PROTECTION PROFILES FOR CERTIFICATE ISSUING & MANAGEMENT SYSTEMS

A. Arsenault, US DoD
Phone: 410-854-7018
FAX: 410-854-7463
E-mail: awarsen@missi.ncsc.mil

R. Housley, Spyrus
Phone: 703-707-0696
FAX: 703-707-8603
E-mail: housley@spyrus.com

ABSTRACT

At the heart of many recent efforts to improve Internet security are a group of security protocols such as S/MIME, TLS, and IPSec. All of these protocols rely on public-key cryptography to help provide services such as confidentiality, data integrity, data origin authentication, and non-repudiation. Support for this public-key cryptography is provided by a Public Key Infrastructure, or PKI. The PKI is responsible for binding public keys into certificates and managing those certificates throughout their life-cycle. The part of the PKI directly responsible for generation, issuance, and revocation of certificates is referred to as the Certificate Issuing and Management System, or CIMS.

It is important to many potential CIMS customers to understand the level of security provided by a specific product or service. Furthermore, in order to accurately compare products and services from many different sources, built using many different architectures, there should be one set of requirements that can be used to evaluate CIMS. This set of requirements should be written in internationally accepted terms, such as the Common Criteria. Furthermore, it should be generic enough so that it can be used for a wide variety of architectures, but sound enough so that it can be used to provide a meaningful evaluation. This paper describes the development of a set of Common Criteria Protection Profiles that can be used to evaluate CIMS products and services.

PKIs and CIMS

At the heart of recent efforts to improve Internet security are a group of security protocols such as S/MIME [SMIME}, TLS [TLS], and IPSec {IPSEC]. All of these protocols rely on public-key cryptography to help provide services such as confidentiality, data integrity, data origin authentication, and non-repudiation. The public-key cryptography is implemented using a Public Key Infrastructure, or PKI. [RDMP]

A PKI is defined as: [RDMP]

> The set of hardware, software, people, policies and procedures needed to create, manage, store, distribute, and revoke certificates based on public-key cryptography.

The purpose of a PKI is to provide trusted and efficient key and certificate management, thus enabling the use of authentication, non-repudiation, and confidentiality.

Traditionally, a PKI has consisted five types of components [MISPC]:

- *Certification Authorities (CAs) that issue and revoke certificates;*
- *Registration Authorities (RAs) that vouch for the binding between public keys and certificate holder identities and other attributes;*
- *Certificate holders that are issued certificates and can sign digital documents and/or encrypt & decrypt information;*

- *Clients that validate digital signatures or encrypted messages and their certification paths from a known public key of a trusted CA;*

- *Repositories that store and make available certificates and Certificate Revocation Lists (CRLs).*

In addition to those, two new components need to be considered to meet other requirements sometimes imposed on PKIs:

- Archive systems *that are used to store information for long-term retrieval (e.g., retrieval for proof in a court case of the correctness of an action taken 10 years earlier)*

- Key recovery systems *to support the recovery of encryption keys and/or data under some particular set of circumstances.*

In many ways, the heart of any PKI is the set of CA's and RA's that combine to issue certificates and revocation notices. Together, the CA's and RA's in a PKI are responsible for: gathering the information necessary to create a certificate that binds together an identity, a set of authorizations, and a public key; for vouching for the correctness of that information; for creating and issuing that certificate; and for managing that certificate during its lifetime – including revoking it, if necessary. Collectively, we refer to the set of CA's and RA's in a PKI as the Ceritificate Issuing and Management System, or CIMS, of that PKI.

The Need for CIMS Evaluation Requirements

Determining the Security Characteristics of a CIMS

There are now a number of different CIMS products and services available in the commercial marketplace. There are CIMS products sold by such companies as Spyrus, Microsoft, Netscape, Entrust, CertCo, and others. There are CIMS services provided by such companies as VeriSign, GTE-Cybertrust, Thawte, and others. A prospective purchaser of CIMS products or services is presented with a daunting array of products with different features, providing different levels of security, quality and assurance, covering a dramatic range of costs in both equipment and people.

How does a prospective customer then decide which product or service to choose? There are a number of studies available which purport to compare costs (see for example [ABER] and [GIGA]), so some information can be gained about that issue. There are ways to directly compare the lists of features (such as protocols) provided by different vendors. However, at this time, there is no way to directly compare the levels of security provided by different vendors. Customers are left to compare long and legalistic Certification Practice Statements (CPS), along with marketing brochures (and perhaps direct discussions with vendors), and these customers must then try to use their own expertise to make comparisons. A CPS may or may not go into the level of hardware/software security provided by the product, and the other information may well make at most general, unverified claims. Customers for whom a very high level of security is needed have no easy way to tell whether a particular vendor's offering is acceptable for them, or is more appropriate for customers who need only a moderate level of security.

