Department of Defense (DOD)
Class 3 Public Key Infrastructure (PKI)
Public Key-Enabled Application Requirements

13 July 2000

Version 1.0

Table of Contents

List of Figures

List of Tables

1.0 INTRODUCTION

This document describes requirements for applications enabled to use *public key* (PK) *technology* and interact with the Department of Defense (DOD) Public Key Infrastructure (PKI). PK technology has promise as an enabling technology to provide security and to provide truly paperless, digital environments. PK techniques have the greatest potential in applications that involve communications or movement of information over communications or computer networks. PK techniques along with the DOD PKI allow secure communication between parties without prior agreement or arrangement.

1.1 Purpose

The purpose of this document is to provide minimum requirements for PK enabled applications to interoperate with the DOD Class 3 PKI and to ensure effective protection for cryptographic functions and objects (e.g., encryption keys) that support the application. Interoperability has multiple meanings. The interoperability requirements herein focus primarily on ensuring interoperation of applications throughout the DOD with the DOD PKI. These interoperability requirements facilitate but do not necessarily ensure other forms of PK-related interoperability. Examples of other forms of interoperability include interoperability of distributed components of an application, of applications or application components from different vendors, and with communities having members who are outside the DOD and not served by the DOD PKI. The requirements herein should be augmented to satisfy other interoperability concerns as necessary on an application specific basis.

DOD organizations should either use the requirements herein as criteria to select commercial products that are PK-enabled or include the requirements in the overall requirements for development and acquisition efforts involving PK-enabled applications.

Government organizations including the Joint Interoperability Test Command (JITC) will use the requirements contained herein as a basis for testing an application's ability to interoperate with the DOD PKI.

1.2 Intended Readers

Intended readers include application developers, providers, and vendors. The target applications include those developed under DOD sponsorship to satisfy specific requirements of a DOD program as well as general applications developed without DOD sponsorship and intended for a broader audience that includes segments of the DOD.

The document assumes readers are already familiar with public key and PKI fundamentals.

1.3 Use of *Shall*, *Must*, *Will*, *Should*, and *May*

This document frequently uses the words *shall*, *must*, *will*, *should*, and *may*. *Shall* specifies that a requirement is binding or mandatory. Applications must satisfy requirements specified with *shall*. *Must* and *shall* are synonyms when specifying requirements. *Should* and *may* express non-mandatory provisions. *Should* or the adjective *recommended*, mean that there may exist valid reasons in particular circumstances to ignore a particular item, but the full implications must be understood and carefully weighed before choosing a different course. An underlined should (*should*) indicates that the requirement may become mandatory in a future version of this document. *May* or the adjective *optional* mean that an item is truly optional. The developer or vendor has the discretion to choose. One developer or vendor may choose to include the item because a particular marketplace requires it or because the vendor feels that it enhances the product while another developer or vendor may omit the same item. An implementation which does not include a particular option MUST be prepared to interoperate with another implementation which does include the option, though perhaps with reduced functionality. In the same vein an implementation which does include a particular option MUST be prepared to interoperate with another implementation which does not include the option (except, of course, for the feature the option provides). *Will* may be used to express a declaration of purpose by the organizations responsible for managing and operating the DOD PKI or for evaluating applications' compliance with requirements. This document does not specify requirements for the DOD PKI. *Will* is used in this document to describe relevant aspects of the DOD PKI. *Will* may also be used in cases where the simple future tense is required.

1.4 Overview

This document begins with preparatory background and contextual information prior to describing the requirements. Sections 2 and 3 contain this background and contextual information. Section 2 provides background information. The purpose of the background section is to establish terms used in the remainder of the document. Section 3 establishes the scope of this document. Section 3 lists the documents that mold and supplement the requirements found in this document. Section 3 also identifies the types of applications to which the requirements apply. Section 4 contains the actual technical requirements that applications must satisfy. Section 5 provides qualification requirements. These requirements identify the methods used to determine that the application satisfies its technical requirements. The qualification requirements include interoperability and assurance requirements. Several appendices supplement the information contained in the body of the document. All except Appendix D supplement information in the document but do not contain technical requirements. Appendix D

supplements Section 4 and provides additional detailed requirements. The final sections contain references and a list of acronyms.

2.0 BACKGROUND

This section provides background. The purpose is to establish terminology for use in describing application requirements. Key terms used in the remainder of the document are highlighted in ***bold Italics***. The section has no requirements.

Public key cryptography uses ***asymmetric*** keys rather than traditional ***symmetric*** keys. With symmetric cryptography communicating parties use the same key for encryption and decryption. The parties have to share the key. The secrecy of their communications depends on:

- A reliable and secure method to distribute the key to the parties and
- The parties keeping the key secret and not making it known to additional parties without the original parties' knowledge and approval.

Asymmetric cryptography uses different keys for encryption and decryption.[1] There are algorithms to generate the two keys, the ***key pair***, for their owner. The two keys are uniquely paired. Operations with the two keys are commutative; that is, data encrypted with one of the keys and then decrypted with the other key results in the original data. Public key cryptography is asymmetric cryptography where the ***owner*** of the key pair designates one key as the private key and the other as the public key. The owner retains the private key and is careful to not reveal it to anyone else. The owner freely distributes the public key and may openly post or publish it where others may easily retrieve it.

Public keys can be used to either sign or encrypt information. To ***sign*** data the owner encrypts the data with his or her private key. The receiver decrypts the information with the owner's public key. The receiver knows that the data was encrypted with the private key, which is known only to the pair's owner. Therefore, the owner signed the ***message***.[2] To ***encrypt*** information the sender encrypts the information using the receiver's public key. Only the owner who has the private key can decrypt the encrypted information and view the original data.

---

[1] When discussing asymmetric cryptography, encryption refers to the operation performed on clear or unencrypted data to produce encrypted data while decryption refers to the operation to transform data from an encrypted form to its original clear form.

[2] This document uses the terms message, send, and receive in a general rather than specific sense. Message refers to a logical collection of data rather than a collection specifically intended for use in communication between two parties (such as an e-mail message). The entire collection of data, the message, is usually the target of a cryptographic operation. Send and receive refer to encryption and decryption related activities respectively. Not all uses of PK methods involve data transmission. Use of message model terms seems appropriate since the usual situation involves transfer of information with different parties performing the encryption and decryption operations.

PK operations are computationally expensive. In practice hash and symmetric algorithms are used to reduce the PK computational overhead for signatures and encryption respectively. ***Hash algorithms*** are cryptographic functions that map a variable length message into a fixed length message ***digest*** or ***hash***. The originator signs a message by computing a hash or digest for the message and then encrypting the digest with his or her private key. The encrypted digest is the ***signature***. The originator sends the signature along with the message. The receiver verifies the signature by:

- Independently computing the digest for the received message.
- Decrypting the signature included with the message using the public key belonging to the apparent sender.
- Comparing the values resulting from the previous steps. If the values are equal, the receiver knows that the message was sent using the private key associated with the public key and that the message had not been altered. If the values are not equal, the message was altered, or the public key used was not the peer of the private key used to sign the message.

To reduce the computation associated with encryption, the sender of a message:

- Generates a random symmetric key, the ***session key***.
- Encrypts the message using the session key.
- Encrypts the session key using the receiver's public key.
- Transmits the encrypted message together with the encrypted session key.

The combination of the encrypted content and an encrypted session key for a recipient is a ***digital envelope*** for that recipient. The message can be sent to multiple recipients by encrypting the session key using each intended recipient's public key and including the encrypted session keys in the envelope.

The previous sections have described the primary functions of PK cryptography: encryption and digital signature. These two basic functions support four distinct security services. These services are:

Authentication. Authentication is the process of verifying and ensuring an individual's (or other ***entity***'s) identity.[3]

Confidentiality. Data confidentiality is the protection of information from unauthorized disclosure.

[3] Identity is a separate service that usually used with authentication. PK methods aid authentication but not identity.

Integrity. Data Integrity is the protection of information from unauthorized and undetected modification.

Non-repudiation. Non-repudiation associates an individual (or entity) with data such that the entity can neither deny the association nor claim modifications were made to the data (forgery).

Public key methods can support all four of the security services. Encryption supports confidentiality, while digital signature provides the basis for the remaining three services. The services based on digital signatures assume that the private key owner controls the private key and ensures its secrecy; that is, only the private key owner can use the private key. Verification of a signed message with a public key ensures that its owner signed the message. Table 1 summarizes the PK operations that provide the services.

Table 1 Using PK Methods to Provide Security Service

Security Service	Public Key Operation
Authentication	Signature
Confidentiality	Encryption
Integrity	Signature
Non-repudiation	Signature

Assuming that a party or entity maintains an association between a public key and its owner's identity, verification of a signed message authenticates the identity of the signer.[4] Following authentication, systems can use the authenticated identity to control access to information. In this situation, the PK-based authentication methods indirectly support confidentiality and integrity.

The signature verification process directly supports integrity services because it can detect (but neither prevent nor correct) unauthorized modification subsequent to the signature.

The association of the private key with its owner prevents the owner from denying that he or she was the message signer. Thus, signatures support technical non-repudiation services. Signatures cannot be denied on technical grounds. A signature that verifies with a particular public key had to be

[4] Caution; the authentication process assumes that the party authenticating the identity knows that the entity presenting the signed message is the owner of the key used for the signature. PK authentication is susceptible to replay attacks where a third party captures and reuses a previous authentication message. Asking the entity being authenticated to sign a unique message for each authentication can minimize replay attacks.

signed with the associated private key. The verification can be performed by anyone (viz., third parties) at any time.[5] However, technical non-repudiation alone may not be sufficient for legal non-repudiation where an individual can be held legally responsible for the digital signature. There may be legal grounds to deny a signature. Legal non-repudiation involves consideration of issues such as whether:

- Someone else had access to the private key despite the owner's exercising due care to protect the key.
- The owner received complete knowledge of what was being signed.
- The owner fully understood the consequences of the signature.
- The owner was fooled, misled, or under duress at the time of signature.

Security in communications with PK techniques depends on whether the user of a public key correctly knows who the key's owner is. A PKI provides a means for public key users to know who owns a key pair. A trusted third party (TTP), the ***certification authority*** (***CA***), creates a digitally signed document, a ***public key certificate*** or ***certificate***, that includes the name of the key pair's owner and the public key. The certificate includes other information related to the owner, the PKI, and the uses the CA intended for the certificate.

The PKI manages certificates. The PKI issues certificates when requested according to prescribed processes and procedures[6] that ensure the owner of the key pair is correctly identified. The owner of the public key contained in a certificate is the ***subscriber***. The PKI also provides a repository for users to obtain copies of certificates belonging to an individual and, therefore, obtain the individual's public key from the certificate. A user who obtains a public key from a certificate and depends on the association between the owner's name and the public key and on other information in the certificate is a ***relying party***. Table 2 shows how the role of the subscriber and the relying party when using PK methods.

