Section 3 **shall** be defined.

2. If Key Pair Generation is implemented:

   a. The vendor **shall** affirm that at least one of the methods in FIPS 186-3 Appendix B.3 is used to generate the key pairs.

   b. The vendor **shall** affirm that at least one of the modulus lengths 1024, 2048 or 3072 bits is supported by the implementation. Note, the length of the modulus is dependent on the generation method selected. See FIPS 186-3 Appendix B.3.1.

   c. The vendor **shall** affirm that the public exponent e **shall** be selected with the following constraints:

      i. The public verification exponent e **shall** be selected prior to generating the primes p and q, and the private signature exponent d.

      ii. The exponent e **shall** be an odd positive integer such that $2^{16} < e < 2^{256}$.

   d. The vendor **shall** use the CAVP to validate the underlying SHA implementation used by this RSA Key Pair Generation implementation and report the validation number.

   e. The vendor **shall** affirm that the length in bits of the hash function output block **shall** meet or exceed the security strength associated with the bit length of the modulus n (see **SP 800-57**).

   f. If the RSA parameters are randomly generated (i.e., the primes p and q, and optionally, the public key exponent e), the vendor **shall** use the CAVP to validate the underlying RNG or DRBG implementation used by this RSA implementation and report the validation number.

3. If ANSI X9.31 RSA Signature Generation or Signature Verification is implemented:

   a. The vendor must run the ANSI X9.31 RSA validation tests and report the validation number. (Note that the specification in FIPS 186-3 Section 5.4 concerning the extraction of the hash value *H(M)'* from the data structure *IR'* is tested in the ANS X9.31 RSA validation testing

supplied by the CAVP.)

    b. The vendor **shall** affirm that at least one of the modulus lengths 1024, 2048 or 3072 bits is supported by the implementation.

    c. The vendor **shall** use the CAVP to validate the underlying RNG or DRBG implementation used by this RSA implementation and report the validation number.

4. If PKCS #1 Version 1.5 and/or PKCS #1 Version PSS is implemented:

    a. The vendor **shall** confirm that implementations that generate RSA key pairs use the criteria and methods in FIPS 186-3 Appendix B.3 to generate those key pairs.

    b. The vendor **shall** use the CAVP to validate the underlying approved SHA implementation used by this implementation and report the validation number.

    c. The vendor **shall** confirm that only two prime factors p and q **shall** be used to form the modulus n.

    d. The vendor **shall** use the CAVP to validate the underlying RNG or DRBG implementation used by this RSA implementation and report the validation number.

    e. If PKCS #1 Version 1.5 is implemented, the vendor must run the PKCS1.5 validation tests for Signature Generation and/or Signature Verification and report the validation number.

    f. If PKCS#1 Version PSS is implemented, the vendor must run the PKCSPSS validation tests for Signature Generation and/or Signature Verification and report the validation number.

    g. If PKCS#1 Version PSS is implemented, the vendor **shall** confirm that the implementation's salt length (*sLen*) satisfies $0 <= sLen <= hlen$, where *hlen* is the length of the hash function output block.

Annotation

Refer to IG G.13 for annotation examples.

FIPS 140-2 Section 4.9 Self-Tests

In addition to the above requirements, all algorithmic implementations **shall** meet all the applicable self-test requirements in FIPS 140-2 Section 4.9.

Derived Test Requirements

Upon the following successful review, the CST Lab **shall** affirm by annotating the algorithm entry per the IG G.13 annotation requirements.

Required Vendor Information

The vendor **shall** provide evidence that their implementation implements the sections outlined above completely and accurately. This **shall** be accomplished by documentation and code review.

Required Test Procedures

> The tester **shall** review the vendor's evidence demonstrating that their implementation conforms to the specifications specified above. This **shall** be accomplished by documentation and code review. The tester **shall** verify the rationale provided by the vendor.

W.9 CAVP Requirements for Vendor Affirmation of NIST SP 800-38E

Applicable Levels:	*All*
Original Publishing Date:	*01/27/2010*
Date Withdrawn:	*05/10/2016*
Transition End Date:	*06/30/2010*
Last Modified Date:	*04/09/2010*
Relevant Assertions:	*AS.01.12*
Relevant Test Requirements:	*TE01.12.01*
Relevant Vendor Requirements:	*VE.01.12.01*

Background

SP 800-38E, *Recommendation for Block Cipher Modes of Operation: The XTS-AES Mode for Confidentiality on Block-Oriented Storage Devices*, was added to FIPS 140-2 Annex A on January 27, 2010. Until CAVP testing for **SP 800-38E** is available, this IG is applicable. **SP 800-38E** approves the XTS-AES mode as specified in the Institute of Electrical and Electronics Engineers, Inc (IEEE) Std. 1619-2007, subject to one additional requirement on the lengths of the data units. That is, the data unit for any instance of an implementation of XTS-AES **shall not** exceed 2^{20} blocks.

Question/Problem

To claim vendor affirmation to **SP 800-38E**; what sections of the IEEE standard and the NIST Special Publication need to be addressed?

Resolution

To claim vendor affirmation to **SP 800-38E**, the information contained in the following sections that are supported by the Implementation Under Test (IUT) **shall** be implemented:

SP 800-38E	Section 4	Conformance
IEEE Std. 1619-2007	Section 5	XTS-AES transform

The following information **shall** be specified:

1. The underlying AES implementation **shall** be validated by the CAVP:

    a. For XTS-AES Encrypt: the validation referenced **shall** include an AES mode of operation that uses the forward cipher function.

    b. For XTS-AES Decrypt: the validation referenced **shall** include an AES mode of operation that uses the forward and inverse cipher function (i.e., AES-ECB or AES-CBC).

2. The XTS-AES key sizes supported: XTS-AES-128 (256 bits) AND/OR XTS-AES-256 (512 bits).

3. The block sizes supported: complete blocks only OR complete and partial blocks

4. Procedures supported: XTS-AES encryption AND/OR XTS-AES decryption

5. Provide assurance that the length of the data unit for any instance of an implementation of XTS-AES **shall** not exceed 2^{20} blocks.

