National Information Assurance Partnership

Common Criteria Evaluation and Validation Scheme Validation Report

Public Key Infrastructure Framework

Version 2.1

Report Number: **CCEVS-VR-VID10235-2008**

Dated: 2008-01-8

Version: 1.0

National Institute of Standards and Technology
Information Technology Laboratory
100 Bureau Drive
Gaithersburg, MD 20899

National Security Agency
Information Assurance Directorate
9800 Savage Road STE 6740
Fort George G. Meade, MD 20755-6740

ACKNOWLEDGEMENTS

Validation Team

Dr. Patrick Mallett, The MITRE Corporation
Dr. Jerome Myers, The Aerospace Corporation

CCTL

atsec information security corporation

Austin, TX

Table of Contents

1. EXECUTIVE SUMMARY

This document is intended to assist the end-user of this product with determining the suitability of the product in their environment. End-users should review both the Security Target (ST), which is where specific security claims are made, and this Validation Report (VR), which describes how those security claims were evaluated.

This Validation Report documents the NIAP validators' assessment of the evaluation of the Public Key Infrastructure Framework Version 2.1. It presents the evaluation results, their justifications, and the conformance results. This validation report is not an endorsement of the IT product by any agency of the U.S. Government and no warranty of the IT product is either expressed or implied.

The evaluation was performed by atsec information security corporation in the United States, and was completed in October 2007. atsec information security corporation is an approved NIAP Common Criteria Testing Laboratory (CCTL). The evaluation was conducted in accordance with the requirements of the Common Criteria for Information Technology Security Evaluation, version 2.3. The information in this report is largely derived from the Evaluation Technical Report (ETR) and associated test report, both written by the CCTL. The evaluation determined the product to be **Part 2 extended, Part 3 conformant**, and to meet the requirements of **Evaluation Assurance Level 4 (EAL4) augmented by ALC_FLR.2**. Additionally, the TOE was shown to satisfy the requirements of the U.S. Government Family of Protection Profiles Public Key-Enabled Applications for Basic Robustness Environments, Version 2.77, February 1, 2007, with the following packages:

- Certification Path Validation (CPV) – Basic Package
- CPV – Basic Policy Package
- CPV – Policy Mapping Package
- CPV – Name Constraints Package
- PKI Signature Generation Package
- PKI Signature Verification Package
- PKI Encryption using Key Transfer Algorithms Package
- PKI Decryption using Key Transfer Algorithms Package
- Online Certificate Status Protocol (OCSP) Client Package
- Certificate Revocation List (CRL) Validation Package

Public Key Infrastructure Framework Version 2.1 (PKIFv2) is a toolkit used by application developers to incorporate secure PKI functionality into an application. It provides a set of C++ programming interfaces comprising of extensible classes that performs PKI-related functions. PKIFv2 is a software only component; the platform/hardware is part of the TOE environment.

The main functions of PKIFv2 are to provide certification path processing, digital signature generation and verification, and public key encryption and decryption. PKIFv2 does not implement any cryptographic functions; instead, it provides an interface to the underlying Cryptographic Service Provider (CSP) that implements the actual cryptographic functions. The CSPs used by PKIFv2 are Microsoft CAPI on Windows and Network Security Services (NSS) on Linux, both of which have been validated according to FIPS 140-2 under the Cryptographic Module Validation Program.

The evaluation covers the operating systems and associated platforms running evaluated configurations of the TOE that are defined in the Security Target.

The validation team monitored the activities of the evaluation team, provided guidance on technical issues and evaluation processes, reviewed successive versions of the Security Target, reviewed selected evaluation evidence, reviewed test plans, reviewed intermediate evaluation results (i.e., the CEM work units), and reviewed successive versions of the evaluation technical report (ETR) and test report. The validation team determined that the evaluation team showed that the product satisfies all of the functional requirements and assurance requirements defined in the Security Target (ST) for an EAL4 evaluation. Therefore, the validation team concludes that the CCTL findings are accurate, and the conclusions justified.