We assume that customers know their environments and requirements. Customers will have in mind a Certificate Policy and Certification Practices Statement, which describe what level of security the customer will achieve with her system, and how that will be achieved. The customer also has an idea of what threats she wants to counter. Now, what remains is for the customer to be able to gauge how well a particular product meets those requirements.

Some of the product/service security characteristics about which a customer may care include:

- *Cryptographic algorithms:* Which algorithms does the product support? Is it only "export strength" 40 to 56 bit algorithms, or is it strong cryptography such as Triple DES or one of the algorithms with 128-bit keys?

- *Random number source*: From where does the cryptographic algorithm get the random numbers necessary to generate keys? How random – or predictable – is that source?

- *Cryptography implementation:* How well is the cryptography implemented? There are any number of examples of good cryptographic algorithms being implemented badly, with the result being a weak, insecure system. Has the cryptographic implementation been evaluated against an established standard, such as FIPS 140-1 [FIPS140]? If so, at what level was it approved?

- *Hardware/software platform:* On what hardware platform/operating system combination(s) does the CIMS run? A platform such as WindowsNT or Solaris generally provides more protection for a CIMS than does a platform such as Windows98.

- *CIMS implementation*: How good is the actual implementation of the CIMS? For example, what design processes were used, what software development methods were used, how well were they carried out? How accurate is the system documentation? Why should a customer believe that a CIMS will actually do what is claimed?

- *Personnel security & training:* What are the requirements that the product imposes for personnel training? What background investigation/bonding requirements are suggested for the product to achieve various levels of security? (This is related to how easy the product is to use securely, and how well it defends itself against malicious users.)

- *Physical security*: What level of physical security needs to be provided to adequately counter the threats that the customer wants countered? How well does this compensate for any lack of technical countermeasures?

- *Procedures followed:* What procedures need to be followed by the CIMS staff in order to operate the CIMS securely?

Each of these areas is important to the overall security of the CIMS. Some of these are already addressed by other guidance or standards. For example, the choice and effectiveness of cryptographic algorithms is generally covered by various national or industry standards. Similarly, the evaluation of the implementation of cryptography is generally addressed by a national or industry standard (for example, FIPS 140-1 is a Government standard in the United States, and it is often used by others throughout North America). Various organizations will have requirements and standards for personnel training and physical security. However, there is to date minimal to no guidance on the actual implementation of the CIMS hardware/software, and only minimal guidance on the implications of the hardware/software platform on which the CIMS is running. Thus, Spyrus believed that it was necessary to develop as guidance a set of requirements against which those things could be evaluated.

Other Efforts

Before we began writing security requirements on our own, we tracked a number of other related efforts. One such effort is the development by NIST of security requirements for CIMS. While much of NIST's work was promising, we were concerned about their initial schedules. We believed that security requirements would need to be established in 6-12 months to be the most useful.

Another effort we tracked is the development by the UK of a set of CIMS-related Protection Profiles for the Cloud Cover project, which is developing a PKI for Her Majesty's Government. That effort was

technically interesting, but it was restricted to a specific architecture, going so far as to require the use of specific cryptographic algorithms. Thus, it was not general enough for our purposes.

A third effort is the OSCAR (Open digital Signature Certification ARchitecture) program. OSCAR is a collaborative project supported by Notary organizations from Belgium, France, Spain and the United Kingdom to define and demonstrate a certification architecture for electronic digital signatures across Europe. OSCAR is part of the ETS (European Trusted Services) program to put in place an organizational, legal and technical infrastructure to support secure information exchange. While this effort is also technically interesting, it is restricted to a specific architecture, and thus not general enough for our purposes.

Spyrus Decisions

Based on this information, Spyrus made the decision to go ahead and try to lead the development of security requirements for CIMS. The development would be coordinated with other major CIMS providers through the NIST Cooperative Research and Development (CRADA) effort, and with major CIMS customers through the ANSI X9F5 working group. The major goal of the effort would be:

> *develop a set of security requirements that could be used to evaluate the hardware/software implementations of CIMS, so that competing products can be understood and fairly compared; and customers can understand the properties of prospective solutions.*

Evaluation of CIMS Implementations

Once it was agreed that a set of requirements needed to be developed for the evaluation of CIMS implementations, a set of goals for the requirements was adopted. Significant goals included:

(1) All product developers and service providers must be able to compete fairly, and evaluated under a common set of rules. Furthermore, there must be a way for potential customers to fairly compare the variety of products and services on the market against their own requirements, and against each other. This implies that there must be a single set of requirements, or related families of requirements, that can cover all CIMS products, if possible.