6. Provide assurance that the XTS-AES key **shall** not be associated with more than one key scope.

Additional Comments

Bullets 5 and 6 above satisfy the **shall** statements included in **SP 800-38E** and IEEE Std 1619-2007 that are not testable by the CAVP.

Upon the following successful review, the CST Lab **shall** affirm by annotating the FIPS approved algorithm entry as follows:

AES (XTS-AES: AES Cert. #nnn, vendor affirmed)

When CAVP CAVS testing is available, the annotation will simply change to:

AES (Cert. #nnn)

Derived Test Requirements

Required Vendor Information

The vendor **shall** provide evidence that their implementation implements the sections outlined above completely and accurately. This **shall** be accomplished by documentation and code review.

Required Test Procedures

The tester **shall** review the vendor's evidence demonstrating that their implementation conforms to the specifications specified above. This **shall** be accomplished by documentation and code review. The tester **shall** verify the rationale provided by the vendor.

W.10 CAVP Requirements for Vendor Affirmation of SP 800-56A

Applicable Levels:	*All*
Original Publishing Date:	*06/21/2007*
Date Withdrawn:	*05/10/2016*
Transition End Date:	*03/24/2009 – See Below*
Transition End Date:	*07/12/2011 – See Below*
Last Modified Date:	*07/15/2011*
Relevant Assertions:	*AS.01.12*
Relevant Test Requirements:	*TE01.12.01*
Relevant Vendor Requirements:	*VE.01.12.01*

Transition

With the December 24, 2008 CAVP release of CAVS 7.0, the Transition End Date for the *vendor affirmation* to one of the complete key agreement schemes from **SP 80056A** was: **March 24, 2009**.

With the April 12, 2011 CAVP release of CAVS 11.1, testing for the **SP 800-56A** primitives have become available. As of **July 12, 2011**, all new module submissions to the CMVP that claim to implement the **SP 800-56A** primitives **shall** be tested by the CAVP (Per IG G.13 will be represented as a **CVL** validation).

Background

SP 800-56A was added to FIPS 140-2 Annex D on January 24, 2007. FIPS 140-2 Implementation Guidance, IG A.3, was added January 25, 2007. Until CAVP testing for **SP 800-56A** is available, IG A.3 is applicable. **SP 800-56A** includes information beyond the specifications of the key agreement algorithm itself; i.e. Instructions to the implementer to aid in the implementation of the algorithm.

Question/Problem

To claim *vendor affirmation* to **SP 800-56A**, what sections of the publication need to be addressed?

Resolution

Validation testing for **SP 800-56A**, *Recommendation for Pair-Wise Key Establishment Schemes Using Discrete Logarithm Cryptography* includes validation testing for the key agreement schemes and key confirmation. To claim *vendor affirmation* to **SP 800-56A**, information contained in the following sections that are supported by the implementation under test (IUT) **shall** be implemented:

Section 5.6.2.4	FFC Full Public Key Validation Routine (if implement FFC)
Section 5.6.2.5	ECC Full Public Key Validation Routine (if implement ECC)
Section 5.7	DLC Primitives
Section 5.8	Key Derivation Functions for Key Agreement Schemes
Section 6	Key Agreement

If key confirmation is supported by the implementation, the applicable information contained in the following section must be implemented:

Section 8	Key Confirmation

Additional Comments

1. The components in **SP 800-56A shall** only be used within the **SP 800-56A** protocol. This includes the full public key validation routines, the DLC primitives, the key derivation functions, the key agreement functions, and the key confirmation functions.

2. The requirements specified in **SP 800-56A** depend on several NIST approved security functions, for example, SHA, DSA, ECDSA, etc. While validation testing for **SP 800-56A** concentrates on the key agreement and key confirmation components, other supporting security functions are not thoroughly tested by the testing in **SP 800-56A**. The validation testing for these supporting security functions are found in the validation test suite for this specific function. Therefore, these supporting security functions **shall** be validated as a prerequisite to **SP 800-56A** vendor affirmation.

   To claim vendor affirmation to **SP 800-56A**, the underlying security functions used by this IUT **shall** be tested and validated prior to claiming vendor affirmation. These include:

   - Supported hash algorithms (SHA1, SHA224, SHA256, SHA384, and/or SHA512)
   - Supported Message Authentication Code (MAC) algorithms (CMAC, CCM, and/or HMAC)
   - Supported Random Number Generators (RNG)
   - If Finite Field Cryptography (FFC) is supported,
     - If the IUT generates domain parameters the DSA PQG generation and/or verification tests.
     - If the IUT generates key pairs, the DSA key pair generation tests.
   - If Elliptic Curve Cryptography (ECC) is supported,
     - If the IUT generates key pairs, the ECDSA key pair generation test and/or the Public Key Validation (PKV) test.

3. **SP 800-56A** self-tests required in cryptographic module implementations must consist of a known answer test that validates the correctness of the implemented DLC primitives and key derivation functions for each key agreement scheme implemented.

Annotation

Refer to IG G.13 for annotation examples.

Derived Test Requirements

Upon the following successful review, the CST Lab **shall** affirm by annotating the algorithm entry per the IG G.13 annotation requirements.

Required Vendor Information

The vendor **shall** provide evidence that their implementation implements the sections outlined above completely and accurately. This **shall** be accomplished by documentation and code review.

Required Test Procedures

The tester **shall** review the vendor's evidence demonstrating that their implementation conforms to the specifications specified above. This **shall** be accomplished by documentation and code review. The tester **shall** verify the rationale provided by the vendor.

W.11 Requirements for Vendor Affirmation of SP 800-108

Applicable Levels:	*All*
Original Publishing Date:	*07/15/2011*
Date Withdrawn:	*05/10/2016*
Transition End Date:	*06/23/2012*
Last Modified Date:	*07/15/2011*
Relevant Assertions:	*AS.07.11 and AS.07.16*
Relevant Test Requirements:	*TE07.11.01-02 and TE07.16.01-02*
Relevant Vendor Requirements:	*VE.07.11.01 and VE.07.16.01*

Background

SP 800-108 was published October 2009 and added to FIPS 140-2 Annex D on January 04, 2011. Until CAVP testing for **SP 800-108** is available, IG A.3 is applicable. This Special Publication defines the methods for deriving additional keying material from the already-established symmetric keys.