2. IDENTIFICATION

The Common Criteria Evaluation and Validation Scheme (CCEVS) is a National Security Agency (NSA) effort to establish commercial facilities to perform trusted product evaluations. Under this program, security evaluations are conducted by commercial testing laboratories called Common Criteria Testing Laboratories (CCTLs) using the Common Evaluation Methodology (CEM) for Evaluation Assurance Level (EAL) 1 through EAL 4 in accordance with National Voluntary Laboratory Assessment Program (NVLAP) accreditation granted by the National Institute of Standards and Technology (NIST).

The NIAP Validation Body assigns Validators to monitor the CCTLs to ensure quality and consistency across evaluations. Developers of information technology products desiring a security evaluation contract with a CCTL and pay a fee for their product's evaluation. Upon successful completion of the evaluation, the product is added to NIAP's Validated Products List.

Table 1 provides information needed to completely identify the product, including:

- The Target of Evaluation (TOE): the fully qualified identifier of the product as evaluated
- The Security Target (ST), describing the security features, claims, and assurances of the product
- The conformance result of the evaluation, the Evaluation Technical Report or ETR
- The Protection Profile to which the product is conformant
- The organizations and individuals participating in the evaluation

Table 1: Evaluation Identifiers

Item	Identifier
Evaluation Scheme	United States NIAP Common Criteria Evaluation and Validation Scheme
Target of Evaluation	Public Key Infrastructure Framework Version 2.1
Protection Profile	U.S. Government Family of Protection Profiles Public Key-Enabled Applications for Basic Robustness Environments, Version 2.77.
Security Target	*Public Key Infrastructure Framework Version 2.1;* Version 1.8 8 January 2008
Evaluation Technical Report	*Evaluation Technical Report a Target of Evaluation: Public Key Infrastructure Framework Version 2.1;* Version 2.0, 16 November 2007
Conformance Result	CC V2.3, Part 2 extended, Part 3 conformant, EAL 4 augmented by ALC_FLR.2, and PKE PP V2.77-compliant
Sponsor	US Marine Corps
Developer	CygnaCom Solutions, Inc.
Evaluators	atsec information security corporation
Validators	Dr. Patrick Mallett, The MITRE Corporation Dr. Jerome Myers, The Aerospace Corporation

3. SECURITY POLICY

3.1. Certification Path Processing

PKIFv2 performs X.509 certification path processing including certification path development, certification path validation, and revocation status determination.

PKIFv2 supports X.509 version 2 certificates and X.509 CRLs, versions 1 and 2.

3.1.1. Certification Path Development

PKIFv2 performs certification path validation beginning with a target certificate and building towards a trusted certificate.

There are three types of public key certificates involved in certificate path validation:

- Trust anchor (TA) certificates: These are certificates containing public keys that do not require any validation. Trust anchors generally take the form of a self-signed certificate. TAs must be delivered to entities that rely on the TA's public key using trusted means. The primary purpose of the trust anchor is to provide a means of conveying a Distinguished Name (DN), public key, algorithm identifier, and the public key parameters (if applicable) for use in validating certification paths.

- Intermediate certificates: These are the certificates issued to CAs. All certificates in a certification path are intermediate certificates, except the trust anchor certificate and end entity certificate.

- End certificates: This is the last certificate in the certification path and is issued to the subscriber of interest. This is an end-entity certificate (i.e., a certificate issued to an entity not functioning as a CA).

PKIFv2 processes security-related certificate extensions including ocsp-nocheck, keyUsage, extendedKeyUsage, and basicConstraints; certificate policy-related extensions including certificatePolicies, policyMapping, inhibitAnyPolicy, policyConstraints, and nameConstraints.

PKIFv2 enforces the following matching rules during path development:

- Name chaining - Issuer field in a certificate must mach Subject field in the parent certificate

- Algorithm chaining – signature algorithm must match the SubjectPublicKeyAlgorithm in the parent certificate

- Validity period checking – the time of interest must be within the validity period

Additionally, certificates in which the Authority Key Identifier (AKID) matches the Subject Key Identifier (SKID) in the parent certificate are given priority. Alternatively, PKIFv2 permits applications to require PKIFv2 to perform basic certification path validation processing during path development.