(2) The set of requirements must be usable to evaluate all or a majority of the commercial products now on the market or coming to the market, AND it must also be usable to evaluate the implementations of service providers (i.e., service providers must be able to use the CIMS requirements to evaluate their own systems, and tell their customers “we meet Level x”).

(3) Developers must be able to perform a self-evaluation of their products/services against the requirements, so that they know roughly where they stand. The requirements must not be such that only a few “wizards” understand them, or understand how a system will fare in an evaluation.

(4) The requirements must also support evaluation by independent (3rd party) laboratories, as addressed under the FIPS 140-1 or Common Criteria [CC] programs.

What Form to Use

After setting these goals, the last remaining question to be answered was then: what form should the requirements document take? There were essentially three choices considered:

(1) Starting with a "blank sheet of paper", and capturing the requirements. Then, when the requirements were set, trying to get them accepted as, for example, a US Federal Information Processing Standard (FIPS).

(2) Writing the requirements as an interpretation of the US Trusted Computer System Evaluation Criteria [TCSEC] or the European Information Technology Security Evaluation Criteria [ITSEC].

(3) Writing the requirements as one or a series of Common Criteria Protection Profiles.

Each option was carefully considered. Choice (2) was quickly eliminated, though, because support for the TCSEC and ITSEC is quickly dwindling with the adoption of Version 2 of the Common Criteria. Various nations have announced that they will be ending their TCSEC/ITSEC evaluation support programs as they transition to the Common Criteria. Given this, there seemed to be no benefit to taking this approach, and the option was eliminated.

Option (1) was appealing for a variety of reasons. First, starting with a blank sheet of paper allowed us to write the requirements as we saw fit, without having to worry about forcing them into any existing structure. Second, it freed us from an implicit reliance on any existing or planned evaluation methodology which may or may not fit the PKI world as it develops. Finally, it allowed us to eliminate the learning curve necessary to use the Common Criteria.

On the other hand, Option (1) had a major drawback: we would be developing a new set of requirements, which would then require an evaluation program, without relying on any existing support structures. Setting up our own evaluation scheme would undoubtedly cost substantial amounts of money and time, and may not even be possible given the political realities.

Thus, we chose Option (3), writing a set of Common Criteria Protection Profiles. The biggest drawbacks of this approach have to do with the newness and complexity of the Common Criteria. Although the theory of the Common Criteria has been around for a number of years, it is still relatively unproven – there aren't large numbers of products of any type on the market that have been evaluated against Common Criteria Protection Profiles or Security Targets. Additionally, because the Common Criteria was deliberately written to be tailorable to cover almost any type of information technology device, developing a Protection Profile of it involves considerable understanding and effort.

To make matters worse, at the time this effort was started, there were essentially no tools in existence to help a prospective Protection Profile author. (Tools have since been made available, for example from Sparta Corp., but they are still relatively immature – they are in essentially beta releases as this is written.) On top of all of that, the Common Criteria uses language that is somewhat arcane – a potential Protection Profile writer has to spend a considerable time learning the Common Criteria meanings of terms like "selection", "assignment", and "refinement", plus a seemingly endless supply of acronyms and bizarre requirement designators. To the uninitiated, this alone can be daunting.

Despite this, we decided that Protection Profiles were the best approach to take, and started on our development effort.

The Protection Profiles

Levels

After discussions with other experts in this area (largely, NIST), we decided that there were four different levels of CIMS security that could be of interest to customers. These levels were distinguished by the types and levels of threats that they countered. We named these levels 1 through 4, with level 1 countering

minimal threats (and thus providing the lowest level of security), and level 4 countering the highest level of threat.