Question/Problem

To claim *vendor affirmation* to **SP 800-108**, what sections of the publication need to be addressed and what are the applicable documentation requirements?

Resolution

The entire text of **SP 800-108** is applicable. The cryptographic module may support key derivation using the key derivation functions from Section 5.1, Section 5.2 or Section 5.3 of **SP 800-108**. The module may implement any combination of these KDFs. This choice and the corresponding capabilities of the cryptographic module **shall** be clearly stated in the module's Security Policy.

The module's Security Policy **shall** state the possible range of the cryptographic strengths of the keys derived using the **SP 800-108** methodology. The instructions for how to determine this strength are in Section 7 of **SP 800-108**.

The vendor **shall** comply with all "**shall**" statements in **SP 800-108**. These refer to but are not limited to the sizes of the parameters used in the KDF computations and the possible uses of the derived keys.

Annotation

Refer to IG G.13 for annotation examples (**KBKDF**).

Derived Test Requirements

Upon the following successful review, the CST Lab **shall** affirm by annotating the algorithm entry per the IG G.13 annotation requirements.

Required Vendor Information

The vendor **shall** provide evidence that their implementation complies with the requirements of **SP 800-108** and of the documentation requirements of this Implementation Guidance. This **shall** be accomplished by documentation and code review.

Required Test Procedures

The tester **shall** review the vendor's evidence demonstrating that their implementation conforms to the specifications specified above. This **shall** be accomplished by documentation and code review.

Additional Comments

No self-tests are required to claim *vendor affirmation* to **SP 800-108**.

W.12 Requirements for Vendor Affirmation of SP 800-135rev1

Applicable Levels:	*All*
Original Publishing Date:	*07/15/2011*
Date Withdrawn:	*05/10/2016*
Transition End Date:	*06/23/2012*
Last Modified Date:	*07/15/2011*
Relevant Assertions:	*AS.07.11 and AS.07.16*
Relevant Test Requirements:	*TE07.11.01-02 and TE07.16.01-02*
Relevant Vendor Requirements:	*VE.07.11.01 and VE.07.16.01*

Background

SP 800-135rev1 was published December 2011 and added to FIPS 140-2 Annex D on April 23, 2012. Until CAVP testing for **SP 800-135rev1** is available, IG A.3 is applicable. This Special Publication defines the recommendation for existing application-specific key derivation functions.

Question/Problem

To claim *vendor affirmation* to **SP 800-135rev1**, what sections of the publication need to be addressed and what are the applicable documentation requirements?

Resolution

The entire text of **SP 800-135rev1** is applicable.

The vendor **shall** comply with all "**shall**" statements in **SP 800-135rev1**.

Annotation

Refer to IG G.13 for annotation examples (**CVL**).

Derived Test Requirements

Upon the following successful review, the CST Lab **shall** affirm by annotating the algorithm entry per the IG G.13 annotation requirements.

Required Vendor Information

The vendor **shall** provide evidence that their implementation complies with the requirements of **SP 800-135rev1** and of the documentation requirements of this Implementation Guidance. This **shall** be accomplished by documentation and code review.

Required Test Procedures

The tester **shall** review the vendor's evidence demonstrating that their implementation conforms to the specifications specified above. This **shall** be accomplished by documentation and code review.

Additional Comments

No self-tests are required to claim *vendor affirmation* to **SP 800-135rev1**.

W.13 Listing of DES Implementations

Applicable Levels:	*All*
Original Publishing Date:	*11/23/2005*
Date Withdrawn:	*12/21/2016*
Last Modified Date:	*01/16/2008*
Relevant Assertions:	*AS.01.12*
Relevant Test Requirements:	*TE01.12.01*
Relevant Vendor Requirements:	*VE.01.12.01*

Background

DEPARTMENT OF COMMERCE
National Institute of Standards and Technology
[Docket No. 040602169-5002-02]

Announcing Approval of the Withdrawal of Federal Information Processing Standard (FIPS) 46-3, Data Encryption Standard (DES); FIPS 74, Guidelines for Implementing and Using the NBS Data Encryption Standard; and FIPS 81, DES Modes of Operation.

Question/Problem

With the withdrawal of the DES cryptographic algorithm, how does the DES and DES MAC algorithms get listed on the FIPS 140-2 validation certificate?

Resolution

The DES transition period ended on May 19, 2007. DES and DES MAC are no longer approved security functions and **shall** be listed on the FIPS 140-2 certificate as non-approved algorithms.

Change Summary

New Guidance

- o 12/04/17: 9.13 Non-Reconfigurable Memory Integrity Test
- o 12/04/17: 9.12 Integrity Test Using Sampling
- o 12/04/17: A.15 Vendor Affirmation for the SP 800-185 Algorithms
- o 08/07/17: G.17 Remote Testing for Software Modules
- o 08/07/17: 1.23 Definition and Use of a non-Approved Security Function
- o 08/07/17: 9.11 Reducing the Number of Known Answer Tests
- o 08/07/17: A.14 Approved Modulus Sizes for RSA Digital Signature and Other Approved Public Key Algorithms
- o 05/10/17: A.13 SP 800-67rev1 Transition
- o 05/10/17: D.13 Elliptic Curves and the MODP Groups in Support of Industry Protocols
- o 11/15/16: 1.22 Module Count Definition
- o 11/15/16: 7.17 Zeroization of One Time Programmable (OTP) Memory
- o 11/15/16: A.12 Requirements for Vendor Affirmation to the Addendum to SP 800-38A
- o 08/01/16: A.11 The Use and the Testing Requirements for the Family of Functions defined in FIPS 202
- o 06/17/16: A.10 Requirements for Vendor Affirmation to SP 800-38G
- o 05/10/16: G.16 Requesting an Invoice Before Report Submission
- o 12/28/15: A.9 XTS-AES Key Generation Requirements
- o 08/07/15: 7.14 Entropy Caveats
- o 08/07/15: 7.15 Entropy Assessment
- o 08/07/15: 7.16 Acceptable Algorithms for Protecting Stored keys and CSPs
- o 08/07/15: D.1-rev2 CAVP Requirements for Vendor Affirmation of SP 800-56A-rev2
- o 08/07/15: D.12 Requirements for Vendor Affirmation to SP 800-133
- o 03/02/15: 1.21 Process Algorithm Accelerators (PAA)
- o 03/02/15: 1.20 Sub-Chip Cryptographic Subsystems
- o 03/02/15: A.8 Use of HMAC-SHA-1-96 and Truncated HMAC
- o 07/25/13: 3.5 Documentation Requirements for Cryptographic Module Services
- o 07/25/13: 9.9 Pair-Wise Consistency Self-Test When Generating a Key Pair
- o 07/25/13: 9.10 Power-Up Tests for Software Module Libraries
- o 07/25/13: D.11 References to the Support of Industry Protocols
- o 05/02/12: 9.8 Continuous Random Number Generator Tests
- o 05/02/12: 7.13 Cryptographic Key Strength Modified by an Entropy Estimate
- o 05/02/12: 7.12 Key Generation for RSA Signature Algorithm
- o 05/02/12: 7.11 Definition of an NDRNG
- o 05/02/12: 3.4 Multi-Operator Authentication