[5] There may be a limit to the period during which the verification may occur as will be discussed later.

[6] Section 3.1 identifies sources that prescribe the processes and procedures for the DOD PKI.

Table 2 Subscriber and Relying Party Operations

Function	Subscriber	Relying Party
Encryption	Decrypts received data using private key.	Encrypts data using the receiving subscriber's public key.
Digital Signature	Encrypts (signs) data using private key.	Decrypts data using signing subscriber's public key.

The PKI also provides information on the current status of a certificate through either a ***Certificate Revocation List*** (***CRL***) or an ***Online Status Check*** (OSC). As a result of certain adverse circumstances, some certificates may become unreliable. The CA will revoke a certificate when informed that circumstances no longer warrant trusting it. The revoked certificate will be included in the next CRL that the CA issues. After an ***OSC Responder*** (OSCR) receives a CRL or other notification from the CA regarding the revoked certificate, it will respond with a revoked indication to subsequent inquires regarding the certificate.

The CRL is the older of the two status checking approaches. The CRL has a header that provides general information including:

- The CRL issuer's name which is usually the same as the CA who issued the revoked certificates.
- The ***date***[7] the CRL was produced.
- The date of the next update. The CA promises to produce a new CRL not later than the date in the CRL. The CA may produce a CRL at an earlier time. The CRL expires on the date of the next update.

The CRL lists individual revoked certificates by their serial number along with the date each certificate was revoked and may include a code indicating the reason for revoking the certificate. Certificates ***expire*** at the end of their validity period and are removed from CRLs issued after their expiration date.

The CA digitally signs the CRL. With the CA's signature CRLs can be transmitted over insecure communication links because any subsequent changes will be detected through the signature verification process.

The CA periodically issues CRLs. The CA may issue the CRLs on a periodic basis or in response to an event such as revoking a certificate because of suspected compromise of the related private key. The CA puts the CRL at location where relying parties may obtain the most current CRL. The

[7] In this document the term date denotes a value that has two components: a calendar day and a time within the day.

location is known as a ***CRL Distribution Point*** (CDP) and is usually specified in terms of a Uniform Resource Indicator (URI).

OSC is the other means of checking a certificate's validity. {RFC 2560} OSC is service that may be provided by the CA or some other TTP. A relying party sends a request to the OSC service with a certificate, the OSC service responds with a digitally signed response that includes the date and time, certificate identification, and the status of the certificate about whose validity the relying party inquired. The possible responses include "unknown" which may be the response to a query regarding an expired certificate.

Regardless of whether CRLs or OSCRs are used to check certificate status, relying parties should request (i.e., pull) the certificate status. The purpose of a certificate status check is to ensure that the certificate remains reliable. Relying parties are responsible for checking certificate status and generally have no recourse for loss resulting from using a revoked certificate.

CAs may exist in ***hierarchies***. One CA may delegate responsibilities to another CA. One CA delegates responsibility to another by issuing a certificate to the other CA. The contents of the certificate may place restrictions on the delegated CAs powers to issue subsequent certificates. The CA at the top of the hierarchy is the ***Root CA***. The Root CA has a certificate that is ***self-signed***. CAs issue certificates to individuals who cannot act as CAs and may not issue certificates. An individual who cannot issue certificates is known as an ***end-entity***.

The relying party has to establish one or more ***trust points***, which are public keys (or certificates containing them) that the relying party designated as reliable and trustworthy. The relying party should obtain the public keys (or certificates) through a reliable out-of-band method. Trust points are usually Root Certificates.[8] Trust is transitive. If the relying party trusts a CA, it also trusts other CAs to which the CA delegates its CA responsibilities.

The relying party must make a decision regarding whether or not to trust a particular certificate, the ***target certificate***. Generally, the target certificate belongs to an end-entity that either sent a signed message to the relying party or to which the relying party desires to send an encrypted message. The relying party will trust the target if there is a sequence of certificates that connect the target to one of the relying party's trust points. The sequence is known as a ***path*** or ***chain***. Construction of the path is known as ***path development***, and verification that the path provides a chain of trust is known ***path processing***. Path development and path processing

[8] Under certain circumstances a relying party may decide to trust an intermediate CA or even an end-entity.

may occur sequentially or in parallel. The path begins with a trust point and ends with the target.[9] Path processing involves:

- Verifying the signatures on the certificates.
- Verifying that the certificates ***chain***. The certificates chain if the subject on one certificate is the issuer of the next.
- Ensuring that specific use of the certificate is consistent with the intended use of the certificate as indicted by the certificate contents.
- Ensuring that none of the certificates included in the path have been revoked.

Path processing for digital signatures may require special considerations because the signature was created on a date prior to its verification. For some applications path processing may have to occur in the context of the ***effective date*** of the signature rather than the current date. Some modifications of path processing are necessary if a certificate involved in the path expired between the effective date and the current date because information on the status of the certificate would not appear in a current CRL and may not be available in an OSC response. Applications may reject paths involving expired certificates if the anticipated frequency and criticality of situations involving expired certificates do not warrant the complexity of processing expired certificates.[10]

The effective date is the date that the document was signed. The relying party may not know the exact effective date. For many situations the relying party may assume that the effective date was the date that relying party received the signed document. The relying party reliably knows that the effective date is before the date received. Assuming the effective date is the received date may cause the relying party to reject a valid signature. However, using an earlier effective date that is not reliable may result in accepting an invalid signature for which the relying party could have no recourse.

There may be a delay between the time when a certificate is revoked and the time when either the CRL is revised or the OSCR is informed of the revocation. As a result, there is a chance that a relying party may rely on a certificate after it is revoked but before CRL update or OSCR notification occurred. Some relying parties may have to be concerned about such events. Two approaches are to:

---

[9] The choice of chain orientation is discretionary. The chain could be ordered from end-entity to a trust point. Ordering is not important as long as the path processing results are t he same.

[10] Processing of paths involving expired certificates would require access to archives of old certificates and previously issued CRLs.

- Hold processing in abeyance pending receipt of a CRL that would have to include any certificate involved. Many CAs including the DOD PKI CAs specify a maximum time between being notified of a certificate revocation request and issuing a CRL that includes the certificate. This approach may prevent accepting an invalid signature but involves delays in processing which may not be practical in situations such as authenticating near real-time access to systems or facilities.
- Review CRLs to determine whether a certificate newly added to the CRL was processed subsequent to its revocation and identify such certificates for special processing. This approach does not prevent accepting invalid signatures but provides after-the-fact detection.

Certificates have a lifetime.[11] The period that certificates are valid is determined by the contents of their validity field. The DOD PKI issues certificates for various lifetimes. Certificates for users generally have either a two or three year lifetime. CA certificates have a longer lifetime. The lifetime of a CA certificate has to encompass the lifetime any certificate that it issues. This nesting of certificate lifetimes is necessary for path processing.

The limited life of certificates may impact some applications. Certificates belonging to CAs and other entities expire and have to be replaced. CAs may have multiple certificates. The need to nest certificate lifetimes means that CAs cannot issue certificates when a validity period of a new certificate would exceed the validity period of their own certificate. For example, if a CA has a certificate with a five year lifetime and issues certificates for users with a maximum lifetime of three years, the CA can only use its certificate[12] to create new user certificates for two years. The CA has to obtain a new certificate when it can no longer issue new subscriber certificates related to its old certificate. Applications that process certificates issued by a CA may have to find one particular certificate among several issued to a CA. For example, consider the case of two users who received certificates valid for three years from the CA mentioned above. If the users received their certificates more than two years apart, an application will use a different CA certificate to verify their digital signatures.

Entities such as individuals that are not CAs may also have multiple certificates. Individuals may have different certificates for different purposes. Some applications may have to be able to select an appropriate certificate from the multiple certificates. This situation may be compounded for applications that have to store information over a long period of time and

---

[11] Reasons for limiting the lifetime include controlling size of the CRL and the risk that the private key will be discovered.

[12] The CA actually creates certificates with the private key associated with the public key in its certificate.

may have to verify signatures using long expired certificates.[13] Such applications may have to be able find the relevant certificate among multiple certificates belonging to an entity.

When a certificate expires, the subscribers and relying parties must react appropriately to obtain new keys and certificates and retire old keys and certificates. The subscriber may need to obtain a replacement certificate. Replacements may be through ***renewal*** or ***rekey***. A renewal certificate contains the same public key as the original certificate and is identical to the original certificate except for the serial number and validity period. A rekey certificate involves a different public key; consequently, relative to the original certificate it has a different public key, serial number and validity period. The DOD PKI does not currently renew certificates.

Subscribers should not use the private key associated with an expired (not renewed) certificate for digital signatures and should destroy the private key including any copies. Although relying parties should not use expired certificates to verify signatures for the first time, they may need the certificates to independently confirm the accuracy of (i.e., reverify) previously verified signatures. Subscribers should retain access to private keys as necessary to decrypt any information retained in encrypted form. For example, the subscriber would need the private key to decrypt and view any previously received messages still retained in encrypted form.[14] Because expired certificates do not appear in CRLs, determining the status of an expired certificate may be complicated as described in the previous discussion of path processing. If an application's use of expired certificates is infrequent and not critical, the application should reject paths involving an expired certificate. Table 3 summarizes the handling of keys and certificates when certificates expire.

---

[13] The assumption here is that encrypted information will not have to be retained for long periods or will be re-encrypted with a new key because of increased risks that the original encryption key will be discovered.

[14] The private key may also be needed to view messages that the subscriber sent encrypted because mail clients often retain the messages as encrypted text rather than as clear text for security reasons. Effectively, the clients include the subscriber as an addressee for encrypted messages that the subscriber sends.

Table 3 Key and Certificate Disposition at Certificate Expiration

Application	Private Key	Public Key and Certificate
Digital signature (Authentication, integrity, non-repudiation)	Destroy; prevent any subsequent use.	Cease use for any new (first time) signature verifications; cannot determine validity. Retain for non-repudiation applications for as long as any previously verified, signed data is retained and remains subject to verification. Must consider providing measures to prove receipt prior to certificate expiration (or revocation) and no subsequent modification after receipt.
Encryption (Confidentiality)	Retain personal and data recovery copies of key while maintaining encrypted stored data. Consider re-encrypting if storage period exceeds the key lifetime.	Destroy; cease use.

The DOD has a ***key recovery*** policy to ensure that encrypted information can be recovered for law enforcement purposes and for operational continuity. Copies of key pairs used for encryption must be provided to the DOD PKI's Key Recovery Manager (KRM). Keys used for encryption may not be used for non-repudiation, and keys used for non-repudiation may not be used for encryption.

Applications in the course of providing security services to their users perform in the capacity of both the subscriber and the relying party. The applications perform the functions described above as appropriate in the application's operating domain. The remainder of this document describes the requirements to PK-enable applications for use with the DOD PKI.