The levels were defined as:

> *Level 1: Designed to not be resistant to any significant threats. CIMS at this level rely entirely on the operating system and environment for protection.*
>
> *Level 2: Designed to resist some threats from the network to which the computer is connected, but not resist any threats from the local environment. More sophisticated network-based threats and all local threats must be countered with operating system and environmental controls.*
>
> *Level 3: Designed to resist most threats from the network to which the computer is connected, and some software-based threats from local users and others who can get physical access to the system. Does not resist hardware-based or sophisticated software-based attacks from those with physical access.*
>
> *Level 4: Designed to resist most network-threats, and most software-based threats from those with physical access to the computer. Relies minimally on the operating system for security protection. The only successful threats should be hardware-modification-based threats from those with local access to the computer.*

CIMS vs. RA & CA requirements

There are a number of different ways in which CA's and RA's can be designed to create a system that is responsible for the issuance and management of certificates. For example, Spyrus has designed a system, called the S2CA, in which the RA is responsible for generating the cryptographic keys, collecting all of the other user information, formatting a proposed certificate and sending it to the CA. The CA, which is an unattended device, receives the certificate, stores important information in its database, signs or refuses to sign the certificate, and sends the certificate or an error message back to the RA. The RA will then load the certificate on to a hardware token, or create a software token containing it, and provide it to the user. By contrast, with VeriSign's OnSite products, keys are generated by the user and provided to the RA. The RA will then send the request to the unattended CA for signature. The certificate can be returned directly to the user, or to the RA**.** Entrust, by contrast, has signature keys generated by the user, while key management keys are generated by the CA in order to facilitate key recovery if desired by the user

Because of the differences in architectures, and the fact that many of these different architectures will ultimately lead to the same user requirements being met, it is difficult to specify a set of security requirements. For the purposes of developing security requirements, we decided to ignore CA and RA requirements, and instead simply write CIMS requirements. Thus, any developer can choose how to allocate functionality among the CA, RA, and client system to meet the overall requirements. As long as the system functions can be mapped in a Common Criteria Security Target (ST) or Target of Evaluation (TOE) into the requirements of a particular-level Protection Profile, the system can be evaluated.

What Constitutes a TOE

According to the Common Criteria, a product or system which is ultimately evaluated is a "Target of Evaluation", or TOE. A Protection Profile must make assumptions and/or define what types of TOEs can be evaluated. In our case, there were two important questions to be answered: Should the platform (i.e., hardware suite, operating system, and any "middleware" software) on which the CIMS executes be part of the TOE? Should the cryptographic module (whether hardware or software) used by the CIMS be part of the TOE?

For each of these two, there are arguments that can be made either way. The strongest argument in favor of including the platform and cryptographic module in the TOE is this: In order to get a good overall understanding of how the system – CIMS, platform, cryptographic module – works as a whole, it should be evaluated together. If you evaluate each part separately, you can't really be sure how much security you have when everything is assembled, because our knowledge of the effects of system composition upon module security is somewhat limited.

On the other hand, adding each of these things to the TOE increases the cost, schedule, and risk associated with a product evaluation. First, we'll look at the cryptographic module. The Common Criteria is very weak in the area of cryptography in general. The sections that exist today (FCS_… and FCO_…) are very generic. There is admittedly additional work going on to strengthen the cryptographic sections of the Common Criteria, but it is not yet mature. Thus, there is some level of risk in trying to include cryptography in a Protection Profile, and in a TOE. Additionally, there are already other programs in existence for the evaluation of cryptographic modules. For example, in North America, there is the FIPS 140-1 program, which is widely regarded as successful. Trying to put a similar program into a CIMS Protection Profile can be rightly deemed "reinventing the wheel", and thus we explicitly excluded the cryptographic module from the Protection Profile and from TOEs that will be evaluated against it.

The platform – largely, the operating system – presents a different challenge. It seems natural to include the operating system in the evaluation of a CIMS – after all, in many cases, it is the operating system that provides such fundamental security mechanisms as user identification and authentication; control of access to CIMS resources; and audit. However, including the operating system in the TOE does cause two significant problems.

The first problem is that it dramatically increases the size of the product being evaluated, and thus necessarily the cost and time taken by the evaluation. Typical operating systems consist of millions of lines of code, in comparison to a few hundred thousand or so for a typical CIMS.

The second problem is that it may not be possible for a CIMS developer to successfully get an operating system evaluated. Evaluation, particularly at the higher levels of assurance, requires the submission and explanation of detailed documentation on the system internals. This information is often unavailable to CIMS vendors (and most other third-party software developers!), and it may not even exist, if the operating system developer is not interested in an evaluation of her own. In practice, if the operating system had to be included in the TOE, it might only be possible for those CIMS vendors who are also operating systems developers to successfully finish evaluation. All other CIMS vendors – which means most of the current marketplace – would be barred from evaluation.