- o 05/02/12: 3.3 Authentication Mechanisms for Software Modules

- o 04/23/12: D.10 Requirements for Vendor Affirmation of SP 800-56C

- o 04/23/12: D.9 Key Transport Methods

- o 04/23/12: D.8 Key Agreement Methods

- o 04/23/12: 1.19 non-approved Mode of Operation

- o 04/23/12: 1.18 PIV Reference

- o 04/23/12: G.15 Validating the Transition from FIPS 186-2 to FIPS 186-3

- o 04/23/12: G.14 Validation of Transitioning Cryptographic Algorithms and Key Lengths

- o 07/15/11: 11.1 Mitigation of Other Attacks

- o 07/15/11: D.4 Requirements for Vendor Affirmation of SP 800-56B

- o 07/15/11: D.5 Requirements for Vendor Affirmation of SP 800-108

- o 07/15/11: D.6 Requirements for Vendor Affirmation of SP 800-132

- o 07/15/11: D.7 Requirements for Vendor Affirmation of SP 800-135

- o 12/23/10: 1.16 Software Module

- o 12/23/10: 1.17 Firmware Module

- o 12/23/10: 2.1 Trusted Path

- o 12/23/10: 5.5 Physical Security Level 3 Augmented with EFP/EFT

- o 12/23/10: 9.7 Software/Firmware Load Test

- o 12/23/10: 14.5 Critical Security Parameters for the SP 800-90 DRBGs

- o 01/27/10: 5.4 Level 3: Hard Coating Test Methods

- o 01/27/10: 14.4 Operator Applied Security Appliances

- o 01/27/10: A.7 CAVP Requirements for Vendor Affirmation of NIST SP800-38E

- o 10/22/09: 7.10 Using the SP 800-108 KDFs in FIPS Mode

- o 10/21/09: 9.6 Self-Tests When Implementing the SP 800-56A Schemes

- o 10/21/09: D.3 Assurance of the Validity of a Public Key for Key Establishment

- o 07/07/09: 1.15 CAVP Requirements for Vendor Affirmation of FIPS 186-3 Digital Signature Standard

- o 04/01/09: 3.2 Bypass Capability in Routers

- o 04/01/09: 9.5 Module Initialization during Power-Up

- o 03/24/09: 7.9 Procedural CSP Zeroization

- o 03/10/09: 1.14 Key/IV Pair Uniqueness Requirements from SP 800-38D

- o 03/10/09: 5.3 Physical Security Assumptions

- o 03/10/09: 7.8 Key Generation Methods Allowed in FIPS Mode

- o 01/24/08: 7.7 Key Establishment and Key Entry and Output

- o 12/18/07: 1.13 CAVP Requirements for Vendor Affirmation of SP 800-38D

- o 11/16/07: 7.6 RNGs: Seeds, Seed Keys and Date/Time Vectors

- o 07/03/07: 14.3 Logical Diagram for Software, Firmware and Hybrid Modules

- o 06/28/07: G.13 Instructions for completing a FIPS 140-2 Validation Certificate

- o 06/21/07: 1.11 CAVP Requirements for Vendor Affirmation of SP 800-56A

- o 06/21/07: 1.12 CAVP Requirements for Vendor Affirmation of SP 800-90

- o 01/26/07: G.12 Post-Validation Inquiries

- o 01/25/07: 1.10 Vendor Affirmation of Cryptographic Security Methods

Modified Guidance

- o 05/25/18: IG G.8 Revalidation Requirements – Removed the "2 year" limitation on 3sub revalidations, which stated that modules on the historical list could not be submitted as a 3sub if the module's sunset date exceeded 2 years. Now, modules that are *Active* or *Historical* are eligible for scenario 3 revalidation without this limitation.

- o 05/25/18: IG 9.11 Reducing the Number of Known Answer Tests – Changed the "type" of the parameter that "remembers" that self-tests were run successfully on a specific environment, from a CSP, to something that is treated the same as a public key, in which case the integrity of this parameter is assured by the module.

- o 03/27/18: IG G.8 Revalidation Requirements - Updated to add Alternative Scenario 3A (allowing vendors to submit module revalidations based on CVE patches).

- o 03/27/18: IG G.13 - Instructions for Validation Information Formatting - Updated to add clarification on how to document the binding module algorithm certificate. The same rules that apply to an embedding module also applies to a binding module.

- o 03/27/18: IG 9.1 Known Answer Test for Keyed Hashing Algorithm – Updated to align with IG 9.4 and IG 9.11. Also, added clarification on HMAC self-testing with additional examples and comments.

- o 03/27/18: IG 9.2 Known Answer Test for Embedded Cryptographic Algorithms – Updated to align with IG 9.11. Also, removed obsolete material (such as self-testing the embedded algorithms by means of the RNG KATs where the RNGs are no longer approved).

- o 03/27/18: IG A.13 SP 800-67rev1 Transition – Updated to incorporate the latest requirements for the published SP 800-67rev2 standard; namely, a module has a limit of either 2^{20} or 2^{16} 64-bit data block encryptions with the same Triple-DES key (as opposed to 2^{32} or 2^{28} from SP 800-67rev1). The transition guidance is explained in this updated IG.