3.0 APPLICABLE DOCUMENTS AND SCOPE

This section sets the context for the technical requirements for enabling applications to use the DOD PKI. The applications must comply and be consistent with the overall DOD policy for application assurance levels and the use of the PKI. The following subsections identify the documents providing policy and technical requirements that impact applications and the types of applications that can use the DOD PKI.

3.1 Applicable Documents

This section identifies key documents that provide overall policy and guidance for the DOD PKI including its operations and technical functions. The following documents are incorporated by reference. The documents appear in order of priority. Applications developers shall comply with the requirements of these documents as well as this document. The acronym inside square brackets ([ ]) that appears before each reference will be used in the remainder of the document to refer to the associated applicable document. The applicable documents are:

[CP] Department of Defense. *X.509 Certificate Policy for the United States Department of Defense*. 13 December 1999.

[CPS] Department of Defense. *Defense Information Infrastructure Certificate Management Infrastructure Certification Practices Statement for Class 3 Assurance*, Draft 1.6. 30 March 2000

[OSD] Deputy Secretary of Defense Memorandum, *Public Key Enabling of Applications for the Department of Defense Public Key Infrastructure (PKI)*, Draft Policy, 24 November 1999.

[CONOPS] Department of Defense. *DoD Information Infrastructure Public Key Infrastructure (PKI) Concept of Operations*, Third Draft, 24 October 1997.

[IF] Department of Defense. *Department of Defense Class 3 Public Key Infrastructure Interface Specification*, Draft Specification, 13 January 2000.

[ECA] Defense Information Systems Agency, *Interim External Certification Authority Applications Guide*.

[FPKI] National Institute of Standards and Technology. *Federal Public Key Infrastructure (PKI) Version 1 Technical Specifications, Part B - Technical Security Policy*, TWG 96-20, Gaithersburg, MD, March 1996.

[FIPS 46] National Institute of Standards and Technology. *Data Encryption Standard (DES)* . FIPS PUB 46-3. October 25, 1999.

[FIPS140] National Institute of Standards and Technology. *Security Requirements for Cryptographic Modules.* FIPS PUB 140-2. November 1999.

[FIPS180] National Institute of Standards and Technology. *Secure Hash Standard (SHS).* FIPS PUB 180-1. April 17, 1995.

[FIPS186] National Institute of Standards and Technology. *Digital Signature Standard (DSS).* FIPS 186-2. January 27, 2000.

[FIPS196] National Institute of Standards and Technology. *Entity Authentication Using Public Key Cryptography.* FIPS 196. February 1997

[RFC 2459] Housley, R., W. Ford, W. Polk, and D. Solo. *Internet X.509 Public Key Infrastructure Certificate and CRL Profile.* January 1999.

3.2 PKI Assurance Levels

The level of assurance associated with a public key certificate is an assertion by a CA of the degree of confidence that a Relying Party may reasonably place in the binding of a Subscriber's public key to the name and attributes asserted in the certificate. The Federal PKI (FPKI) Security Policy [FPKI] defines multiple assurance levels.

The multiple levels have increasing requirements for issuing certificates, protecting private keys, and protecting certification authorities, and it is generally accepted that a single level of assurance for the PKI will not suffice for all applications. Some applications that are less critical or of lower monetary value can stand greater risk; other applications require a more robust PKI.

The DOD PKI follows the Federal guidelines to the extent feasible and intends to field medium and high assurance levels by supporting the Class 3 and Class 4 Assurance Levels as described in the following subsections.

3.2.1 Class 3 Assurance Level (C3)

The DOD X.509 Certificate Policy [CP] specifies that Class 3 DOD certificates are intended for:

Applications handling unclassified medium value information in Moderately Protected Environments, unclassified high value information in Highly Protected Environments, and discretionary access control of classified information in Highly Protected Environments. This assurance level is

appropriate for applications that require identification of an entity as a legal person, rather than merely as a member of an organization.

And in particular for:

1. Digital signature services for mission critical and national security information on an encrypted network;
2. Privacy and authentication in support of access control security services (e.g., separation of communities of interests) for access to classified Special Compartmented or Special Access information on networks protected using NSA approved CLASS 1 cryptography appropriate to the data being protected, or on networks that are physically isolated and approved to process the classified data.
3. Technical non-repudiation for medium value financial or electronic commerce applications such as payroll, contracting, vehicle purchases, etc.

In this document the term DOD PKI refers to the Class 3 PKI.

3.2.2 Class 4 Assurance Level (C4)

The DoD X.509 Certificate Policy [CP] specifies that Class 4 certificates are intended for applications handling high value unclassified information (Mission Critical, NSSI) in Minimally Protected environments.

And in particular for:

1. All applications appropriate for Class 3 certificates.
2. Digital signature services for unclassified mission critical or national security information in an unencrypted network.
3. Protection (authentication and confidentiality) for information crossing classification boundaries when such a crossing is already permitted under a system security policy (e.g. sending unclassified information through a High Assurance Guard (HAG) from SIPRNET to NIPRNET).
4. Technical non-repudiation for large value financial or electronic commerce applications.

4.0 REQUIREMENTS

This section describes the specific functions that PK-enabled applications must provide. Services have to be selected and tailored to the needs of the application. Applications may either provide the needed functions internal to the application or ensure that the application operates in an environment where there are other external applications or services to provide the functions. Applications shall satisfy the requirements below regardless of whether or not the services are internally or externally provided. Organizations sponsoring an application for DOD use are responsible for ensuring that the application in its operational environment satisfies the requirements of this section.

4.1 Requirements Tailoring

The requirements of this section may be tailored to the particular needs of an application. There are many application dependent issues that may affect the application. Organizations responsible for obtaining or developing the applications may tailor the requirements to meet their specific application's needs.

4.2 General Requirements

The following general requirements apply broadly to the application.

4.2.1 Automation Preferred over Procedures

There are alternative methods for satisfying the requirements described below. Some requirements may be satisfied using either automated features or manual procedures. Applications should, whenever possible, provide automated features. When not possible, applications shall include appropriate instructions in related documents such as configuration guides and user and administrator manuals (or automated equivalents thereof such as help files).

4.2.2 Use of Evaluated Cryptographic Modules

Applications should use Cryptographic Modules evaluated under the Federal Information Processing Standard (FIPS) 140 [FIPS 140]. Successful evaluation satisfies many of the requirements contained herein. Because applications may employ several cryptographic functions, applications may depend on multiple modules. For example, a hardware token used for public key operations and software used for symmetric encryption may be in separate modules.

4.2.3 Computer Security

Applications shall provide a secure computer environment for the application's operation. Applications should ensure that measures exist and are employed to:

- Identify and authenticate users of the application.
- Control access to the application's critical cryptographic objects and functions based on the user's identity and authorization for the access. Cryptographic objects include keys, trust points, and certificates. Access to private keys shall be limited to authorized individuals (entities).
- Maintain audit records regarding the use of the application's cryptographic features.
- Protect the application's cryptographic objects and functions from tampering.
- Ensure the separation of encrypted and unencrypted information.

4.3 Specific Requirements

Application functions can be segregated into several related families. These families are:

- Key management that includes functions for generating and key pairs, storing keys, and maintaining and storing trust points. Security of private keys and trust points is critical.
- PKI interface that includes the functions to use the services of the PKI.
- Encryption services that include the functions to encrypt and decrypt information using both symmetric and asymmetric encryption algorithms and to calculate message digests. These services also include generating symmetric keys and random numbers.
- Relying party processing that includes functions to obtain certificate chains to include checking the validity of certificates in the chain.

The following subsections describe requirements for each family of functions.

4.3.1 Key Management

Applications shall perform key management functions as necessary. The key management functions are concerned with maintaining and protecting persistent cryptographic objects. These functions include:

- Generating asymmetric (public and private) key pairs.

- Storing the key pair and related certificates. The key storage service has to protect the private key from compromise or loss.
- Storing and protecting certificates that are trust points.
- Escrowing or making a copy of keys used for encryption for data recovery purposes.
- Importing and exporting key pairs and possibly related certificates.

The following subsections address each of these issues.

Although generation and protection of symmetric keys is important, these topics are addressed in Section 4.3.3 because these keys are either not persistent or are protected as part of another cryptographic object.

Applications should use modules evaluated under [FIPS 140], Level 1. Applications should use features of evaluated modules to satisfy the requirements in the remainder of this section for generating, storing, and using private keys. If the application uses different asymmetric algorithms than used by the DOD PKI, services related to the additional algorithms should also be provided by [FIPS 140] evaluated modules. Evaluated modules' features satisfy this section's requirements for generating, storing, and using private keys.

4.3.1.1 Generating Key Pairs

This section applies to key pairs whose public key will be contained in a DOD PKI issued certificate and does not apply to key pairs whose public key will not be contained in a DOD PKI issued certificate.

Applications shall not generate key pairs and include the public key in a request for a DOD PKI certificate for a subscriber who is an individual. Applications shall use key pairs and certificates created for individuals using DOD PKI methods and procedures. See Section 4.3.1.5.

If the application needs to generate its own key pair for use by entities other than individuals, the generation process should follow the appropriate guidance and standards for the algorithm selected. Table 4 lists public key algorithms and the source for information on generating keys for them. Applications that use other algorithms shall provide appropriate means to generate key pairs for their application.

Applications should generate keys whose length is at least 1024 bits for RSA and DSA.

Applications that generate keys must use a random source to generate the keys. Applications should use a generator that meets the requirements of Section 4.3.3.4.

Table 4 Algorithms and Key Generation

Algorithm	Generation
Rivest, Shamir, and Adleman (RSA)	[FIPS 186]
Digital Signature Algorithm (DSA)	[FIPS 186]
Key Exchange Algorithm (KEA)	{KEA}

Applications shall generate and submit certificate requests in accordance with methods described in the [IF].

Private keys generated for actual or possible encryption use shall be provided to the DOD Key recovery system in accordance with the [IF]. All copies of private keys generated for actual or possible non-repudiation purposes shall remain under the owning entity's sole control. Applications should provide entities with a capability to save and restore non-repudiation private keys. This capability shall be under the sole control of the entity.

4.3.1.2 Storing Keys and Related Certificates

Applications shall protect private keys. The application should meet the requirements for Level 1 of [FIPS 140]. If the module has not been evaluated under [FIPS 140], an assessment should be made of the module's ability to meet the requirements.

If the application performs operations with the private key in software, the application shall encrypt the private key when not in use. The unencrypted private key should exist in memory for the minimum time necessary to perform private key operations. All copies of the unencrypted private key should be destroyed (e.g., overwritten) when the private key operation is complete.

If access to or use of private keys is protected through passwords, the password should be randomly selected from a space of at least 2^{56} possible passwords unless there is a means to detect and protect against deliberate attempts to search for passwords.

