The 186-4 RSA Validation System (RSA2VS)

Updated: July 8, 2014
Previous: April 16, 2014
Original: March 03, 2011

Sharon S. Keller

National Institute of Standards and Technology

Information Technology Laboratory

Computer Security Division

Cryptographic Algorithm Validation Program

Update Log

7/8/14

- Fix typo in Section 6.2.2 – instead of sentences saying “if i is not a prime”, changed to “if p is not a prime” and “if q is not a prime”.

4/16/14

- Update Section 6.2 to indicate the IUT supplies all the values for Key Generation.

3/18/14

- Update Key Gen section to reflect change in way testing performed. Now the IUT supplies all the values instead of the CAVS supplying some initial values first.
- Moved the 186-2 Legacy Signature Verifications tests to this document from the 186-2 RSAVS document

9/5/13

- Updated References to FIPS 186-4 throughout document.

10/25/12

- Section 6.2.1 Clarified what the IUT is responsible for providing for four of the methods of Key Generation that are specified in FIPS186-4 Appendices B.3.2, B.3.4, B.3.5, and B.3.6.

3/22/12

- Section 6.2.2 Validation Test for Key Generation Methods Specified in Appendices B.3.3
    - Distinguished the testing requirements for IUTs supporting only fixed *e* value(s) vs. IUTs supporting random *e* values.
- Section 6.2.2.1 The Known Answer Test for B.3.3 Probably Primes
    - Clarified that this test is only for use with IUTs supporting random *e* values.
- Section 6.2.2.2 The KeyGen_RandomProbablyPrime3_3 Test
    - Clarified that this test is required for both IUTs supporting random *e* values and IUTs supporting fixed *e* values.

6/29/11

- Section 2
    - Removed second paragraph on testing of prerequisites.
- Section 4
    - Removed DRBG, DRBGVS, SHA and SHAVS from list of abbreviations.
- Section 6
    - Removed second paragraph on testing of prerequisites.

TABLE OF CONTENTS

1 Introduction

The 186-4 RSA Validation System (RSA2VS) specifies the procedures involved in validating implementations of public key cryptography based on the RSA algorithm. This document deals with three variations of the RSA algorithm which are referenced in FIPS186-4, *Digital Signature Standard (DSS)* [1]. They are:

- RSA algorithm specified in American National Standard (ANS) X9.31 [2], and
- Two signature schemes with appendix specified in *Public Key Cryptography Standards(PKCS) #1 v2.1: RSA Cryptography Standard-2002* [3]. These two signature schemes with appendix are
    - RSASSA-PSS, and
    - RSASSA-PKCS1-v1_5.

The 186-4RSAVS is designed to perform automated testing on Implementations Under Test (IUTs). It provides the basic design and configuration of the 186-4RSAVS. Included are the specifications for testing the individual RSA components of the IUT. These components include:

- Key Generation
- Signature Generation
- Signature Verification
- FIPS 186-2 Legacy testing for Signature Verification

This document defines the purpose, the design philosophy, and the high-level description of the validation process for RSA. The requirements and administrative procedures to be followed by those seeking formal validation of an implementation of 186-4RSA are presented. The requirements described include the specification of the data communicated between the IUT and the 186-4RSAVS, the details of the tests that the IUT must pass for formal validation, and general instruction for interfacing with the 186-4RSAVS.

2 Scope

This document specifies the tests required to validate IUTs for conformance to the 186-4RSA as specified in [1]. When applied to IUTs that implement any of the three different algorithm variations of the RSA, the 186-4RSAVS provides testing to determine the correctness of the algorithm components contained in the implementation. The 186-4RSAVS is composed of three different tests, one to validate each of the various algorithm components and the 186-2 Legacy tests for Signature Verification. In addition to determining conformance to the cryptographic specifications, the 186-

4RSAVS is structured to detect implementation flaws including pointer problems, insufficient allocation of space, improper error handling, and incorrect behavior of the RSA implementation.