- o 01/19/18: G.13 Instructions for Validation Information Formatting – Removed non-SP-800-38F compliant key wrapping methods from the allowed algorithm listing per SP 800-131A transition. Added allowed non-SP-800-38F compliant key *unwrapping* examples.

- o 01/19/18: D.9 Key Transport Methods – Removed non-SP-800-38F compliant key wrapping methods from the allowed algorithm section per SP 800-131A transition. Added two additional comments for clarity on SP 800-131A transition and KTS implementations.

- o 12/04/17: A.9 XTS-AES Key Generation Requirements – added text requiring the CST testing lab to document in TE.01.12.01 of the Test Report how the module meets IG A.9.

- o 12/04/17: G.13 Instructions for Validation Information Formatting – added a caveat example when a module implements a DRBG but does not meet IG 7.14 and IG 7.15 requirements.

- o 12/04/17: A.5 Key/IV Pair Uniqueness Requirements from SP 800-38D – added bullet 4 in scenario 2 requiring the module to meet IG 7.15 for the strength of the IV.

- o 12/04/17: G.8 Revalidation Requirements - added notes about which scenarios should be included on the MIP list. Also updated scenario 2 to allow for modules on the Historical list to be validated via this scenario.

- o 12/04/17: Revised IG for grammatical and formatting inconsistencies.

- o 08/07/17: G.13 Instructions for Validation Information Formatting – added example for Truncated HMAC and moved virtual environment information to Section 10: Operational Environment.

- o 08/07/17: 3.1 Authorized Roles – added Additional Comment to explain an exception to the rule that prohibited a CSP modification by a service called from an unauthenticated role at Levels 2+.

- o 08/07/17: 14.1 Level of Detail When Reporting Cryptographic Services – clarified the requirement that all security services were listed in the Security Policy.

- o 08/07/17: 14.4 Operator Applied Security Appliances – minor updates including addressing the confusing dates in the header.

- o 08/07/17: 14.5 Critical Security Parameters for the SP 800-90 DRBGs – updated to remove the text that used to discuss the relationship between this IG and AS07.13 which is not relevant here. Also, made it clear that the FIPS 140-2 requirements on the "seed key" parameter was no longer applicable as none of the approved DRBGs used this parameter.

- o 08/07/17: A.2. Use of non-NIST-Recommended Elliptic Curves – updated to allow new key sizes for RSA, DSA, and the corresponding key-establishment algorithms.

- o 08/07/17: A.5 Key/IV Pair Uniqueness Requirements from SP 800-38D – updated to include MacSec.

- o 08/07/17: A.11 The Use and the testing Requirements for the Family of Functions defined in FIPS 202 – updated to reflect testing available in CAVS.

- o 08/07/17: A.13 SP 800-67rev1 Transition – added transition dates that were originally only published in an email.

- o 08/07/17: D.2 Acceptable Key Establishment Protocols – modified key entry description and added pre-loading of a key.

- o 06/13/17: 9.9 Pair-Wise Consistency Self-Test When Generating a Key Pair – the scope is limited to the pair-wise consistency tests for keys used in RSA signature and RSA key transport schemes and removed "allowed" provision.

- o 05/10/17: G.8 Revalidation Requirements – added definition for scenario 2.

- o 05/10/17: G.13 Validation Certificate Formatting – removed non-approved algorithms from the validation certificate, added examples for key establishment and included formatting instructions for virtual environments.

- o 05/10/17: G.14 Validation of Transitioning Cryptographic Algorithms and Key Lengths,
  7.5 Strength of Key Establishment Methods,
  A.11 The Use and the Testing Requirements for the Family of Functions defined in FIPS 202,
  D.8 Key Agreement Methods,
  D.11 References to the Support of Industry Protocols
  removed references to certificate formatting for non-approved algorithms.

- o 05/10/17: 3.1 Authorized Roles – addressed relationship between authorized roles and operator authentication.

- o 05/10/17: 3.4 Multi-Operator Authentication – resolve a conflict between IG 3.1 and IG 3.4.

- o 05/10/17: A.8 Use of a Truncated HMAC – updated text, clarified examples and incorporated SP 800-107rev1 for all uses of a message authentication code.

- o 05/10/17: D.9 Key Transport Methods – updated to explain that all approved key transport schemes shall use the KTS acronym and to allow an unwrapping of a key past the 2017 transition deadline.

- o 04/25/17: D.12 Requirements for Vendor Affirmation to SP 800-133 – clarified some of the provisions.

- o 04/17/17: 1.21 Processor Algorithm Accelerators (PAA) & Processor Algorithm Implementation (PAI) – add PAI where an accelerated function to support cryptographic algorithms is deemed to be the complete cryptographic algorithm and updated the list of known PAAs and PAIs.

- o 02/06/17: 1.20 Sub-Chip Cryptographic Subsystems – updated 1.20 and 7.7 to resolve the asymmetric treatment of CM software and CM hardware.

- o 02/06/17: 7.7 Key Establishment and Key Entry and Output – updated 1.20 and 7.7 to resolve the asymmetric treatment of CM software and CM hardware.

- o 02/06/17: D.11 References to the Support of Industry Protocols – clarified items 2 and 3.

- o 12/21/16: G.8 Revalidation Requirements – incorporated sunset validation policy and removed scenario 2.

- o 12/21/16: 1.8 - moved to W.13

- o 11/15/16: G.13 Instructions for Validation Information Formatting – Added example for IG A.12, added HMAC-SHA-1-96 examples and corrected a KAS example.

- o 11/15/16: 3.5 Documentation Requirements for Cryptographic Module Services – Updated to reduce the burden for documentation for some modules (e.g. IPSec VPN or PKCS#11 module).

- o 08/01/16: G.9 FSM, Security Policy, User Guidance and Crypto Officer Guidance Documentation – added source code information is considered vendor-provided documentation and may be used in the FSM and/or Security Policy.

- o 08/01/16: G.13 Instructions for Validation Information Formatting – FIPS algorithms shall be listed in alphabetical order; Other algorithms shall be listed with allowed algorithms in alphabetical order first, followed by non-approved algorithms in alphabetical order.

- o 05/10/16: G.2 Completion of a Test Report: Information that must be provided to NIST and CSE - updated the file types, removed requirement for Report Overview.