When applications operate on systems where there are not strict controls on other software that may coexist on the system, applications shall protect the private key against surreptitious use by malicious software. For example, the application may provide a user interface that informs the user that there is a pending request for access to the private key and identify the requesting application. Such an interface should provide the opportunity to allow or deny the use of the key.

Applications should be able to maintain multiple key pairs for an owner.

Applications shall store certificates for a subscriber. Applications should be able to maintain multiple certificates containing the same public key. The certificates may have different issuers, different uses, or different validity periods.

Applications should be capable of maintaining private keys from key pairs used for data or key encryption after the expiration of related certificates for the purpose of decrypting any residual encrypted information.

Applications should be capable of maintaining public keys or certificates from key pairs used for digital signature after the expiration of related certificates for the purpose of verifying signatures on any residual signed information.

4.3.1.3 Storing Trust Points

Applications shall provide a capability to manage and store (e.g., add, modify, or delete) trust points. Applications should restrict access to and use of this capability to select individuals or organizations as appropriate for the specific application. For example, some applications may:

- Allow a organization responsible for distributing the application to embed the trust points prior to releasing the application to its user community.
- Allow only the application user to manage trust points.
- Allow only the network administrator to manage trust points.
- Accept only authorized trust points where authorization could be determined by virtue of a digital signature on the list of trust points.[15]

4.3.1.4 Data Recovery

Applications should provide a capability for authorized individuals who have recovered a private key from the DOD PKI key recovery management system to decrypt information that the application originally encrypted (See [IF]).

4.3.1.5 Importing and Exporting Keys

PKCS #12, *Personal Information Exchange Syntax Standard* {PKCS 12}, is an industry standard for importing and exporting keys and related certificates. The DOD PKI supports this standard (see [IF]) and can provide PKCS #12 files to import keys and certificates to the application's key management environment.

---

[15] This approach requires the prior existence of a designated trust point for signing the list of trust points.

Applications using standard DOD PKI certificates for individuals shall be capable of importing keys and certificates in accordance with the [IF]. Other applications should be capable of importing keys and certificates issued by the DOD PKI in accordance with the [IF]. Applications that generate their key pairs should be capable of exporting generated keys and certificates in accordance with the [IF].

4.3.2 PKI Interface

The application may have to interact with the PKI to request and obtain a certificate to hold the public key for the application user as well as to obtain certificates for communicating with other PKI subscribers. This section concentrates on the interface with the PKI and identifies requirements for applications to interact with the PKI interface. The [IF] describes the PKI interface details. Section 4.3.1.5 provided application requirements for importing and exporting subscriber keys pairs and related certificates. Many applications will interact with the PKI to obtain certificates and certificate status for path processing in support of relying party operations. The process of using the certificates and status check results in relying party path processing is the subject of Section 4.3.3.

4.3.2.1 Communication Protocols

Applications must use the Lightweight Directory Access Protocol (LDAP), the Hypertext Transmission Protocol (HTTP), or the Hypertext Transmission Protocol over SSL (HTTPS) when communicating with the DOD PKI. The specific interaction with the DOD PKI determines which protocol the application must use. LDAP is adequate for most typical applications. However, selected interactions with the DOD PKI require the use of HTTPS. Applications that have to use HTTPS must be capable of communicating using the SSL versions and ciphersuites that the DOD PKI accepts (See [IF]).

4.3.2.2 Requesting and Obtaining New Certificates for Subscribers

As stated in Section 4.3.1.1, only limited applications may be permitted to generate key pairs and request certificates. Applications must use HTTPS to request and obtain certificates for the subscriber. The DOD PKI accepts prepared requests and subsequently provides requested certificates according to industry standards. Certificate requests shall be in accordance with the [IF]. Certificates generated in response to requests will be available for retrieval in accordance with the [IF]. Applications shall be capable of retrieving and accepting requested certificates in accordance with the [IF].

4.3.2.3 Retrieving Certificates

Many applications need to obtain or store certificates belonging to other entities for purpose of interacting or communicating with them. Relying

parties need certificates to obtain an entity's public key for the purpose of performing public key operations. Some applications may provide alternate methods to obtain needed certificates. For example, an application employing digital signatures may include relevant certificates with each signed message and, thereby, avoid having to retrieve the certificates from the DOD PKI's certificate repository.

The DOD PKI maintains certificates in its directory. The directory has entries for all subscribers including the CAs. Certificates for the DOD PKI CA's are in the CA's directory entry. The [IF] provides details on the interface and the criteria for searching for certificates. Appendix C provides examples of directory entries.

Applications needing to interact with other entities shall have the capability of requesting and accepting certificates from the DOD PKI unless the application provides alternative means. Such applications shall be able to request and accept certificates over LDAP in accordance with the [IF]. Applications may provide an ability to retain certificates for subsequent use.

4.3.2.4 Checking Certificate Status

Applications supporting relying party operations generally need to verify the validity of unexpired certificates being used. There are two general approaches to status checking: CRLs and OSCs. Using CRLs is the older approach, while OSC is newer. Applications must be capable of requesting and accepting information regarding status of certificates upon which the application relies. Applications shall be able to check certificate status using CRLs. Applications may use OSCs to check certificate status when the DOD PKI has operational OSCRs.

4.3.2.4.1 Retrieving CRLs

Applications operating in environments with network connectivity to a CRL distribution point should be able to obtain a current CRL. Applications should be able without user intervention to obtain a current CRL to check the status of a certificate that contains a CRL distribution point extension (see [IF]). Applications with network connectivity unable to automatically find CRL distribution points should be capable of being configured with a distribution point that the application then uses to obtain CRLs as needed. Applications shall be capable of accepting a CRL for use in certificate status checking.

4.3.2.4.2 Status Checking with an OSC Responder

Applications may use an OSC responder to check the status of a particular certificate when the DOD has an operation responder. Applications shall

prepare and transmit the request to the responder using HTTP in accordance with the IF. The application shall be able to accept OSC responses.

4.3.2.5 Retrieving Certificates and CRLs from the Archive

The requirements of this section apply to applications needing to obtain old certificates (i.e., certificates that have expired or been revoked) or old CRLs (i.e., CRLs whose next update is after the current date). Applications should be capable of requesting old certificates and CRLs from the DOD PKI archive over HTTPS in accordance with the [IF]. Applications shall be capable of accepting needed old certificates and CRLs.

4.3.3 Encryption Services

Applications need to perform various encryption functions. The functions that an application must perform are application dependent. PK-enabled applications have to perform operations with the public and private keys. The applications also will likely have to perform symmetric encryption or prepare message digests because asymmetric operations are not generally used on bulk data. Standards describe how individual algorithms operate. There are multiple standards for some algorithms.[16] Since the algorithms typically operate on fixed size, blocks of data, the standards usually prescribe the method to pad the data when the last block is not full.

The DOD PKI places no restrictions on applications in terms of the specific algorithms that the application must use or support other than those necessary to process the DOD PKI certificates, CRLs, and OSC responses. In general, the DOD PKI supports the use of algorithms that the FIPS prescribe for Government use. Table 5 shows algorithms that are FIPS approved and used by the DOD PKI.

[16] The variations often involve issues such as key generation and padding conventions. These variations can lead to incompatibilities. Specifically, there are variations in the RSA algorithm regarding both of these issues. The variations are not inherently interoperable because of differences in the padding conventions. There are also differences in the Triple DES algorithm regarding the number of keys (2 or 3) used in the three applications of DES.

Table 5 FIPS Algorithms and PKI Use

Algorithm Function	FIPS Approved Algorithms	DOD Class 3 PKI Algorithms[17]
Message Digest (Hash)	Secure Hash Algorithm 1 (SHA-1)	Secure Hash Algorithm 1 (SHA-1)
Digital Signature	Digital Signature Algorithm (DSA), Rivest, Shamir, Adleman (RSA) (PKCS #1 and X9.31 versions)	RSA (PKCS #1)
Key Exchange	None	RSA (PKCS #1)
Symmetric encryption	Triple Data Encryption Algorithm (TDEA), DES	TDEA

Applications should use cryptographic modules approved under [FIPS 140], Level 1. If the module has not been evaluated under [FIPS 140], an assessment should be made of the module's ability to meet the requirements of both [FIPS 140] and the FIPS governing the individual algorithm. Applications using [FIPS 140] evaluated modules satisfy the requirements of this section for operations included in the evaluation.

4.3.3.1 Asymmetric Services

Applications shall be capable of performing public key operations necessary to verify signatures on DOD PKI signed objects (viz., certificates, CRLs, and OSC responses). Applications shall perform operations with asymmetric keys as necessary for the individual application.

Applications should use FIPS approved algorithms provided by [FIPS 140] approved modules for application specific functions.

4.3.3.2 Symmetric Services

The need to provide symmetric services is application dependent. Applications that support encryption generally need capabilities to encrypt bulk data using symmetric encryption.

---

[17] The Class 3 PKI does not constrain the algorithms that the application may use. However, all certificates will be signed using the indicated algorithms and applications must use those algorithms to verify the validity of certificates and certificate chains. Interaction with the PKI over SSL will require applications to support ciphersuites used by the DOD PKI ([IF]).

Applications that interact with the DOD PKI using SSL (i.e., HTTPS) must be capable of encrypting and decrypting data using the Triple Data Encryption Algorithm (TDEA). (See the [IF].)

Applications needing symmetric services should use algorithms from [FIPS 46].

Applications should use FIPS approved algorithms (TDEA or DES) provided by [FIPS 140] approved modules for application specific functions. TDEA is the recommended symmetric algorithm.

Applications using symmetric encryption must be capable of generating random symmetric encryption keys. Applications should use a random number generator when composing a key. Section 4.3.3.4 has the requirements for random number generators. The length of symmetric encryption keys shall be at least 56 bits and should be at least 100 bits.

Applications shall protect symmetric keys for the life of their use. An unencrypted key should exist in memory for the minimum time necessary to perform operations using the key. All copies of an unencrypted key should be destroyed (i.e., overwritten) when an operation is complete. Applications shall encrypt keys when not in use.

4.3.3.3 Digest Services

Digest services provide the basic functions to create message digests used in applications involving digital signature. SHA[18] is FIPS approved. [FIPS180]

Applications shall be capable of producing SHA digests of messages to support verification of DOD PKI signed objects.

Applications should use FIPS approved digest or hash algorithms (SHA) provided by [FIPS 140] approved modules for application specific functions.

4.3.3.4 Random Number Generators

A Random Number Generator (RNGs) is a critical but sometimes overlooked component of the basic encryption services. Applications employing cryptography need an RNG to create both symmetric and asymmetric encryption keys. The DSA also needs a RNG to generate digital signatures.