3 Conformance

The successful completion of the tests contained within the 186-4RSAVS is required to be validated as conforming to the RSA. Testing for the cryptographic module in which the RSA is implemented is defined in FIPS PUB 140-2, *Security Requirements for Cryptographic Modules*.[4]

4 Definitions and Abbreviations

DEFINITION	**MEANING**
CST laboratory	Cryptographic and Security Testing laboratory that operates the 186-4RSAVS
IUT	Implementation Under Test
PKCS	Public Key Cryptography Standards

5 Design Philosophy of the 186-4RSA Validation System

The 186-4RSAVS is designed to test conformance to RSA rather than provide a measure of a product's security. The validation tests are designed to assist in the detection of accidental implementation errors, and are not designed to detect intentional attempts to misrepresent conformance. Thus, validation should not be interpreted as an evaluation or endorsement of overall product security.

The 186-4RSAVS has the following design philosophy:

1. The 186-4RSAVS is designed to allow the testing of an IUT at locations remote to the 186-4RSAVS. The 186-4RSAVS and the IUT communicate data via *REQUEST* and *RESPONSE* files.

2. The testing performed within the 186-4RSAVS utilizes statistical sampling (i.e., only a small number of the possible cases are tested); hence, the successful validation of a device does not imply 100% conformance with the standard.

6 186-4RSAVS Tests

The 186-4RSAVS for RSA provides conformance testing for three of the components of the algorithm, as well as testing for apparent implementation errors. The components tested are key generation as specified in FIPS186-4, signature generation and signature verification. There are 5 different methods used for key generation which are discussed in FIPS 186-4, Appendix B.3.

The 186-4RSAVS also contains 186-2 Legacy testing for the Signature Verification

6.1 Configuration Information

To initiate the validation process of the 186-4RSAVS, a vendor submits an application to an accredited laboratory requesting the validation of its implementation of 186-4 RSA. The vendor's implementation is referred to as the Implementation Under Test (IUT). The request for validation includes background information describing the IUT along with information needed by the 186-4RSAVS to perform the specific tests. More specifically, the request for validation includes:

1. Vendor Name;

2. Product Name;

3. Product Version;

4. Implementation in software, firmware, or hardware;

5. Processor and Operating System with which the IUT was tested if the IUT is implemented in software or firmware;

6. Brief description of the IUT or the product/product family in which the IUT is implemented by the vendor (2-3 sentences); and

7. For RSA from X9.31 and RSASSA_PKCS1.5, the modulus size-SHA size combination(s) supported by the IUT.

8. For RSASSA-PSS implementations, the modulus size-SHA size-SALT length and, if applicable, the SALT value combination(s) supported by the IUT.

9. For IUT's implementing Key Generation,

    a. The prime methods supported by the IUT. Each prime method requires some combination of the SHA sizes supported, the modulus sizes supported and/or the table used to determine the number of iterations of the Miller-Rabin routine.

6.2 186-4 RSA Key Generation Test

An implementation of the FIPS 186-4 RSA may generate the key components used in the 186-4 RSA algorithm's signature generation and verification processes. This option tests the ability of an IUT to produce correct values for these components.

FIPS186-4 Key Pair generation specifies 5 ways (or methods) to generate the primes *p* and *q*. These include:

1. Random Primes:
    - Provable primes (Appendix B.3.2)
    - Probable primes (Appendix B.3.3)
2. Primes with Conditions:
    - Primes p1, p2, q1,q2, p and q shall all be provable primes (Appendix B.3.4)
    - Primes p1, p2, q1, and q2 shall be provable primes and p and q shall be probable primes (Appendix B.3.5)
    - Primes p1, p2, q1,q2, p and q shall all be probable primes (Appendix B.3.6)

The Random Provable primes (Appendix B.3.2), and all the Primes with conditions (Appendix B.3.4, B.3.5, and B.3.6) can be tested in the same manner because each of these begin with a starting random number and calculate the p and q values from this value. The 186-4 RSAVS instructs the IUT to generate intermediate values and the p, q, n, and d values. The RSAVS then validates the correctness of the values generated by the IUT.