- o 05/10/16: G.13 Instructions for Validation Information Formatting – updated FIPS approved algorithm examples and non-approved random number generator description.

- o 05/10/16: 14.5 Critical Security Parameters for the SP 800-90 DRBGs – removed Dual_EC_DRBG mechanism.

- o 05/10/16: D.4 Requirements for Vendor Affirmation for SP 800-56B – updated RBG to DRBG, removed hash algorithm examples, fixed fonts for the equations and removed the exception phrase.

- o 05/10/16: 7.6 – moved to W.5

- o 05/10/16: 7.11 – moved to W.6

- o 05/10/16: A.4 – moved to W.7

- o 05/10/16: A.6 – moved to W.8

- o 05/10/16: A.7 – moved to W.9

- o 05/10/16: D.1 – moved to W.10

- o 05/10/16: D.5 – moved to W.11

- o 05/10/16: D.7 – moved to W.12

- o 01/11/16: G.14 Validation of Transitioning Cryptographic Algorithms and Key Lengths – update references to FIPS 186-4, define legacy use of 186-2 and other post RNG transition changes

- o 01/11/16: 7.5 Strength of Key Establishment Methods - update references to FIPS 186-4 and other post RNG transition changes

- o 01/11/16: 7.8 Key Generation Methods Allowed in FIPS Mode - update references to FIPS 186-4 and other post RNG transition changes

- o 01/11/16: 7.12 Key Generation for RSA Signature Algorithm - update references to FIPS 186-4 and other post RNG transition changes

- o 01/11/16: D.4 Requirements for Vendor Affirmation of SP 800-56B- update references to FIPS 186-4 and other post RNG transition changes

- o 01/11/16: C.1 – moved to W.3

- o 01/11/16: C.2 – moved to W.4

- o 01/04/16: G.15 - moved to W.2

- o 01/04/16: A.9 XTS-AES Key Generation Requirements – minor editorial change of the last sentence in Additional Comments.

- o 12/22/15: 9.8 Continuous Random Number Generator Tests – introduced advanced options for continuous random number generation testing.

- o 11/20/15: G.5 Maintaining validation compliance of software or firmware cryptographic modules – fixed a discrepancy in the wording of user porting rules. Now user affirmation is similar to that of vendors so that validation is only user-affirmed and does not imply a CMVP endorsement.

- o 11/13/15: G.5 Maintaining validation compliance of software or firmware cryptographic modules – fixed a typo/poor text formatting - removed d) in 1) as it is just a continuation of c).

- o 11/12/15: 7.16 Acceptable Algorithms for Protecting Stored Keys and CSPs – Fixed a typo – misspelled Tripe-DES.

- o 11/12/15: G.5 Maintaining validation compliance of software or firmware cryptographic modules – fixed a logically inconsistent wording related to porting modules to a new untested operational environment.

- o 11/12/15: 7.15 Entropy Assessment – introduced a transition period for third-party hardware entropy sources that cannot meet all documentation and test requirements.

- o 09/15/15: 1.20 Sub-Chip Cryptographic Subsystems – Updated with multiple disjoint sub-chip subsystems and refinements of testing and documentation requirements.

- o 08/07/15: 7.13 – moved to W.1.

- o 08/07/15: A.5 Key/IV Pair Uniqueness Requirements from SP 800-38D – Allow IPSec- and TLS 1.2-style of IV generation for AES-GCM cipher suites.

- o 08/07/15: D.9 Key Transport Methods – Updated with more SP 800-38F examples.

- o 08/07/15: G.1 Request for Guidance from the CMVP and CAVP – Updated contacts and set in writing requirement for requests.

- o 08/07/15: G.2 Completion of a test report: Information that must be provided to NIST/CSE –Changed CSEC to CSE.

- o 08/07/15: G.7 Relationships Among Vendors, Laboratories, and NIST/CSE –Changed CSEC to CSE.

- o 08/07/15: G.9 FSM, Security Policy, User Guidance and Security Officer Documentation –Changed CSEC to CSE.

- o 08/07/15: G.12 Post-Validation Inquiries –Changed CSEC to CSE.

- o 08/07/15: G.13 Instructions for Validation Information Formatting – Updated with more examples.

- o 01/15/15: A.5 Key/IV Pair Uniqueness Requirements from SP 800-38D – Allow TLS 1.2-style of IV generation for AES-GCM cipher suites.

- o 04/25/14: 9.10 Power-Up Tests for Software Module Libraries – Editorial changes for additional clarity.

- o 01/17/14: G.15 Validating the Transition from FIPS 186-2 to FIPS 186-4 – Editorial change.

- o 01/17/14: 7.13 Cryptographic Key Strength Modified by an Entropy Estimate – Changed the minimum entropy requirement based on SP 800-131A transition effective 01-01-2014.

- o 01/15/14: G.13 Instructions for Validation Information Formatting – Removed incorrect examples based on SP 800-131A transition effective 01-01-2014.

- o 01/08/14: G.13 Instructions for Validation Information Formatting – Updated and corrected examples based on SP 800-131A transition effective 01-01-2014.

- o 01/07/14: G.2 Completion of a test report: Information that must be provided to NIST and CSEC – Removed old report submission process information

- o 01/07/14: G.3 Partial Validations and Not Applicable Areas of FIPS 140-2 – Minor editorial updates.

- o 01/07/14: G.4 Design and testing of cryptographic modules – Updated "other associated documents" references.

- o 01/07/14: G.13 Instructions for Validation Information Formatting – Updated examples based on SP 800-131A transition effective 01-01-2014.

- o 01/07/14: G.14 Validation of Transitioning Cryptographic Algorithms and Key Lengths – Updated based on SP 800-131A transition effective 01-01-2014.

- o 01/07/14: G.15 Validating the Transition from FIPS 186-2 to FIPS 186-4 – Updated based on SP 800-131A transition effective 01-01-2014.

- o 07/25/13: D.8 Key Agreement Methods – Resolution section has been updated.

- o 07/25/13: D.9 Key Transport Methods – Resolution section has been updated.

- o 06/07/13: G.8 Revalidation Requirements – Added Alternative Scenarios 1A and 1B.