Applications should use the algorithm described in [FIPS 186] to generate random numbers. Individual instances of an application should have unique and random initial seeds for their RNG. The application may be initially configured with a random seed unique among other instances or generate a

[18] SHA refers to the current algorithm version specified in the referenced version of the Secure Hash Standard [FIPS 180] (i.e., SHA-1).

random seed based on random events or values in the instance's operating environment.

4.3.4 Path Development and Path Processing

Applications supporting relying party processes must be capable of developing a certificate path and processing the path. The path development process produces a sequence of certificates that connect a given end-entity certificate to a trust point. Path processing determines the validity of the path in the context of the intended use of the end-entity certificate. The requirements for each of these processes are the subject of the next two subsections. Although the discussion describes development and processing as separate and sequential processes, the processes may be integrated as long as the determination of path validity meets the requirements of this section. This section does not preclude applications from developing and storing paths prior to their use for path processing.

The DOD PKI anticipates eventual participation with the FPKI and its Federal Bridge Certification Authority (FBCA) . Future versions of this document may include requirements to interoperate with the FBCA. Specifically, applications in the future may have to develop and process paths in the FBCA environment.

4.3.4.1 Path Development

Path development involves collecting certificates and ordering them in a chain from a trust point to the given end-entity. Applications should construct paths automatically without involvement of a human.

Over time both CAs and end-entities may have multiple certificates. CAs may have more than one certificate valid at a given point in time. End-entities will also have multiple valid certificates. For example, end-entity certificates may have different intended uses. End-entity certificates contain information such as the issuer's name and optional extensions that is useful in finding the previous certificate in the chain, issuer alternate name and authority key identifier. The DOD PKI populates these extensions for some certificates. Applications should automatically construct paths when a path exists.

4.3.4.2 Path Processing

This section provides general path processing requirements. Appendix D supplements the requirements of section with a path processing algorithm and requirements for the use of certificate extensions that the PKI uses.

Path processing depends on two dates, the effective date and the current date. The effective date is when the transaction was initiated. For encryption uses, the effective date and the current date are the same. For signature

uses, the effective date is the date the subscriber created the signature. The effective date must not follow the current date. Applications should consider the current date to be the effective date for signature applications unless there is reliable evidence to establish an earlier effective date.

Applications should reject a path where an included certificate expired between the effective date and the current date unless the application is re-verifying a signature that was verified at an earlier date when none of the involved certificates were expired. When an application must validate a path involving expired certificates, the application must check the status of using CRLs issued after the effective date but prior to the expiration of a currently expired certificate and should use the most current CRL preceding a certificate's expiration.

There are several steps to perform in path processing. Appendix D describes the path processing in more detail. Applications must perform the major steps including:

- Verifying certificate signatures. This verification shall use the certificate issuer's public key.
- Ensuring effective date falls within the certificate's validity period.
- Ensuring certificate use is consistent with extensions. See Appendix D.
- Ensuring the validity of certificates through a status check. The following section will address status checking.

The steps above have to be performed for each certificate in the chain. The process terminates either when the chain tracks froma trust point to an end-entity or a problem that prohibits validation of the chain occurs. The chain validations shall succeed in the former case and shall fail in the latter case.

4.3.4.3 Certificate Status Checking

Processing the certificate chain involves checking the status of certificates in the chain to ensure that none has been revoked. Status checking involves use of a CRL or an OSC responder. This section requires the use on an unexpired CRL or OSC response but does not require the use of the most current CRL or OSC response. Requirements for currency of CRLs or OSC responses are application dependent.

There are several steps applications must perform in processing a CRL. These steps must be performed for each certificate in the chain. The target certificate is the certificate whose validity is being verified. Applications shall:

- Verify the signature on the CRL. This verification requires developing and processing a certificate path establishing that the target

certificate's issuer trusted the CRL issuer to issue CRLs. See Appendix D.

- Verify that the CRL has not expired if the target certificate has not expired. A CRL is expired if the current date is after the CRL's next update field value. (See above paragraph regarding paths with expired certificates.) If the target certificate has expired, verify that the CRL's issue date follows the effective date and precedes the certificate's expiration date.
- Search the list of revoked certificates to determine that the target certificate either is not included or its revocation date is after the effective date.

Status checking for applications using an OSC responder involves a different set of steps. Applications using OSC responses shall:

- Verify the signature on the OSC response. This activity includes developing and processing a path that establishes that the certificate issuer or a trust point trusted the responder for the express purpose of issuing responses. See Appendix D.
- Verify the response indicates the certificate is valid.

If the status check fails any of the above checks, then the path shall be rejected.

4.3.4.4 Extension Processing

Applications shall ensure that the intended use of the certificate is consistent with the extensions. Applications must process the critical extensions and should process non-critical extensions.

Applications shall ensure that in the Key Usage extension in the end-entity certificate:

- The digital signature bit is set for authentication uses.
- The non-repudiation bit is set for non-repudiation uses.
- The key encipherment bit or the data encipherment bit is set for encryption uses depending on whether keys or data are to be encrypted.

Applications shall ensure that for certificates other the Root CA in the Key Usage extension:

- The Certificate Signature bit is set in the certificate containing the public key used to sign the next certificate in the chain.
- The CRL Signature bit is set in the certificate containing the public key used to sign a CRL.

Applications shall ensure that Basic Constraints extension is true in the certificate containing a public key used to sign a subsequent certificate in the path.

Appendix D contains more detailed requirements regarding extensions.

4.4 Application Configuration

Applications must identify the operating conditions required of the application's operating environment. Applications must identify all necessary conditions and dependencies for the application to securely perform its functions. Specifically, applications must identify dependencies on supporting computer systems (e.g., processor, primary and secondary memory capacity), operating systems (e.g., Version and release numbers), subsystems (e.g., cryptography toolkits), and peripherals (e.g., network connection and speed, card readers, hardware tokens).

Applications shall be capable of being configured to operate with the DOD PKI. Applications should be able to automatically operate with the DOD PKI with minimal configuration. Applications should be able to self-configure as much as possible. For example, the applications could detect the interface to the PKI for obtaining CRLs by looking for a certificate extension, the CRL Distribution Point, which has a URI to retrieve the CRL. Applications should be capable of being centrally managed as much as possible. Trust points should be centrally administered. Applications should be designed to permit remote updating or modification of the application from a network site. Such updates should preserve security. For example, sites should be authenticated or updates signed by an authority trusted to supply updates.

Applications shall be capable of being configured to operate with only DOD PKI trust points.

Applications must be capable of being configured for secure operation in their intended environments. Applications should be capable of automatic configuration and report any deficiencies that preclude complete configuration. Applications should be able to verify that their operating environment is properly configured and report any deficiencies.

If automated features to initially configure and subsequently maintain application configuration are not feasible or practical, manual procedures by administrators and end-users shall be required. In this case, applications must provide procedures that are well documented and easily followed and ensure that administrator and user training addresses the procedures.

4.5 Application Documentation

Applications shall include user and administrator manuals (or electronic equivalents) that instruct personnel with little advance knowledge of public

key cryptography on the proper and secure configuration and use of the application.

Documents shall include instructions for configuring the application to interoperate with the PKI to include:

- Installing DOD PKI trust points.
- Removing non-DOD PKI trust points.
- Generating a key pair and requesting and obtaining certificates or importing existing keys and certificates.
- Installing URIs for DOD PKI services such as obtaining certificates for other entities and performing status checking.
- Selecting encryption algorithms. Selections should indicate algorithms that must be used, may be used, or cannot be used.
- Configuring the application for SSL access to the DOD PKI if necessary.

Documents shall instruct users and administrators regarding their responsibilities as PKI users to include:

- Instructions on technical and procedural measures to protect the private key against compromise and misuse.
- Guidance on the actions to take when there is suspected compromise of key (e.g., a token has been lost).

5.0 QUALIFICATION REQUIREMENTS

The section identifies the requirements for verifying that the application meets its technical requirements. The purpose of the verification is to ensure the application interoperates with the DOD PKI and is secure. The following subsections describe the verification methods, interoperability verification, and security verification. Interoperability requirements focus on the methods used to determine an application is capable of interacting with the DOD PKI. The security assurance requirements identify the methods that will be used to determine that an application adequately protects information that it processes and maintains.

5.1 Verification Methods

Each requirement shall be verified through one of four methods: Analysis, Demonstration, Test, or Inspection. These methods are defined in Table 6.

Table 6 Requirements Verification Methods

METHOD	DEFINITION
Demonstration	The operation of the application, or a part of the application, that relies on observable functional operation not requiring the use of instrumentation or special test equipment.
Test	The operation of the application, or part of the application, using instrumentation or special test equipment to collect data for later analysis.
Analysis	The processing of accumulated data obtained from other qualification methods. Examples are reduction, interpolation, or extrapolation of test results.
Inspection	The visual examination of application components, documentation, etc.

5.2 Interoperability Verification

This section describes efforts to verify whether an application is able to interoperate with the DOD PKI. Table 7 lists the verification approach for each of the interoperability-related requirements from Section 4.0. The stated demonstration and test activities refer to use of the DOD PKI. These activities may involve either the operational or test DOD PKI. The Government will make the determination at the time of the test.

Table 7 Requirements Verification

Section	Verification Approach
4.3.1.5	Importing and Exporting Keys: The ability of the application to import keys associated with standard certificates for individuals shall be demonstrated. The application shall import at least one set of keys and certificates for each certificate type supported by the application. The applications shall demonstrate interoperability by performing representative subscriber and relying party operations with each certificate type and its related keys. The application shall demonstrate the ability to export keys and certificates. The correctness of the exported file shall be verified through analysis.
4.3.1.1	Generating Key Pairs. If the application generates subscriber keys, the application shall demonstrate the ability to generate keys, request new certificates, and obtain new certificates through interaction with the DOD PKI. If the generated keys are for encryption applications, the application shall demonstrate its ability to provide keys to the DOD PKI KRM.
4.3.1.3	Storing Trust Points. The application shall demonstrate its ability to store DOD PKI trust points.
4.3.1.4	Data Recovery. If the application generates keys for encryption uses, the application shall demonstrate its ability to recover a key provided by the DOD PKI KRM. The recovered key shall be the one provided to the KRM in the demonstration related to Section 4.3.1.1.
4.3.2.1	Communication Protocols. The application shall demonstrate its ability to communicate with the PKI using LDAP and, as necessary, HTTP and HTTPS. These demonstrations may be performed in conjunction with other demonstrations
4.3.2.2	Requesting and Obtaining New Certificates for Subscribers. The application shall demonstrate its ability to request and obtain new certificates for subscribers. This test may be performed in conjunction with tests for Section 4.3.1.1.
4.3.2.3	Retrieving Certificates. The application shall demonstrate its ability to retrieve certificates and use them in relying party operations. This demonstration may be in conjunction with other verification activities.