The Random Probable primes (Appendix B.3.3) must be tested in a different way because this test generates different random numbers, not related to each other, until the number satisfies the "probably prime" requirements. The 186-4 RSAVS validation requires two tests for Random Probable primes. These include the Known Answer Test and the KeyGen_RandomProbablyPrime3_3 test.

6.2.1 Validation Test for Key Generation Methods Specified in Appendices B.3.2, B.3.4, B.3.5, and B.3.6

To test the key generation method specified in Appendices B.3.2, B.3.4, B.3.5, and B.3.6, the 186-4 RSAVS requests the IUT to supply 25 sets of values for each prime method/modulus/hash(if applicable)/Miller-Rabin Table used(if applicable) combination supported by the IUT. The 186-4 RSAVS verifies the received results by calculating the values based on the values used by the IUT. This will assure that the IUT can generate the primes using the key generation method specified.

The 186-4 RSAVS:

A. Creates a *REQUEST* file (Filename: KeyGen_186-3.req) containing:

1. A header consisting of:
    a. The CAVS version,
    b. The algorithm being tested and the product name,
    c. For each prime method selected (Appendices B.3.2, B.3.4, B.3.5, and B.3.6):
        i. The name of the prime method selected,
        ii. The modulus or modulus/hash combinations supported,
        iii. If applicable, the Miller-Rabin information; and
    d. The date generated.
2. For each prime method – modulus size – hash algorithm (if applicable) – Miller-Rabin table combination (if applicable) tested,
    a. a section header indicating the value for each the prime method, the modulus size, the hash algorithm (if applicable), and the Miller-Rabin table (if applicable).
    b. N = 25 indicating the number of entries the IUT is to send in the response file.

Note: The CST laboratory sends the REQUEST file to the IUT.

B. Creates a *FAX* file (Filename: KeyGen_186-3.fax) containing:

1. The information from the *REQUEST* file.

The IUT:

A. Produces N sets of values per prime method, the modulus size, the hash algorithm (if applicable), and the Miller-Rabin table (if applicable) combination supported by the IUT.
B. Creates a *RESPONSE* file (Filename: KeyGen_186-3.rsp) containing:

a. 25 sets of values per prime method, the modulus size, the hash algorithm (if applicable), and the Miller-Rabin table (if applicable) combination supported by the IUT. The values in each set differs for each prime method:
    i. Prime method: Provable Random primes (Appendix B.3.2)
        1. Public key, *e*,
        2. Seed
        3. The private prime factor *p*
        4. The private prime factor *q*
        5. The value of the modulus *n*
        6. The value of the private signature exponent *d.*

    ii. Prime method: Primes with Conditions where p1, p2, q1,q2, p and q shall all be provable primes (Appendix B.3.4)

1. Public key,*e*,
2. Seed,
3. Bitlen1 for length of p_1
4. Bitlen2 for length of p_2
5. Bitlen3 for length of p_3
6. Bitlen4 for length of p_4
7. The private prime factor *p*
8. The private prime factor *q*
9. The value of the modulus *n*
10. The value of the private signature exponent *d.*

iii. Prime method: Primes with Conditions where p1, p2, q1, and q2 shall be provable primes and p and q shall be probable primes (Appendix B.3.5)

1. Public key, e,
2. Seed,
3. Bitlen1 for length of p_1
4. The prime factor *p1*
5. Prime seed for *p2*
6. Bitlen2 for length of p_2
7. The prime factor *p2*
8. Prime seed for *q1*
9. Xp – the random number used in Step 3 of Appendix C.9 *Compute a Probable Prime Factor Based on Auxiliary Primes* to generate prime p
10. The private prime factor *p*
11. Bitlen3 for length of p_3
12. The prime factor *q1*
13. Prime seed for *q2*
14. Bitlen4 for length of p_4
15. The prime factor *q2*
16. Xq– the random number used in Step 3 of Appendix C.9 *Compute a Probable Prime Factor Based on Auxiliary Primes* to generate prime q
17. The private prime factor *q*
18. The value of the modulus *n*
19. The value of the private signature exponent *d* .