- o 12/21/12: G.5 Maintaining validation compliance of software or firmware cryptographic modules – Included reference to the impact to the generated key strength assurance when porting, and vendor Security Policy updates.

- o 12/21/12: G.13 Instructions for Validation Information Formatting – For all embodiments, the OE shall be specified on the validation entry.

- o 12/21/12: G.14 Validation of Transitioning Cryptographic Algorithms and Key Lengths – Addressed two-key Triple-DES requirements.

- o 12/21/12: D.8 Key Agreement Methods – IG updated to address SP 800-135rev1

- o 06/29/12: 7.7 Key Establishment and Key Entry and Output - References to key encryption changed to reference Key Establishment methods (e.g. Key Transport and Key Agreement).

- o 06/20/12: G.2 Completion of a test report: Information that must be provided to NIST and CSEC – Added transition date for report submissions using CRYPTIK integrated review process.

- o 06/20/12: 1.19 non-Approved Mode of Operation – Re-written to associate with existing clauses in FIPS 140-2 and this Implementation Guidance.

- o 06/20/12: 7.12 Key Generation for RSA Signature Algorithm – Added Transition End Date.

- o 06/20/12: 9.4 Known Answer Tests for Cryptographic Algorithms – Added Transition End Date in reference to NOTE4.

- o 05/02/12: 1.19 non-approved Mode of Operation – Modified resolution when annotating non-approved services.

- o 05/02/12: 1.7 Multiple Approved Modes of Operation – Modified resolution and additional comments text.

- o 05/02/12: 1.2 FIPS Approved Mode of Operation – Modified resolution and additional comments text.

- o 05/02/12: G.13 Instructions for Validation Information Formatting – Added annotation note regarding EFP/EFT when Section 4.5 is Level 3.

- o 04/23/12: D.7 Requirements for Vendor Affirmation of SP 800-135rev1 – Transition end date of 06/23/2012 added and updated reference to SP 800-135 Revision 1.

- o 04/23/12: D.6 Requirements for Vendor Affirmation of SP 800-132 – Algorithm validation acronym reference updated.

- o 04/23/12: D.5 Requirements for Vendor Affirmation of SP 800-108 – Transition end date of 06/23/2012 added and algorithm validation acronym reference updated.

- o 04/23/12: D.2 Acceptable Key Establishment Protocols – Completely revised as an umbrella IG for approved and allowed key establishment methods.

- o 04/23/12: A.3 Vendor Affirmation of Cryptographic Security Methods – Removed caveat examples and replaced with referenced to IG G.13.

- o 04/23/12: 9.6 Self-Tests When Implementing the SP 800-56A Schemes – IG expanded and clarifications added.

- o 04/23/12: 9.4 Known Answer Tests for Cryptographic Algorithms – IG revised and expanded.

- o 04/23/12: G.13 Instructions for Validation Information Formatting – Updated 2nd, 3rd, 4th, 8th, 9th and 10th bullets.

- o 04/23/12: G.2 Completion of a test report: Information that must be provided to NIST and CSEC – Added clause to 3rd bullet regarding physical security test evidence traceability to DTR. Added 5th bullet regarding table templates.

- o 04/23/12: G.1 Request for Guidance from the CMVP and CAVP – Updated CSEC contact.

- o 07/15/11: G.3 Partial Validations and Not Applicable Areas of FIPS 140-2 – Modified in regard to new IG 11.1.

- o 07/15/11: G.6 Modules with both a FIPS mode and a non-FIPS mode – Clarification that all implemented algorithms shall be referenced on the validation certificate.

- o 07/15/11: G.8 Revalidation Requirements – Added security policy requirements for revalidation Scenarios 1 and 4.

- o 07/15/11: G.13 Instructions for Validation Information Formatting – Added examples for CVL and KTS.

- o 07/15/11: 1.4 Binding of Cryptographic Algorithm Validation Certificates – Added examples of an operational environment change.

- o 07/15/11: D.1 CAVP Requirements for Vendor Affirmation of SP 800-56A – Modified the testing for primitives.

- o 07/15/11: D.2 Acceptable Key Establishment Protocols – Modified the transition text and key agreement guidance.

- o 05/11/11: G.13 Instructions for Validation Information Formatting – Corrected format of examples.

- o 03/03/11: G.2 Completion of a test report: Information that must be provided to NIST and CSEC – Changes relative to the release of CRYPTIK v8.6b

- o 03/03/11: G.13 Instructions for Validation Information Formatting – Changes relative to the release of CRYPTIK v8.6b

- o 03/03/11: A.2 Use of Non-NIST-Recommended Asymmetric Key Sizes and Elliptic Curves – Updated for consistency with recent standards.

- o 03/03/11: A.6 CAVP Requirements for Vendor Affirmation of FIPS 186-3 Digital Signature Standard – Transition end date for FIPS 186-3 RSA is defined.

- o 01/03/11: D.2 Acceptable Key Establishment Protocols – Change NIST CSD CT Group Contact to Mr. Tim Polk.

- o 12/23/10: 9.6 Self-Tests When Implementing the SP 800-56A Schemes – Requirements changed.

- o 08/02/10 G.8 Revalidation Requirements – For scenarios 1 and 4 added clarification on required submission documents sent to the CMVP.

- o 06/15/10: 5.4 Level 3: Hard Coating Test Methods – Removed reference to environmental conditions other than temperature and added Security Policy requirements.

- o 06/10/10: G.2 Completion of a test report: Information that must be provided to NIST and CSEC – Updated submission and billing information requirements.

- o 06/10/10: G.13 Instructions for completing a FIPS 140-2 Validation Certificate – Additional caveat examples.

- o 06/10/10: 1.3 Firmware Designation – Updated platform versioning requirements if physical security is Level 2, 3 or 4.

- o 06/10/10: 5.4 Level 3: Hard Coating Test Methods – Modified temperature testing limits and removed testing methods using solvents.

- o 06/10/10: 7.5 Strength of Key Establishment Methods – Added reference to draft SP 800-131.

- o 06/10/10: A.6 CAVP Requirements for Vendor Affirmation of FIPS 186-3 Digital Signature Standard – Updated with transition end date for ECDSA.