Section	Verification Approach
4.3.2.4	Checking Certificate Status. The application shall demonstrate its ability to check certificate status. This demonstration may determine the status of an individual certificate and in the context of path processing. The correct operation of the application shall be verified by observing responses or analyzing the response received by the application.
4.3.2.4.1	Retrieving CRLs. The application shall demonstrate the ability to retrieve a CRL. The correct operation may be verified by analyzing the retrieved copy of the CRL, analyzing activity at the test DOD PKI systems, or through the demonstrating actions of the application after using the CRL. If the latter is used, the demonstrations shall show that the application responds appropriately to valid and revoked certificates. The application shall demonstrate its ability to obtain all needed CRLs.
4.3.2.4.2	Status Checking with an OSC Responder. The application shall demonstrate the ability to retrieve an OSC response. The correct operation may be verified by analyzing the retrieval's results, analyzing activity at the test DOD PKI systems, or through demonstrating actions of the application after using the response. If the latter is used, the demonstrations shall show that the application responds appropriately to valid and revoked certificates. The application shall demonstrate its ability to obtain responses for certificates at all levels of the DOD PKI hierarchy.
4.3.2.5	Retrieving Certificates and CRLs from the Archive. If the application must retrieve certificates or CRLs from the archive, the application shall demonstrate its ability to retrieve them as appropriate. The correct operation shall be verified by analysis of activity on PKI systems, by analysis of the internal records or the application after the retrievals, or through demonstrations of the application behavior in response to the retrievals. If verification involves the latter, the demonstrations shall involve several cases to demonstrate the applications ability to perform correctly based on the results of the retrievals.

Section	Verification Approach
4.3.4	Path Development and Path Processing. Although subsections of this section of requirements described subordinate activiities, there is no requirement to distinctly perform the subordinate activities. Verification shall be by demonstration that the application is able to correctly perform path processing under several cases. The Government will provide several paths for purposes of verifying the path processing capability. The cases may involve both valid and invalid paths. Reasons that the paths are invalid may include expired and revoked certificates, invalid signatures, broken chains, and improper use of extensions. The Government may provide certificates that do not satisfy the profile for DOD certificates for purposes of verifying these requirements.
4.4	Application Configuration. The application shall demonstrate its capability to be configured for use with the DOD PKI. Tests of this requirement may also involve inspections of user and administrator manuals.
4.5	Application Documentation. These requirements shall be verified by inspection of the manuals and by a demonstration that the application performs as documented when the configuration guidance is followed.

5.3 Security Verification

Verification of an application's security has several variables. The variables include the functionality that the application provides and the application's design architecture and implementation. Because of this variability, this section does not provide detail verification requirements similar to the previous section. Use of [FIPS 140] evaluated cryptographic modules will simplify an applications security verification requirements and activities.

APPENDIX A: X.509 CERTIFICATES

This appendix is for information purposes only and provides additional details regarding the content, format, and representation of a certificate as defined by the X.509 Standard. A certificate is a variable length, hierarchical data structure. The certificate components may be variable length, primitive (direct) data values or may be recursive, indirect hierarchical structures. The following sections will describe the contents of a certificate, the various versions of certificates, and the format of certificate extensions.

Certificate Components

A signed certificate has a hierarchical structure that at the top level consists of a sequence of three elements. Table 8 lists the three elements and summarizes their contents, while Figure 1 illustrates the certificate structure and components.

Table 8 Signed Certificate Components

Field	Purpose and Content
To-be-signed certificate	A variable length, hierarchical data structure. The structure is further elaborated in Table 9.
Signature algorithm identifier	A variable length, hierarchical data structure describing the algorithm that the CA who created the certificate used to sign the certificate. This field actually duplicates a field in the To-be-signed certificate.[19] Signature algorithm identifiers include an object identifier for the algorithm and algorithm parameters. Some algorithms do not have parameters.[20]
Signature	The encrypted value resulting from computing a hash of the to-be-signed field. This is a primitive or direct value.

Figure 1 Signed Certificate and its Components

The first component of the signed certificate is the To-Be-Signed Certificate (TBSC) and is the component that has all of the certificate information. The TBSC itself is variable length, hierarchical structure. Table 9 lists the TBSC components and describes the content of each component. Figure 1 also illustrates the TBSC components.

---

19 This field allows a certificate user to verify the signature on the To-be-signed certificate without looking at the contents of the certificate. If the field is used as such, part of the subsequent certificate signature verification process must ensure that this algorithm identifier is consistent with (if not the same as) the To-be-signed Certificate's Signature algorithm identifier field.

20 The parameters field should be empty and, if present, should not be used (or otherwise relied upon).

Table 9 To-Be-Signed Certificate Components

Field	Purpose and Content
Version[21]	A primitive value that identifies which version of the X.509 standard applies to this certificate. Tells the certificate user which fields to expect in the certificate. Thus far, three versions are defined.[22]
Serial Number	The entity that created the certificate, the issuing CA, is responsible for assigning a serial number to distinguish a certificate from others it issues. Serial numbers are primitive and must be unique for the CA. This information is used in numerous ways. For example, when a certificate is revoked, its serial number is placed in a Certificate Revocation List (CRL).
Signature Algorithm Identifier	A variable length, hierarchical structure that identifies the algorithm used by the CA to sign the certificate. Should be the same as the Algorithm Identifier of the Signed Certificate. For signature verification purposes of the certificate, the parameters field should be null and should not be relied upon in any case.
Issuer Name	The X.500 name of the entity that signed the certificate. This field is also a variable, length hierarchical structure. This is normally a CA. Using this certificate implies trusting the entity that signed this certificate. (Note that in some cases, such as root or top-level CA certificates, the issuer signs its own certificate.)
Validity Period	Each certificate is valid only for a limited amount of time. This period is described by a start (not before) date and time and an end (not after) date and time (after which the certificate expires), and can be as short as a few seconds or almost as long as a century. The validity period chosen depends on a number of factors, such as the strength of the issuer's private key used to sign the certificate, the strength of the subject's private key associated with the subject public key, or considerations related to the CRL.[23] This is the expected period that entities can rely on the public value, if the associated private key has not been compromised. The validity period usually begins with the certificate's creation.

[21] Added with Version 2 of the certificate.

[22] Unfortunately, the values in this field start at 0; thus, a value of 2 in this field represents Version 3 of the X.509 standard.

[23] Longer validity periods increase the likelihood of revoking a certificate and the period that a certificate will have to be included in a CRL. Consequently, longer validity periods lead to longer CRLs.

Field	Purpose and Content
Subject Name	The name of the entity whose public key the certificate identifies. This name uses the X.500 standard, so it is intended to be unique. This is the Distinguished Name (DN) of the entity. For example: CN=Smith.John.J.1234567890, OU=DISA, OU=DOD, OU=PKI, O=U.S. Government, C=US
Subject Public Key Information	This is the public key of the entity being named, together with an algorithm identifier that specifies which public key cryptography system this key belongs to and any associated key parameters.
Subject Unique Name[21]	An optional field that contains a unique identifier that would allow the reuse of subject names. There is general consensus discouraging use of this field.
Issuer Unique Name[21]	An optional field that contains a unique identifier that would allow the reuse of issuer names. There is currently general consensus discouraging use of this field.
Extensions[24]	An optional collection of extensions. Each extension may provide additional information to certificate users. Extensions may be standard or local (i.e., extension standardization may occur through different levels of standards bodies and organizations). The extension format provides for three components: An identifier, criticality indicator, and content for the extension. See Table 11.

X.509 Certificate Versions

Like many standards, X.509 has evolved and is now in its third version. Table 10 identifies the versions and provides highlights of the characteristics of each version.

[24] Added with Version 3 of the certificate.

Table 10 X.509 Certificate Versions

Version	Highlights
1	Available since 1988, is widely deployed, and is the most generic.
2	Introduced the concept of subject and issuer unique identifiers to handle the possibility of reuse of subject and/or issuer names over time. Most certificate profile documents strongly recommend that names not be reused, and that certificates should not make use of unique identifiers. Consequently, Version 2 certificates are not widely used.
3	Most recent version. Adds certificate extensions. Defines an extension format and a set of standard extensions. CAs can define and incorporate additional extensions. Standard extensions include: Key Usage (limits the use of the keys to particular purposes such as "signing-only") and Subject Alternative Names (allows other identities to also be associated with this public key such as DNS names, E-mail addresses, and IP addresses).

Certificate Extensions

Version 3 of X.509 added extensions to certificates. Extensions provide additional information to certificate users. The standard provided a scheme for adding extensions and defined several specific Standard Extensions. Other standards bodies and PKI operators may define new extensions. Extensions are identified with Object Identifiers (OIDs) . Table 11 identifies extension components and their content. Figure 1 also includes extensions.

Table 11 Certificate Extension Components

Extension Component	Purpose
Identifier	An object identifier (OID) that provides a reference to identify the organization responsible for defining the extension and the rules for encoding and interpreting the extension's content component.
Criticality	A Boolean (true/false) value indicating whether the certificate user (relying party) must understand and use or enforce the extension. If the user does not understand a critical extension, the user should reject use of the certificate.
Content	The data describing the meaning of the extension. The format, values, and interpretation of the content are dependent on the individual extension. For example, for the key usage extension the content describes how the CA intended that the key be used (e.g., digital signature, and key encipherment) and, for the subject alternative name extension, the content might include the certificate subject's e-mail address.

APPENDIX B: CERTIFICATION AUTHORITIES

This appendix is for information purposes and describes Certification Authorities with emphasis on the CA hierarchies and their operation. The information in this appendix supplements the information provided in Section 2.0.

CA Hierarcies

A CA may delegate responsibilities to issue certificates to other subordinate CAs. Thus, CAs may form hierarchies or trees. The leaf nodes[25] are end-entities. Subscriber certificates are end-entities. Non-leaf nodes belong to CAs. The CA that has no ancestors is the Root CA.

CAs have certificates and may digitally sign certificates of others. The CA certificate contains the public key associated with the private key that the CA uses to sign certificates. The CA's certificate is important to relying parties. Relying parties use the CA's certificate to obtain the CA's public key and use it to verify certificates that the CA has issued. If the relying party trusts the CA, the relying party believes that the certificates' binding of public keys to their owners is accurate and trustworthy. The Root CA's certificate is self-signed; that is, the Root CA signs its own certificate.

The DOD Class 3 PKI uses a three level hierarchy.[26] The Root CA comprises the top or first level; a set of signing CAs makes up the second level, and end-entities constitute the third level. Issues such as span of control, workload distribution, and geographic separation determine the number of signing CAs that exist. Figure 2 illustrates the hierarchy.