iv. Prime method: Primes with Conditions where p1, p2, q1,q2, p and q shall all be probable primes (Appendix B.3.6)

1. Public key, e,
2. Bitlen1 for length of Xp1

3. Xp1 – Random odd integer for Step 4.1 to calculate p_1
4. The prime factor *p1*
5. Bitlen2 for length of Xp2
6. Xp2 – Random odd integer for Step 4.1 to calculate p_2
7. The prime factor *p2*
8. Xp – the random number used in Step 3 of Appendix C.9 to generate prime p
9. The private prime factor *p*
10. Bitlen3 for length of Xq1
11. Xq1– Random odd integer for Step 4.1 to calculate q_1
12. The prime factor *q1*
13. Bitlen4 for length of Xq2
14. Xq2 – Random odd integer for Step 4.1 to calculate q_2
15. The prime factor *q2*
16. Xq – the random number used in Step 3 of Appendix C.9 to generate prime q
17. The private prime factor *q*
18. The value of the modulus *n*
19. The value of the private signature exponent *d*.

Note: The IUT sends the *RESPONSE* file to the CST laboratory for processing by the 186-4 RSAVS.

The 186-4 RSAVS:

A. Using the supplied values, the RSAVS computes all values and compares them to the values in the response file.
B. If all the values pass, the test is successful. If any of the values do not pass, the test will fail.

6.2.2 Validation Test for Key Generation Methods Specified in Appendices B.3.3

To test the key generation method specified in Appendices B.3.3, the IUT will perform either one or two tests depending on whether or not the IUT supports random *e* values. If the IUT supports random *e* values, the IUT will be instructed to perform both the Known Answer Test (Section 6.2.2.1) and the KeyGen_RandomProbablyPrime3_3 test (Section 6.2.2.2).

6.2.2.1 The Known Answer Test for B.3.3 Probably Primes

The Known Answer Test will supply the IUT with multiple iterations of public key, *e*, and random numbers for *p* and, if applicable, *q*. These values are supplied for each combination of modulus size and Miller-Rabin Table used. These values will either be successful in obtaining primes p and q or will fail. The values that are successful get to Step 5.6.2 Q is PROBABLY PRIME – Result = SUCCESS. Note to get to Step 5.6.2, the *p* value supplied is PROBABLY PRIME and has passed Step 4.5.2. The values that fail will fail at one of several locations in the B.3.3 function. Possible reasons for failure include:

Step 4.4 If $((p < (\sqrt{2})(2^{(nlen/2)-1}))$, then go to Step 4.2 and generate another random number.

Step 4.7 If (i < (5*(nlen/2)) and *p* is not prime go to Step 4.2 and generate another random number.

Step 5.5 If $((q < (\sqrt{2})(2^{(nlen/2)-1}))$, then go to Step 5.2 and generate another random number.

Step 5.8 If (i < 5*(nlen/2)) and *q* is not prime then go to Step 5.2 and generate another random number.

Note that the KAT generates random *e* values and other inputs to assure that all possible errors listed above are tested by the IUT. If an IUT supports only fixed *e* values, this IUT may not be able to process this file. Therefore this test is not required for IUTs only supporting fixed *e* value(s).

The 186-4 RSAVS:

Creates a *REQUEST* file (Filename:KeyGen3_3_KAT.req) containing:

A header consisting of:

- The CAVS version
- The algorithm being tested and the product name,
- The modulus supported,
- The Miller-Rabin information
- The date generated.
- For each modulus size/Miller-Rabin table combination tested,
- A section header indicating the value for each modulus size, and the Miller-Rabin table.
- Groups of data including
    - Public key, *e*
    - Random number for *p*
    - Random number for *q* (if applicable)

Note: The CST laboratory sends the *REQUEST* file to the IUT.

B. Creates a *FAX* file (Filename: KeyGen3_3_KAT.fax) containing:

1. The information from the *REQUEST* file and
2. Result = P or F failure code. The failure code is in the form of the instruction step number from FIPS186-4 Appendix B.3.3 (see above).