Since we did not consider this desirable, or useful from the customer's viewpoint, we adopted another approach. As part of the system assumptions, the CIMS developer must specify those features to be provided by the operating system – e.g., process isolation, user authentication. As part of the evaluation, the CIMS developer must specify at least one operating system that provides those features. Then, the evaluation facility will ensure that the operating system does what is expected. The evaluation facility may simply refer to an earlier evaluation of that operating system that included the relevant features, and check that the CIMS does not nullify them. (For example, evaluators should ensure that installation of the CIMS does not require configuring the operating system in such a way that the necessary security features are turned off.) If the operating system specified by the CIMS vendor has not been previously evaluated, then the evaluators will have to examine it to the extent that they are satisfied that the necessary features are present.

If there is no operating system that provides the features identified by the CIMS developer when configured as specified, then the evaluation of the CIMS should not proceed.

The Four Protection Profiles

Level 1

Functional Requirements

The functional requirements chosen for this Protection Profile are:

FAU_GEN.1	Audit data generation
FAU_GEN.2	User identity association
FAU_SAR.1	Audit review
FAU_STG.1	Protected audit trail storage
FCO_NRO.1	Selective proof of origin
FIA_AFL.1	Authentication Failure Handling
FIA_UAU.2	User authentication before any action
FIA_UID.2	User identification before any action
FMT_SMR.1	Security Roles
FPT_STM.1	Reliable time stamps

Assurance Requirements

The assurance requirements chosen for this Protection Profile are those from EAL-1, plus ATE_FUN.1, "Functional Testing". EAL-1 does not require that the developer actually test her own product; we added this requirement to ensure that a CIMS developer has performed and documented at least some level of testing prior to evaluation.

Level 2

Functional Requirements

The functional requirements chosen for this Protection Profile are:

FAU_GEN.1	Audit data generation
FAU_GEN.2	User identity association
FAU_SAR.1	Audit review
FAU_SEL.1	Selective Audit
FAU_STG.1	Protected audit trail storage
FCO_NRO.1	Selective proof of origin
FDP_ACC.1	Subset access control
FDP_ACF.1	Security attribute based access control
FDP_IFC.1	Subset information flow control
FDP_IFF.1	Simple security attributes
FDP_ITT.1	Basic internal transfer protection
FDP_ITT.3	Integrity monitoring
FIA_AFL.1	Authentication Failure Handling
FIA_UAU.2	User authentication before any action
FIA_UID.2	User identification before any action
FMT_MSA.1	Management of security attributes
FMT_MTD.1	Management of TSF data
FMT_SMR.1	Security Roles
FPT_AMT.1	Abstract machine testing
FPT_STM.1	Reliable time stamps

Assurance Requirements

The assurance requirements chosen for this Protection Profile are those from EAL-3, except for ALC_DVS.1, “Identification of Security Measures”. That requirement largely deals with the physical and personnel security measures applied to the development and maintenance of the CIMS. However, at this level, it is expected that most CIMS products will be developed under commercial practices, and there is generally no extra security and no documentation covering that. While it is true that commercial developers probably should document their practices in this area (and some of them should probably develop some practices in this area), we believe that imposing this requirement would cause more harm than good in the short term, as some companies would object to having to do something like this just for evaluation, and not change their practices.

Level 3

Functional Requirements

The functional requirements chosen for this Protection Profile are:

FAU_GEN.1	Audit data generation
FAU_GEN.2	User identity association
FAU_SAR.1	Audit review
FAU_SEL.1	Selective Audit
FAU_STG.1	Protected audit trail storage
FCO_NRO.1	Selective proof of origin
FDP_ACC.1	Subset access control
FDP_ACF.1	Security attribute based access control
FDP_IFC.1	Subset information flow control
FDP_IFF.1	Simple security attributes
FDP_ITT.1	Basic internal transfer protection
FDP_ITT.3	Integrity monitoring
FDP_RIP.1	Subset Residual information protection
FDP_SDI.1	Stored data integrity monitoring
FDP_DAU.1	Basic data authentication
FDP_ETC.1	Export of user data without security attributes
FDP_ITC.1	Import of user data without security attributes
FIA_AFL.1	Authentication Failure Handling
FIA_UAU.2	User authentication before any action
FIA_UID.2	User identification before any action
FMT_MSA.1	Management of security attributes
FMT_MTD.1	Management of TSF data
FMT_SMR.1	Security Roles
FPT_AMT.1	Abstract machine testing
FPT_STM.1	Reliable time stamps
FTP_TRP.1	Trusted path

Assurance Requirements

The assurance requirements chosen for this Protection Profile are those from EAL-4, with the addition of ADV_INT.1, “Modularity”. We believe that a CIMS being developed to resist this type of threats needs to be well designed, with a level of modularity acceptable under good software engineering practices.