3.1.2. Certification Path Validation

PKIFv2 implements a certification path validation algorithm with all the requirements of X.509/RFC3280 in the areas of name chaining, signature chaining, certificate policy processing, and name constraints and other constraints processing.

PKIFv2 uses the following input parameters provided from the application to perform path validation:

1. Acceptable certificate policies or if a common certificate policy should be required for the certification path

2. An initial inhibitPolicyMappings indicator

3. An initial inhibitAnyPolicy indicator

4. An initial requireExplicitPolicy indicator

PKIFv2 does not process the entire certification path; as soon as a path validation error is detected, PKIFv2 notifies the calling application of the failure and provides an error indication. Alternatively, applications can provide an error handler to overwrite specific errors.

By default, PKIFv2 assumes that the path validation is being done as of the current system time, as opposed to verification of signature relative to a point in time in the past. However, applications can specify a time other than the current time for use during path validation.

3.1.3. Revocation Status Checking

PKIFv2 supports the use of both CRL and OCSP for checking the revocation status of public key certificates.

3.1.3.1. CRL Processing

PKIFv2 processes CRLs to determine the revocation status of a certificate in accordance with RFC3280 including execution of certification path development and validation operations to support verification of the signature on the CRL, if necessary.

PKIFv2 processes CRLs obtained from following sources:

- Locations indicated by a CRL Distribution extension in a certificate
- Local storage facilities
- LDAP-accessible directories

PKIFv2 permits but does not enforce the same public key used for both verifying a CRL signature and a certificate signature.

For CRLs, the current system time is used for revocation status checking and fresh revocation status is not enforced. However, PKIFv2 permits an application to bypass revocation, to specify the time of interest for path validation and revocation status information and to indicate if fresh revocation information must be used.

3.1.3.2. OCSP Processing

PKIFv2 can generate Online Certificate Status Protocol (OCSP) requests and validate OCSP responses to determine the revocation status of public key certificates in accordance with the IETF RFC 2560. PKIFv2 verifies OCSP Responder as a trust anchor or as an end entity authorized to sign OCSP responses. PKIFv2 establishes trust in the OCSP responder certificates by performing Certification Path Validation.

PKIFv2 supports the following modes of OCSP client configurations:

- Trusted responder. When this method is used, the TOE forwards OCSP requests for status of all certificates to an application-specified responder. A responder can be a trust anchor, CA, or CA delegate.
- AIA-specified OCSP responder. When this method is used, the TOE queries OCSP responders identified in AIA certificate extensions.

For OCSP responses, PKIFv2 provides two additional configurable parameters: a Boolean value and integer to permits applications to require the thisUpdate field in a response to be within a specified number of seconds from the current time; a Boolean value that indicates whether nonce values included in a request must be present in the response.

3.2. Digital Signature Generation

PKIFv2 uses a private key provided by the application using PKIFv2 to create a signature using the Cryptographic Message Syntax (CMS) SignedData format.

SignedData messages that are generated by PKIFv2 always identify signers via subject key identifier, not issuer name and serial number.

3.3. Digital Signature Verification

PKIFv2 provides functions to verify a signature and to establish trust in the signer’s public key. Trust in the signer’s public key is established via certification path development and path validation. Certification path development function is called iteratively until a valid path is found or no more certification paths exist.

PKIFv2 verifies that if the signer’s certificate contains a key usage extension, the digital signature or nonrepudiation bit is set.

PKIFv2 also verifies ContentType and MessageDigest attributes.

3.4. Public Key Encryption

PKIFv2 provides functions to encrypt the data and to establish trust in the recipient’s public key using key transfer algorithms such as RSA. Trust in the recipient’s public key is established via certification path development and path validation. Path certification development and validation is called iteratively until a valid path is found or no more certification paths exist. Encryption is performed using the public key and public key algorithm obtained from the recipient’s certificate.

PKIFv2 verifies that if the recipient’s certificate contains a key usage extension, the key encipherment bit is set.