The IUT:

A. Uses the mod size, the public key exponent *e* and the random number for *p*, and, if applicable, *q* to determine if the random number is PROBABLY PRIME.

B. Creates a *RESPONSE* file (Filename: KeyGen3_3_KAT.rsp) containing:

1. The information from the *REQUEST* file and

2. Result = P or F.

Note: The IUT sends the *RESPONSE* file to the CST laboratory for processing by the 186-4 RSAVS.

The 186-4 RSAVS:

A. Compares the contents of the *RESPONSE* file with the contents of the *FAX* file.

B. Records PASS for this test if the results for all public key *e*, private prime factor *p*, private prime factor *q*, and Result match; otherwise, records FAIL.

6.2.2.2 The KeyGen_RandomProbablyPrime3_3 Test

The KeyGen_RandomProbablyPrime3_3 test for Appendix B.3.3 consists of the 186-4 RSAVS asking the IUT to supply 10 prime *p, q* pairs for each modulus size/Miller-Rabin table combination supported by the IUT. This will assure that the IUT can generate correct probable primes using Appendix B.3.3. This test is required for both IUTs supporting fixed *e* value(s) AND for IUTs supporting random *e* values.

The 186-4 RSAVS:

A. Creates a *REQUEST* file (Filename:KeyGen_RandomProbablyPrime3_3.req) containing:
   1. A header consisting of:
      i. The CAVS version
      ii. The algorithm being tested and the product name
      iii. The modulus supported
      iv. The Miller-Rabin information
      v. The date generated.
   2. For each modulus size/Miller-Rabin table combination tested,
      i. A section header indicating the value for each modulus size, and the Miller-Rabin table.
      ii. N = 10 indicating the number of entries the IUT is to send in the response file.

   Note: The CST laboratory sends the *REQUEST* file to the IUT.

B. Creates a *FAX* file (Filename: KeyGen_RandomProbablyPrime3_3.fax) containing:

1. The information from the *REQUEST* file.

The IUT:

A. Produces N sets of values per modulus size/Miller-Rabin table combination supported by the IUT.

B. Creates a *RESPONSE* file (Filename: KeyGen RandomProbablyPrime3_3.rsp) containing:

a. 10 sets of values per modulus size/Miller-Rabin table combination supported by the IUT. The values in each set includes:

i. *e*, the public key exponent,
ii. *p*, the probable prime,
iii. *q*, the probable prime,
iv. *n*, and
v. *d*, the private key exponent.

Note: The IUT sends the *RESPONSE* file to the CST laboratory for processing by the 186-4 RSAVS.

The 186-4 RSAVS:

A. Using the *e, p,* and *q,* the RSAVS runs Appendix B.3.3 using *p* in Section 4.2 and *q* in Section 5.2 to assure that each number is a probable prime.

B. Compute *n* and *d* and compare them to the values in the response file.

C. If all the values pass, the test is successful. If any of the values do not pass, the test will fail.

6.3 186-4 RSA Signature Generation Test

An implementation of FIPS 186-4 RSA may generate the digital signature. There are 3 different RSA algorithms included in FIPS 186-4. They include the RSA from X9.31, the PKCS#1 1.5, and the PKCS#1 PSS algorithms. FIPS 186-4 introduces additional constraints for each of these RSA algorithms.

The FIPS 186-4 RSA Signature Generation Test tests the ability of an IUT to produce correct signatures. To test signature generation, the 186-4RSAVS supplies ten messages to the IUT for each modulus size/SHA algorithm combination. The IUT generates the corresponding signatures and returns them to the 186-4RSAVS. The 186-4RSAVS validates the signatures by using the associated public key to verify the signature.

The 186-4RSAVS:

A. Creates a *REQUEST* file and a *FAX* file for each RSA algorithm being tested, (Request Filenames: SigGen931_186-3.req, SigGen15_186-3.req, SigGenPSS_186-3.req) (FAX Filenames: SigGen931_186-3.fax, SigGen15_186-3. fax, SigGenPSS_186-3. fax) containing:

1. A header consisting of:

   a. The CAVS version,
   b. The algorithm being tested and the product name,
   c. The modulus size/SHA size (with SALT len if applicable) combinations, and
   d. The date generated.