Level 4

Functional Requirements

The functional requirements chosen for this Protection Profile are:

FAU_GEN.1	Audit data generation
FAU_GEN.2	User identity association
FAU_SAR.1	Audit review
FAU_SEL.1	Selective Audit
FAU_STG.1	Protected audit trail storage
FCO_NRO.1	Selective proof of origin
FDP_ACC.2	Complete access control
FDP_ACF.1	Security attribute based access control
FDP_IFC.2	Complete information flow control
FDP_IFF.1	Simple security attributes
FDP_ITT.1	Basic internal transfer protection
FDP_ITT.3	Integrity monitoring
FDP_RIP.2	Full residual information protection
FDP_SDI.2	Stored data integrity monitoring and action
FDP_DAU.1	Basic data authentication
FDP_ETC.1	Export of user data without security attributes
FDP_ITC.1	Import of user data without security attributes
FIA_AFL.1	Authentication Failure Handling
FIA_UAU.2	User authentication before any action
FIA_UID.2	User identification before any action
FMT_MSA.1	Management of security attributes
FMT_MSA.2	Secure Security Attributes
FMT_MSA.3	Static Attribute Initialisation
FMT_MTD.1	Management of TSF data
FMT_SMR.1	Security Roles
FPT_AMT.1	Abstract machine testing
FPT_STM.1	Reliable time stamps
FTP_TRP.1	Trusted path

Assurance Requirements

The assurance requirements chosen for this Protection Profile are those from EAL-6, except that AVA_CCA.2, "Systematic Covert Channel Analysis," has been eliminated. Although Covert Channels can indeed be a significant security concern in a high-assurance system, Covert Channel Analysis itself can prove to be a significant consumer of people, money, and time. Thus, we have deleted this requirement from the EAL-6 level.

Status of the Effort & Lessons Learned

At the time of this writing, we have developed a set of draft Protection Profiles and circulated them for review to a selected group of people. By the time of the NISS Conference in October, we expect revised drafts to be the subject of a work item in the ANSI X9F5 committee, and expect the work to by synchronized with NIST.

References:

[ABER] Aberdeen Group, "Evaluating the Cost of Ownership for Digital Certificate Projects", Executive White Paper, July 1998.

[CC] Common Criteria Editorial Board, "Common Criteria for Information Security Technology, Version 2.0", Parts 1 (CCIB-98-026), 2 (CCIB-98-027), and 3 (CCIB-98-028), May 1998. (Also known as: International Standards Organization, Draft International Standard 15408-1, 15408-2, and 15408-3, respectively.)

[FIPS140] Federal Information Processing Standard 140-1, “Security Requirements for Cryptographic Modules”, 11 January 1994.

[GIGA] Machefsky, Ira, “A Total Economic Impact Analysis of Two PKI Vendors: Entrust and VeriSign,“ Giga Information Group, September 1998.

[IPSEC] Thayer, R., N. Doraswamy, and R. Glenn, “IP Security Document Roadmap”, RFC 2411, November 1998.

[ITSEC} “Information Technology Security Evaluation Criteria”,

[MISPC] Burr, William, Donna Dodson, Noel Nazario, and W. Timothy Polk, “Minimum Interoperability Specification for PKI Components, Version 1”, NIST Special Publication 800-15, 3 September, 1997.

[RDMP] Arsenault, A., and S. Turner, “PKIX Roadmap,” Internet Draft, <draft-ietf-pkix-roadmap-00.txt>, 8 September 1998.

[SMIME] Ramsdell, B., “S/MIME Version 3 Certificate Handling”, Internet Draft, 14 December 1998.

[TCSEC] “Department of Defense Trusted Computer System Evaluation Criteria”, DoD Standard 5200.28-STD, 26 December 1985.

[TLS] Dierks, T., and C. Allen , “The TLS Protocol Version 1.0”, RFC 2246, January 1999.

ACKNOWLEDGEMENTS

The authors would like to thank the following people, who have provided significant input into this project over its lifetime: Donna Dodson, David Cooper, Kathy Lyons-Burke, Bill Burr, and Tim Polk of NIST; Dave Fillingham and Chris Yazbeck of NSA; Charlie Moore of Spyrus; Pat Cain of GTE; and Pierre Boucher of Entrust.