3.5. Public Key Decryption

PKIFv2 provides functions to decrypt the data using one of the following methods:

- Obtains a private key provided by the user/application, or
- If no private key was presented, PKIFv2 automatically searches for the correct decryption key in the user’s keystore. Depending on the mediator collection defined, it searches either the Microsoft CAPI keystore (on Windows) or NSS keystore (on Linux).

4. ASSUMPTIONS

4.1. Usage Assumptions

As stated in the Security Target, the assumptions regarding the minimum physical and procedural measures to maintain security of the TOE are:

- The TOE environment is responsible for providing the TOE with appropriate physical security, commensurate with the value of the IT assets protected by the TOE.
- The TOE is assumed to be properly installed and configured.
- The attack potential for the TOE is assumed to be low.
- Those responsible for the administration of the TOE and the TOE environment are non-hostile individuals who are appropriately trained and follow all administrator guidance.
- Users of the TOE are assumed to be trusted and follow the guidance provided with the TOE with respect to the configuration of the TOE, the use of TOE functions and interfaces, and the protection of the TOE code and data.

4.2. Clarification of Scope

PKIFv2 is a set of functions packaged as a dynamic-link library or dynamically loaded shared library installed as a separate product that any authorized application program running on the platform may call to utilize PKI support. PKIFv2 relies on the IT environment to provide the essential security features.

PKIFv2 provides a collection of extensible C++ programming interfaces that can be used to add secure PKI support to software applications. It provides the capability to perform certification path validation, digital signature generation and verification, and public key encryption and decryption.

PKIFv2 also provides a collection of Java and .NET interfaces for an application to access the PKIFv2 functionality. These interfaces, which do not enforce any security, simply are wrapper interfaces that allow convenient access to the full PKIF library.

PKIFv2 ships only compiled object code and header files.

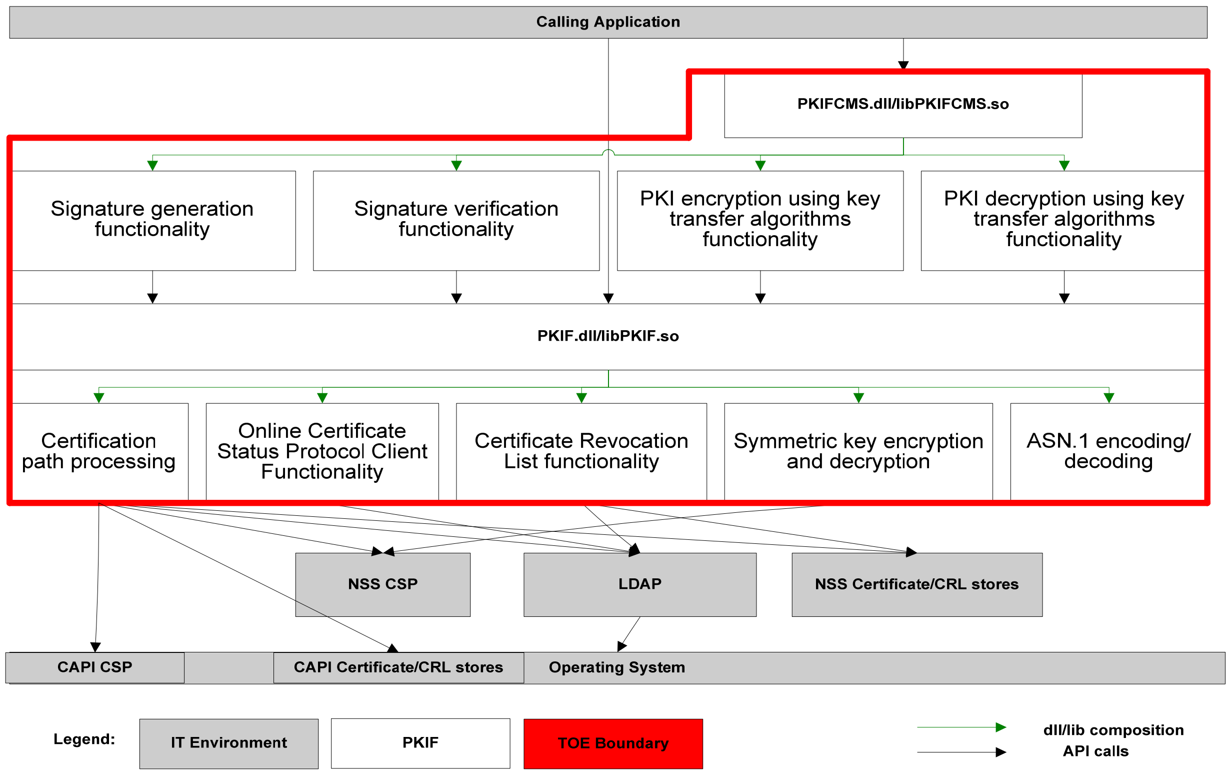
The TOE consists of:

- libPKIF.so, libPKIFCMS.so (Linux)
- PKIF.dll, PKIFCMS.dll (Windows)

The TOE does not implement any cryptographic operations; instead, it provides an interface to the Cryptographic Service Provider (CSP) to perform such operations. The CSP, however, is not part of the TOE, but is part of the TOE environment. The cryptographic capability of the TOE is determined by the implementation of the CSP.

The operating system is not part of the TOE. It provides identification and authorization for TOE users, and auditing capabilities. The operating system is also responsible for protecting access to TOE services and PKIF software, certificate stores, private keys, and audit logs. As a software library, the TOE itself has very limited ability to protect against the corruption of TSF; hence, the underlying IT environment responsible for preventing bypass of the TSF or compromise of the TOE and its data by restricting access to authorized users only. Furthermore, the TOE also relies on the operating system for protection against object reuse and domain separation.

5. ARCHITECTURAL INFORMATION


Calling Application
PKIFCMS.dll/libPKIFCMS.so
Signature generation functionality
Signature verification functionality
PKI encryption using key transfer algorithms functionality
PKI decryption using key transfer algorithms functionality
PKIF.dll/libPKIF.so
Certification path processing
Online Certificate Status Protocol Client Functionality
Certificate Revocation List functionality
Symmetric key encryption and decryption
ASN.1 encoding/ decoding
NSS CSP
LDAP
NSS Certificate/CRL stores
CAPI CSP
CAPI Certificate/CRL stores
Operating System
Legend:
IT Environment
PKIF
TOE Boundary
dll/lib composition
API calls

PKIFv2 is a software library and does not do anything in of itself. PKIFv2 is intended to be integrated into a software Application product to provide that product with the capability to use secure PKI-related functionality, including certification path validation, signature generation and decryption, and public key encryption and decryption. In other words, PKIFv2 provides an API to a calling Application to obtain TOE security functionality. PKIFv2 also makes API calls to the IT environment to obtain functionality like cryptography, certificate and CRL storage and retrieval.

For certificate processing, X.509 certificates are used. These certificates are provided from the environment. The TOE does not offer certificate generation and management functionality.

PKIFv2 and the Application require the underlying operating system to provide protection of the PKIFv2 executable, security attributes and data, cryptographic keys and certificates. Providing protection also requires a correct configuration of the Application and TOE and location of the processing resources of the TOE within controlled access facilities.

Note that the application is not automatically become an EAL4 evaluated product just because it uses the evaluated API.   The Application itself will require formal evaluation at EAL4 if required.

The application using the TOE must satisfy the appropriate assumptions and organizational security policies of the IT environment imposed by the Security Target (e.g., application developers are trusted and follows guidance properly) when using the TOE in the evaluated configuration. The application using the TOE, additionally, must also possess the appropriate level of robustness when operate the TOE as intended.

The TOE is provided in the form of binaries that have been compiled for the different operating systems supported by PKIFv2 and the hardware architectures these operating systems run on (see Section 8). The TOE does not directly interface with the hardware, but receives direct support for its security functions from the operating system and cryptographic devices. Provision of identification and authentication of TOE users is a direct dependency of the TSF on the underlying operating system.

6. DOCUMENTATION

The following documents form the TOE guidance:

- PKIFv2 Usage Guide
- Public Key Infrastructure Framework Version 2.1 (PKIFv2) Delivery, Installation, Generation and Start-up Procedures

PKIFv2 is an Open Source product; therefore, its source code and guidance documentation are publicly available for download at http://pkif.sourceforge.net/. There is, however, a process in place by which PKIFv2 binaries, header files and documentation can be verified for integrity and authentication.