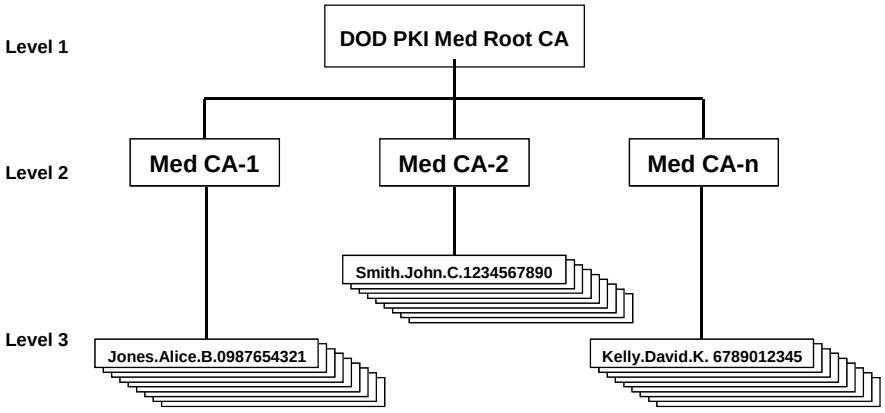

Figure 2. DOD Class 3 PKI Certification Authority Hierarchy

[25] Leaf nodes are nodes in a tree that have ancestors but no descendants.

[26] The hierarchy could change and applications should not assume that the hierarchy will always have three levels.

Multiple PKIs within DOD Class 3 PKI

The DOD Class 3 PKI actually consists of multiple independent hierarchies. There are separate hierarchies for the unclassified network, the NIPRNET, and the classified network, SIPRNET. Figure 2 shows the hierarchy for the unclassified network. The classified network PKI is essentially identical but there is slight difference in the names of the CAs. The DOD has also sanctioned commercial CA services providers to serve as External Certification Authorities (ECAs).[27] Each ECA provides both a separate Root CA and a separate PKI hierarchy. Additional hierarchies may exist during periods of evolutionary transition between major versions of the DOD Class 3 PKI.

[27] Actually the current ECAs are Interim ECAs (IECAs) because the full details of agreements between the Government and the ECAs are not finalized.

APPENDIX C: DIRECTORY ORGANIZATION AND ACCESS

This appendix is for information purposes and briefly describes organization and contents of DOD PKI Class 3 directory. The appendix describes the overall directory structure, the composition of directory entry names, and the major types and contents of directory entries. Refer to the *Department of Defense Class 3 Public Key Infrastructure Interface Specification* [IF] for a complete description of the DOD Class 3 directory.

Directory Organization

The directory is a Lightweight Directory Access Protocol (LDAP) directory. LDAP is based on the overall X.500 directory architecture. The directory has a hierarchical organization. The DOD PKI Class 3 directory has a relatively shallow hierarchy. The hierarchy has branches for the major DOD organizations, the Commanders-in-Chief (CINCs), services, and agencies. Individuals are immediately under their major organization. Because entries must have unique names, individuals are assigned a unique number that augments their name and provides uniqueness.

Entry Names

Each entry has a unique name, the Distinguished Name (DN). The DN has multiple components. Each component is an ***attribute-value pair***. The pair consists of an attribute name and it value. Commas (,) separate pairs. The format for DNs is *little endian*, the least significant information appears first. An example of a DOD Class 3 DN is:

```
cn=AMES.ALICE.A.0506000009, ou=DISA, ou=PKI, ou=DoD, o=U.S. Government, c=US
```

Table 12 lists the typical attributes that appear in DNs from the most significant to least significant. DNs do not need to have all components. DNs may include multiple pairs involving the same attribute (e.g., the multiple organizational units in the above example).

Table 12 Typical Common Name Components

Attribute	Designator
Country	C
State	ST
Locality	L
Organization	O
Organizational Unit	OU
Common Name	CN
E-mail Address	E

Each node in the directory has an additional component in its DN relative to its ancestor node. The parent of the node with the above DN has the DN:

```
ou=DISA, ou=PKI, ou=DoD, o=U.S. Government, c=US
```

Objects, Attributes, and Values

The directory structure is based on object classes. The directory maintains a list of defined object classes. Object classes have a hierarchical organization. An object class can extend its parent class. An object class definition includes the parent object and the attributes that members of the object class may have. The attributes can be designated as required and optional. Objects must have values assigned to the required attributes. Values are optional for the optional attributes.

Entries in the directory are instances of objects. An entry can be an instance of multiple objects. The entry consists of attributes and related values. Most attributes can be multi-valued; the attribute may have more than one value. For example, the e-mail address attribute for an individual with more than one e-mail address would have a value for each address.

The primary entries of interest in the DOD Class 3 PKI are the entries for individuals and CAs. Table 13 and Table 14 illustrate entries for an individual and a CA respectively. Entries for individuals and CAs include a certificate attribute whose associated value is the certificate. CA entries also include a attribute that holds the CRL. Refer to the *Department of Defense Class 3 Public Key Infrastructure Interface Specification* [IF] for a complete description of the objects and related attributes used and a description of the directory hierarchy.

Table 13 Directory Entry Attributes and Values for People

Attribute	Value
DN	cn=AMES.ALICE.A.0506000009, ou=DISA, ou=PKI, ou=DoD, o=U.S. Government, c=US
Object Class	top, person, organizationalPerson, inetOrgPerson, diiPerson
Name	AMES.ALICE.A.0506000009
Last Name	AMES
First Name	ALICE
Email	ames1a@ncr.disa.mil
Organizational Unit	DISA
City	FALLS CHURCH
st	VA
uid	0506000009
lradn	cn=fletcher.james.c.0506000000, OU=RA, OU=PKI, OU=DoD, O=u
creatorsname	cn=Directory Manager, ou=PKI, ou=DoD, o=U.S. Government, c=US
createtimestamp	199808191723O2Z
modifiersname	cn=Directory Manager, ou=PKI, ou=DoD, o=U.S. Government, c=US
modifytimestamp	199808191736S2Z
User Certificate:	This Certificate belongs to: cn=AMES.ALICE.A.0506000009 ou=DISA ou=PKI ou=DoD o=U.S. Government c=US This Certificate was issued by: cn=Med CA-1 ou=PKI ou=DoD o=U.S. Government c=US Serial Number: 00:9F This Certificate is valid from Wed Aug 19, 1998 to Sun Aug 19, 2001 Certificate Fingerprint: BB:BC:C1:F6:D7:87:7E:1B:5E:D0:0C:01:03:E8:CD:5E

Table 14 Directory Entry Attributes and Values for CAs

Attribute	Value
DN	cn= Med CA-1, ou=PKI, ou=DoD, o=U.S. Government, c=US
Object Class	top, person, organizationalPerson, inetOrgPerson, diiPerson, certificationAuthority
Name	Med CA-1
Last Name	Med CA-1
uid	Med CA-1
creatorsname	cn=Directory Manager,ou=PKI,ou=DoD,o=U.S. Government,c=US
createtimestamp	19980810170237Z
cacertificate;binary	<Binary Value>[28]
certificaterevocationlist;binary	<Binary Value>[28]
modifiersname	cn=Directory Manager,ou=PKI,ou=DoD,o=U.S. Government,c=US
modifytimestamp	20000503124805Z

[28] The binary value is the DER encoding of the respective certificate or CRL.

APPENDIX D: CERTIFICATE CHAIN PROCESSING

This appendix describes Certificate chain process and supplements the requirements provided in Section 4.3.3. The appendix describes a path processing algorithm and provides additional details on processing certificate extensions.

Certification Path Processing and Validation

Certification path processing procedures for the DOD PKI are based on Section 6 of [RFC 2459]. Certification path processing verifies the binding between the subject's distinguished name and the subject's public key. Path processing requires obtaining a sequence of certificates that support that binding. The binding is limited by constraints that are specified in the certificates that comprise the path. The basic constraints and policy constraints extensions allow the certification path processing logic to automate the decision making process.

This section describes an algorithm for validating certification paths. Any algorithm may be used by a particular application. However, the results shall be equivalent to those that would result from the algorithm provided below. The description assumes that all valid paths begin with certificates issued by the DOD PKI Root CA. The algorithm has been tailored to the DOD PKI and provides the minimum requirements for path processing. The algorithm assumes that it is operating on a path of DOD PKI certificates. Processing related to considerations such as policy mapping and name constraints has been removed from the algorithm because the PKI currently uses neither policy mapping nor name constraints. Applications needing to interoperate with other PKIs may need to include these considerations in their path-processing algorithm.

Basic Path Validation

The validation algorithm assumes that the trusted public key (and related information) comes from the DOD PKI's root certificate. This simplifies the description of the path processing procedure. Note that the signature on the root certificate does not provide any security services. The trusted public key (and related information) may be obtained in other formats; the information is trusted because of other procedures used to obtain and protect it. The goal of path validation is to verify the binding between a subject's distinguished name and the subject public key, as represented in the "end entity" certificate, based on the public key of the DOD Root CA.

For purposes of the discussion of the algorithm, a certification path is a sequence of n certificates[29] where:

- For all x in {1, (n-1)}, the subject of certificate x is the issuer of certificate x+1.
- Certificate x=1 is the Root (self-signed) certificate, and
- Certificate x=n is the end entity certificate.

This section assumes the parameters shown in Table 15 are inputs to the path processing algorithm.

Table 15 Path Processing Algorithm Parameters

Parameter	Description
path	A certification path of length n.
Initial_policy	Either a set of identifiers (each comprising a sequence of policy element identifiers), which identifies one or more certificate policies, any one of which would be acceptable for the purposes of certification path processing, or the special value "any-policy".
C	The current date/time (if not available internally to the certification path processing module).
T	The time for which the validity of the path should be determined. (This may be the current date/time, or some point in the past.)

From the inputs, the procedure initializes two state variables shown in Table 16.

[29] For the DOD PKI n is normally 3 for a complete path. The depth of the hierarchy could change in the future.

Table 16 Path Processing Algorithm State Variables

Variable and Type	Initial Value	Description
Acceptable_policy (set)	the special value any-policy.	A set of certificate policy identifiers comprising the policy or policies recognized by the public key user together with policies deemed equivalent through policy mapping.
Explicit_policy (Integer)	n+1	An integer which indicates if an explicit policy identifier is required. The integer indicates the first certificate in the path where this requirement is imposed. Once set, this variable may be decreased, but may not be increased. (That is, if a certificate in the path requires explicit policy identifiers, a later certificate cannot remove this requirement.)

The actions performed by the path processing software for each certificate i=1 through n are described below. The self-signed certificate is certificate i=1, the end entity certificate is i=n. The processing is performed sequentially, so that processing certificate i affects the state variables for processing certificate (i+1).

The path processing actions to be performed are:

Step 1: Set i=1.

Step **2**: Verify the basic certificate information for path(I), including that:

1) the certificate was signed using the subject public key from certificate i-1 (in the special case i=1, this step may be omitted; if not, use the subject public key from the same certificate),
2) the certificate validity period includes time T,
3) the certificate had not been revoked at time T based on status information current as of time C, and
4) the subject and issuer names chain correctly (that is, the issuer of this certificate was the subject of the previous certificate.)