2. For each modulus size tested, a section header indicating the modulus length

3. Ten groups of data for each modulus size/SHA size combination supported consisting of

   a. The SHA algorithm supported,

   b. A message to be signed.

Note: The CST laboratory sends the *REQUEST* file to the IUT.

The IUT:

A. Generates the signatures for the messages supplied in the *REQUEST* file.

B. Creates a *RESPONSE* file for each RSA algorithm being tested (Request Filenames: SigGen931_186-3. rsp, SigGen15_186-3. rsp, SigGenPSS_186-3. rsp) containing:

1. The header described above,

2. For each mod size tested,

   a. a section header indicating the mod length,
   b. a modulus, *n*,
   c. a public key, *e*, corresponding to the private key, *d*, used to generate the signatures,

3. Ten groups of data for each SHA algorithm supported consisting of

   a. The SHA algorithm supported,

   b. A message, *Msg*,

   c. Signature value, *s*.

Note: The IUT sends the *RESPONSE* file to the CST laboratory for processing by the 186-4RSAVS.

The 186-4RSAVS:

A. Uses the respective public keys to verify the signatures in the *RESPONSE* file.

B. Records PASS for this test if all conditions are met; otherwise, records FAIL.

6.4 186-4 RSA Signature Verification Test

An implementation of FIPS 186-4 RSA may verify the digital signature. There are 3 different RSA algorithms included in FIPS 186-4. They include the RSA from X9.31, the PKCS#1 1.5, and the PKCS#1 PSS algorithms. FIPS 186-4 introduces additional constraints for each of these RSA algorithms.

The FIPS 186-4 RSA Signature Verification Test tests the ability of the IUT to recognize valid and invalid signatures. For each modulus size/SHA algorithm selected, the 186-4RSAVS generates a modulus and three associated key pairs, (*d*, *e*). Each private key *d* is used to sign six pseudorandom messages each of 1024 bits. Some of the public keys, *e*, messages, IR format, or signatures are altered so that signature verification should fail. The modulus, SHA algorithm, public key *e* values, messages, and signatures are forwarded to the IUT. The IUT then attempts to verify the signatures and returns the results to the 186-4RSAVS, which compares the received results with its own stored results.

The 186-4RSAVS:

A. Generates 3 groups of data for each supported modulus size. Each group consists of a modulus and 6 sets of data for each supported SHA algorithm. Each set of data contains

1. The SHA algorithm supported,

2. A public/private key pair that is consistent with the modulus,

3. A pseudorandom message and

4. A signature for the message using the private key.

B. Alters the public key *e*, the message, the IR format, or the signature for five out of six of the public key/message/signature sets such that the message verification fails.

C. Creates a *REQUEST* file (Filename: SigVer931_186-3.req, SigVer15_186-3.req, SigVerPSS_186-3.req) containing:

1. A header consisting of:

a. The CAVS version,

b. The algorithm being tested and the product name,

c. The modulus size/SHA size (with SALT len if applicable) combinations, and

d. The date generated.

2. For each modulus size supported:

a. A section header indicating the modulus length

b. The modulus *n* for the supported modulus size,

c. 3 groups of data consisting of 6 sets of data for each supported SHA algorithm. Each set of data contains:

i. The SHA algorithm supported,

ii. A public key *e* corresponding to the private key used to sign the messages,

iii. The pseudorandom message and

iv. The signature.

Note: The CST laboratory sends the *REQUEST* file to the IUT.

D. Creates a *FAX* file (Filename: SigVer931_186-3.fax, SigVer15_186-3. fax, SigVerPSS_186-3.fax) containing:

1. The information from the *REQUEST* file,
2. The private key, *d*, associated with the public key, *e*, and
3. The result indicating whether the signature verification process should pass or fail, for each public key/message/signature set, and, if it should fail, why.

The IUT:

A. Attempts to verify the signatures for the messages supplied in the *REQUEST* file using the corresponding modulus *n*, SHA algorithm and the public key *e*.