- o 04/09/10: A.6 CAVP Requirements for Vendor Affirmation of FIPS 186-3 Digital Signature Standard – Updated with transition end date.

- o 04/09/10 A.7 CAVP Requirements for Vendor Affirmation of NIST SP800-38E – Updated with transition end date.

- o 03/19/10: 1.9 Definition and Requirements of a Hybrid Cryptographic Module - Updated the annotation for software-hybrid and, firmware-hybrid modules.

- o 03/19/10: G.13 Instructions for completing a FIPS 140-2 Validation Certificate - Added examples for software-hybrid and firmware-hybrid modules.

- o 01/27/10: 7.2 Use of IEEE 802.11i Key Derivation Protocols
  Guidance updated in regard to references to SP 800-56A and IG 7.10.

- o 01/27/10: G.13 Instructions for completing a FIPS 140-2 Validation Certificate
  Removed PIV Middleware reference. Added XTS-AES annotation reference.

- o 10/21/09: To align Implementation Guidance that is associated with underlying algorithmic standards referenced in FIPS 140-2 Annexes A, C and D, the following algorithm specific IGs have been moved to new IG Annex sections:

  Moved IG 1.5 to IG A.1, IG 1.6 to IG A.2, IG 1.10 to A.3, IG 1.11 to IG D.1, IG 1.12 t IG C.1, IG 1.13-15 to IG A..4-6, IG 7.1 to IG D.2 and IG 7.3 to IG C.2

- o 10/20/09: G.1 Request for Guidance from the CMVP and CAVP – Updated contact information.

- o 10/20/09: G.2 Completion of a test report: Information that must be provided to NIST and CSEC – Minor editorial changes.

- o 10/20/09: G.13 Instructions for completing a FIPS 140-2 Validation Certificate – Added FIPS 186-3 and SP 800-56A annotation examples.

- o 10/20/09: 1.11 CAVP Requirements for Vendor Affirmation of SP 800-56A – Added reference to the annotation requirements in IG G.13.

- o 10/20/09: 1.15 CAVP Requirements for Vendor Affirmation of FIPS 186-3 Digital Signature Standard – Added transition information and reference to the annotation requirements in IG G.13.

- o 10/20/09: 7.1 Acceptable Key Establishment Protocols – Added transition information.

- o 08/31/09: 7.1 Acceptable Key Establishment Protocols – Added references to DTLS.

- o 08/04/09: G.13 Instructions for completing a FIPS 140-2 Validation Certificate – Added additional certificate annotation examples.

- o 08/04/09: 1.10 Vendor Affirmation of Cryptographic Security Methods – Additional certificate annotation examples.

- o 08/04/09: 1.15 CAVP Requirements for Vendor Affirmation of FIPS 186-3 Digital Signature Standard – Certificate annotation examples.

- o 08/04/09: 7.1 Acceptable Key Establishment Protocols – For Key Agreement; removed the KDF specified in the SRTP protocol (IETF RFC 3711). For Key Transport; added reference to EAP-FAST and PEAP-TLS.

- o 03/10/09: G.1 Request for Guidance from the CMVP – Updated NIST POC.

- o 03/10/09: G.5 Maintaining validation compliance of software or firmware cryptographic modules – Updated references to firmware and hybrid modules.

- o 03/10/09: G.13 Instructions for completing a FIPS 140-2 Validation Certificate – Updated examples.

- o 03/10/09: 1.9 Definition and Requirements of a Hybrid Cryptographic Module – Updated to include hybrid firmware modules.

- o 03/10/09: 7.1 Acceptable Key Establishment Protocols – For Key Agreement; added the KDF specified in the SRTP protocol (IETF RFC 3711) is allowed only for use as part of the SRTP key derivation protocol. For Key Transport; wrapping a key using the GDOI Group Key Management Protocol described in the IETF RFC 3547.

- o 07/09/08: 1.10 Vendor Affirmation of Cryptographic Security Methods – Updated examples of certificate algorithm notation.

- o 06/25/08: G.13 Instructions for completing a FIPS 140-2 Validation Certificate – Updated file naming convention syntax

- o 05/22/08: G.13 Instructions for completing a FIPS 140-2 Validation Certificate – Updated reference for symmetric key wrapping annotation

- o 02/07/08: 7.1 Acceptable Key Establishment Protocols – Updated AES Key Wrap URL.

- o 01/24/08: G.2 Completion of a test report: Information that must be provided to NIST and CSE – Added reference to CMVP comments document.

- o 01/24/08 G.8 Revalidation Requirements – Added reference to the CMVP FAQ in change scenario 1.

- o 01/16/08: G.13 Instructions for completing a FIPS 140-2 Validation Certificate – Added reference for listing multiple operating systems, and reference for symmetric key wrapping annotation.

- o 01/16/08: 1.8 Listing of DES Implementations – Updated to reflect the ending of the DES transition period.

- o 01/16/08: 7.1 Acceptable Key Establishment Protocols

- o 01/16/08: 9.4 Cryptographic Algorithm Tests for SHS Algorithms and Higher Cryptographic Algorithms Using SHS Algorithms – Added RSA KAT requirements regarding the relationship of the exponents.

- o 11/08/07: G.2 Completion of a test report: Information that must be provided to NIST and CSE – Added clarification on output type of draft certificate.

- o 10/18/07: Updated links

- o 07/26/07: Minor editorial updates.

- o 06/26/07: 7.1 Acceptable Key Establishment Protocols – Updated to reflect the publishing of SP 800-56A.

- o 06/26/07: G.8 Revalidation Requirements – Additional guidelines for determining <30% change for Scenario 3.

- o 06/22/07: G.2 Completion of a test report: Information that must be provided to NIST and CSE - editorial changes for clarification.

- o 06/22/07: G.8 Revalidation Requirements - editorial changes for clarification.

- o 06/14/07: 3.1 Authorized Roles

- o   03/19/07: Updated references to revision of SP 800-57
- o   02/26/07: 1.6 Use of Non-NIST-Recommended Asymmetric Key Sizes and Elliptic Curves
- o   02/23/07: 7.4 Zeroization of Power-Up Test Keys
- o   01/25/07: G.8 Revalidation Requirements
- o   01/25/07: 7.5 Strength of Key Establishment Methods

End of Document