Step **3**: If i<n, Verify that the certificate is a CA certificate using the basicConstraints extension.

Step **4**: If there's a critical keyUsage extension,[30] then:

1) if i<n ensure keyCertSign is set.

[30] The keyUsage extension is critical in DOD PKI certificates.

2) Otherwise ensure the keyUsage setting is consistent with the intended use. Key usage is consistent if the application is using the key for:
   (a) Authentication and the digitalSigniture bit is set.
   (b) Technical non-repudiation and the nonrepudiation bit is set.
   (c) Encryption and the keyEncipherment bit is set.

Step **5**: Verify that policy information is consistent with the initial_policy set: if the explicit_policy state variable is less than or equal to i, a policy identifier in the certificate shall be in the initial_policy set;

Step **6**: Verify that policy information is consistent with the acceptable policy set: if the certificatePolicies extension is marked critical, then:

1) the intersection of the policies extension and the acceptable_policy set shall be non-null;
2) the acceptable_policy set is assigned the resulting intersection as its new value.

Step **7**: Verify that the intersection of the acceptable_policy set and the initial_policy set is non-null.

Step **8**: If a policyConstraints extension is included and marked critical in the certificate, then if requireExplicitPolicy is present and has value r, the explicit policy state variable is set to the minimum of its current value and the sum of r and i (the current certificate in the sequence).

Step **9**: Increment i by 1 and if i is less than or equal to n go back to Step **2**.

Step **10**: The path process is now complete and was successful.

If any one of the above checks fail, the procedure terminates, returning a failure indication and an appropriate reason. If none of the above checks fail on the end-entity certificate, the procedure terminates, returning a success indication together with the set of all policy qualifier values encountered in the set of certificates.

The procedure described above is basic. The algorithm will validate a path that does not contain any critical certificatePolicies extensions regardless of the setting of the initial_policy parameter. A DOD PKI path would validate against any initial policy setting because the PKI does not mark the certificatePolicies extension critical. Several modifications are possible.

- If the relying party wants to ensure that the path validates only with a path explicitly containing a Class 3 policy identifier, modify Step **6** to process non-critical certificate policy extensions as if they were critical.
- If the relying party wants to ensure that all certificates except the root, follow the DOD PKI's *weak* requirement that the intermediate CAs' and their descendants' certificates have DOD PKI policies, modify Step **8** to use a non-critical policyConstraints extension as if it was critical.

- If the relying party wants to require that all certificates except the root have DOD PKI policies, initialize the explicit_policy set variable to 2.

With the current DOD certificate profiles, the basic algorithm and the suggested alternatives all perform identically.

Use of Expired Certificates

Application processing on behalf of relying parties should exercise caution in the use of the value for T.[31] Use of the value C is recommended. That is relying parties should process the path as if the subscriber just signed the document. If T and C are different, and a certificate expired between T and C, a CRL issued after the certificate expired will not include the certificate since CRLs do not include expired certificates. Applications that must allow T to be less than C must recognize situations where involved certificates have expired. In such cases the application should fail the verification unless the application has access to a CRL issued after T and before the certificate expired.[32]

Extension Processing

Applications must be able to process the two extensions that the DOD PKI marks critical. These extensions are key usage and basic constraints. While the X.509 view of non-critical extensions is that clients do not need to be able to process non-critical extensions, the handling of these extensions is application dependent. For some applications non-critical extensions may be as important and necessary as the critical applications. The remainder of this section describes the extensions that the DOD PKI uses and how applications should use the extensions:

- **Key Usage.** This critical extension indicates the purpose for which the CA intended the key to be used. Applications must ensure their use of keys is consistent with the key usage indicated in the associated certificate. Uses of both private and public keys should be consistent with the key usage in the certificate containing the key. Clients supporting the relying party have responsibility to follow the key usage. Relying parties should ensure that the certificates containing keys used to verify CA signatures on certificates and CRLs have key usage for CA and CRL signing respectively.

---

[31] This discussion primarily applies to signature applications and assumes that T is less than or equal to C. For encryption uses, C is equal to T, while for signature applications T is the time of signature and T is less than C.

[32] Ideally the CRL should be the last CRL previous to the certificate expiration. Also, the application should fail the verification if the difference between T and the certificate expiration is less than the window the CA has for publishing a CRL after receiving notification to revoke a certificate.

- **Extended Key Usage (EKU).** This extension generally is used to indicate that the certificate's use is restricted to a particular application or function. Certificates issued to OSC responders will contain this extension and will have a value indicating that the certificate was issued for use in signing OSC responses. The processing of paths involving an OSC response must ensure this EKU extension indicating use for OSC responses is in the certificate belonging to the signer of the OSC response. The response must be rejected otherwise.
- **Basic Constraints.** The basic constraints extension indicates whether the certificate holder is a CA or an end-entity. If the holder is a CA, then the extension may constrain the depth of the hierarchy below the CA. The DOD PKI uses the extension to signify whether the holder is a CA or end-entity only and does not plan to use the path length constraints. Applications should ensure that paths comply with the extension value. Specifically, a certificate with basic constraints indicating an end-entity must not be used to sign a certificate.
- **Authority Key Identifier.** This extension provides an identifier for the public key needed to verify the certificate. CAs over their lifetime may have multiple keys. The purpose of this extension is to help a relying party find the relevant CA certificate in such circumstances when the CA has multiple active certificates. The key identifier in the DOD PKI is simply a SHA hash of the public key field in the relevant CA certificate.
- **Subject Key Identifier.** The purpose of this extension is to distinguish the certificate from other certificates belonging to the same subscriber. A subscriber may have certificates intended for different uses (e.g., identity, e-mail signature, e-mail encryption). If the relying party has to deal with expired certificates from an archive, the key identifier can distinguish the relevant certificate from several expired certificates. The identifier is simply a SHA hash of the subject public key field of the certificate. The issuer name, the subject name, and key identifier combination should uniquely identify a key.[33]
- **Certificate Policies.** The certificate policies extension has the OIDs of the policies that the certificate supports or asserts. For the DOD PKI certificates assert either the Medium-Pilot or Class 3 policies. Applications that require Class 3 certificates should use this extension to identify certificates as such. Currently multiple, separate PKIs (e.g., the NIPRnet and SIPRnet PKIs) issue Class 3 certificates. In the future certificates from more than one class may be under a common root CA.

[33] Key identifiers are not unique in a PKI that allows users to renew a certificate. A renewed certificate uses the same key as the original certificate.

- **Policy Constraints.** This extension constrains the policies that a CA can assert. Normally, a higher level CA imposes these constraints on a lower level CA. This extension only appears in the intermediate CAs in the DOD PKI. Its contents mean that all certificates in the DOD PKI except the root must assert at least one of the Medium-Pilot or Class 3 policies. Applications should process this extension and should process the extension when marked non-critical as if it were marked critical. Relying parties should ensure that end-entity certificates under the CA contain the policy.
- **Subject Alternate Name.** This extension contains alternate name forms for certificate subject. Allowable forms include URIs, e-mail addresses, and directory names. The DOD PKI will put e-mail addresses in this extension. Applications may assume that the e-mail address belongs to the subscriber. Applications that integrate with e-mail may want to consider enforcing the association between the e-mail address and the subscriber. The extension may have multiple names to include multiple names of the same form (e.g., the extension could include a personal URI, a directory URI, a directory name, and several e-mail addresses).
- **Issuer Alternate Name.** This extension contains alternate name forms for a certificate's issuer. Allowable forms are the same as those for the subject alternate name. The DOD PKI will put an LDAP URI for the issuer's entry in the DOD PKI directory. Relying party applications should use the URI to locate the issuer's certificate for path development if it is needed and not otherwise available. Subsequent directory links may help obtain other parts the certificate path.
- **CRL Distribution Points.** This extension provides information on how or where to obtain a CRL that could include the certificate. The extension allows use of multiple name forms for the distribution point. The DOD PKI provides an LDAP URI for the CRL in some cases. Applications operating with network access should retrieve needed CRLs using the contents of this extension when it is available.
- **Authority Information Access.** This extension provides a URI for an OSC responder that can validate the certificate. Applications with network access should use the URI from this extension to obtain the status of the certificate.

References

{ABA} American Bar Association. Digital Signature Guidelines. August 1, 1996.

{FKK} Freier, A. O., P Karlton, and P. C. Kocher. The SSL Protocol Version 3.0. Transport Layer Security Working Group INTERNET-DRAFT, November 18, 1996

{KEA} Department of Defense. *SKIPJACK and KEA Algorithm Specifications*. Version 2.0, 29 May 1998.

{PKCS 12} RSA Laboratories. *Personal Information Exchange Syntax Standard*, PKCS #12, Version 1.0 DRAFT, 30 April 1997

{RFC2560} X.509 Internet Public Key Infrastructure Online Certificate Status Protocol - OSCP (RFC 2560)

{RSA} Rivest, R.L., A. Shamir, and L. Adleman. *A Method for Obtaining Digital Signatures and Public-Key Cryptosystems*. *Communications of the ACM,* 21(2):120–126, February 1978.

{Sch} Schneier, B. Applied Cryptography: Protocols, Algorithms, and Source Code in C, Second Edition. New York: John Wiley & Sons, 1996. ISBN: 0471117099

List of Acronyms

C3	DOD PKI Class 3 Assurance Level
C4	DOD PKI Class 4 Assurance Level
CA	Certification Authority
CDP	CRL Distribution Point
CINC	Commander-in-Chief
CRL	Certificate Revocation List
DES	Data Encryption Standard
DN	Distinguished Name
DOD	Department of Defense
DSA	Digital Signature Algorithm
ECA	External Certification Authority
EKU	Extended Key Usage
FBCA	Federal Bridge Certification Authority
FIPS	Federal Information Processing Standard
FPKI	Federal Public Key Infrastructure
HAG	High Assurance Guard
HTTP	Hypertext Transmission Protocol
HTTPS	Hypertext Transmission Protocol over SSL
IECA	Interim External Certification Authority
JITC	Joint Interoperability Test Command
KEA	Key Exchange Algorithm
KRM	Key Recovery Manager
LDAP	Lightweight Directory Access Protocol
NIPRNET	Unclassified Internet Protocol Router Network
OID	Object Identifier
OSC	Online Status Check
OSCR	Online Status Check Responder
PK	Public Key
PKCS	Public Key Cryptography Standard
PKI	Public Key Infrastructure
RNG	Random Number Generator
RSA	Rivest, Shamir, and Adleman
SHA	Secure Hash Algorithm 1
SHA-1	Secure Hash Algorithm 1
SIPRNET	Secret Internet Protocol Router Network
TBSC	To-Be-Signed-Certificate
TDEA	Triple Data Encryption Algorithm
TTP	Trusted Third Party
URI	Uniform Resource Indicator