B. Creates a *RESPONSE* file (Filename: SigVer931_186-3.rsp, SigVer15_186-3.rsp, SigVerPSS_186-3.rsp) containing:

1. The information from the *REQUEST* file and

2. An indication of whether the signature verification passed or failed for each public key/message/signature set.

Note: The IUT sends the *RESPONSE* file to the CST laboratory for processing by the RSAVS.

The 186-4RSAVS:

A. Compares the contents of the *RESPONSE* file with the contents of the *FAX* file.

B. Records PASS for this test if the results for all public key/message/signature sets match; otherwise, records FAIL.

6.5 186-2 Legacy RSA Signature Verification Test

This option tests the ability of the IUT to recognize valid and invalid signatures. For each modulus size/SHA algorithm selected, the RSAVS generates a modulus and three associated key pairs, (d, e). Each private key d is used to sign four pseudorandom messages each of 1024 bits. Some of the public keys, e, messages or signatures are altered so that signature verification should fail. The modulus, SHA algorithm, public key *e* values, messages, and signatures are forwarded to the IUT. The IUT then attempts to verify the signatures and returns the results to the RSAVS, which compares the received results with its own stored results.

The RSAVS:

A. Generates 3 groups of data for each supported modulus size. Each group consists of a modulus and 4 sets of data for each supported SHA algorithm. Each set of data contains

1. The SHA algorithm supported,
2. A public/private key pair that is consistent with the modulus,
3. A pseudorandom message and
4. A signature for the message using the private key.

For the efficiency of the tool, a modulus will sometimes be used for more than one group of data when it is consistent with more than one public key. Therefore when loading the modulus for each group of data, the modulus specified for this group may be the same as that specified for the previous group.

B. Alters the public key e, the message or the signature for three fourths of the public key/message/signature sets such that the message verification fails.

C. Creates a *REQUEST* file (Filename: SigVer931_186-2.req, SigVer15_186-2.req, SigVerPSS_186-2.req) containing:

1. The Product Name;
2. For each modulus size supported:
   a. The modulus *n* for the supported modulus size,
   b. 3 groups of data consisting of 4 sets of data for each supported SHA algorithm. Each set of data contains:
      i. The SHA algorithm supported,
      ii. The information from step B, including:
         1. A public key *e* corresponding to the private key used to sign the messages,
         2. The pseudorandom message and

3. The signature.

Note: The CMT laboratory sends the *REQUEST* file to the IUT.

D. Creates a FAX file (Filename: SigVer931_186-2.fax, SigVer15_186-2. fax, SigVerPSS_186-2. fax) containing:

1. The information from the *REQUEST* file and
2. An indication of whether the signature verification process should pass or fail, for each public key/message/signature set.

The IUT:

A. Attempts to verify the signatures for the messages supplied in the *REQUEST* file using the corresponding modulus *n*, SHA algorithm and the public key *e*.

B. Creates a *RESPONSE* file (Filename: SigVer931_186-2.rsp, SigVer15_186-2. rsp, SigVerPSS_186-2. rsp) containing:

1. The information from the *REQUEST* file and
2. An indication of whether the signature verification passed or failed for each public key/message/signature set.

Note: The IUT sends the *RESPONSE* file to the CMT laboratory for processing by the RSAVS.

The RSAVS:

A. Compares the contents of the *RESPONSE* file with the contents of the FAX file.

B. Records PASS for this test if the results for all public key/message/signature sets match; otherwise, records FAIL.

Appendix A. References

[1] FIPS186-4, Digital Signature Standard (DSS), FIPS Publication 186-4, National Institute of Standards and Technology, July, 2013.

[2] American National Standard (ANS) X9.31 *Digital Signatures Using Reversible Public Key Cryptography for the Financial Services Industry (rDSA)*, September, 1998.

[3] PKCS#1 v2.1: RSA Cryptography Standard, RSA Laboratories, June 14, 2002.

[4] *Security Requirements for Cryptographic Modules*, FIPS Publication 140-2, National Institute of Standards and Technology, May 2001.