7. IT PRODUCT TESTING

7.1. Developer Testing

7.1.1. Functional Test

The developer's test environment for the TOE is comprised of the systems listed in the Security Target (see Section 8) as part of the evaluated configuration. In addition, the developer provided an Apache web server which replicates the functionality of an OCSP responder in the IT environment (for Windows platform) and an NSS keystore (for Linux platform). The developer performed his tests on these platforms. The TOE software was installed and configured as defined in the "Common Criteria compliant installation" page provided as part of the user guidance and in the developer's test plan.

The developer employs a test strategy in which all function testing is executed in an automated fashion. Tests are implemented using the TOE's APIs. Direct test coverage is achieved for the complete TOE security functional interface (TSFI), while subsystem internal interfaces are primarily tested implicitly. In part, the developer utilized the Public Key Interoperability Test Suite (PKITS) developed by NIST to ensure that path validation is performed in accordance with the X.509 and the RFC 3280 specifications.

The developer's test plan provided by the developer organizes the test cases by the functionality they test. This document also maps the TOE security functional requirements and their specifics to associated test cases. The developer additionally maps the test cases to the corresponding TSFI and the high-level design subsystems in a separate supplemental document. The developer's test plan focuses on the security functions identified in the TOE Summary Specification and Security Functional Requirements for the TOE as defined in the Security Target.

Test results were provided by the developer. Test results are written to files. In addition, log files were generated that report more details on the flow of the tests. All tests were executed successfully. The developer tests the TSFI and the internal subsystem interfaces in sufficient detail. The evaluators, thus, ascertained that the testing was complete and comprehensive, covering explicit functionality as well as error conditions (e.g., invalid parameters, invalid credentials).

7.1.2. IT Environment Test

The developer additionally provided tests to demonstrate that the TOE environment satisfies the IT environment security requirements specified in the ST claiming conformance with a Protection Profile in the family of the U.S. Government Family of Protection Profiles Public Key-Enabled Applications for Basic Robustness Environments, Version 2.77, February 1, 2007 (PKE PP). The developer tests mainly focus on the IT security requirements which are beyond the requirements previously defined for the respective evaluated underlying operating systems supported in the evaluated configuration of the TOE.

The tests, overall, were performed on same systems and used the same configurations as the developer functional tests. The developer performed these tests manually by applying the test instructions documented in the test case descriptions in the developer's test plan. The test results are presented as screenshot captures. All tests were executed successfully. The test results demonstrated that the TOE environment satisfies the IT environment security requirements levied by the Security Target and claimed PKE Protection Profile.

7.2. Evaluator Testing

As an integral component of testing, the evaluator independently installed and configured the following binary packages provided by the developer:

- PKIFv2 binaries for Windows x86 (32 bit)
- PKIFv2 binaries for Linux x86 (32 bit)

on the following systems for independent testing:

- The PKIFv2 binaries for Windows were installed on a Windows Server 2003;,R2 system running on a 32-bit Intel x86 microprocessor (used for execution of developer tests and evaluator tests).

- The PKIFv2 binaries for Linux were installed on a Red Hat Enterprise Linux 5 WS, Update 1, system running on a 32-bit Intel x86 microprocessor (used for execution of developer tests).

These configurations matched supported platforms as specified in the Security Target. All tests used the TOE in its evaluated configuration, as specified in the Security Target and the "Common Criteria compliant installation" page in the guidance documentation.

The evaluator repeated all of the developer tests. In addition, the evaluator devised tests for a subset of the TOE. The tests are described in the evaluator's test plan. The evaluator testing effort consisted of three parts. The first part is observation of the developer's test approach, the second part is execution of the functional tests devised the evaluator, and the third part is the execution of penetration tests.

The tests were performed at the CCTL facility located in Austin, Texas. The systems available for testing are listed above.

The log files generated by the test cases were analyzed for completeness and failures. The evaluator determined that all security functions defined in the TOE summary specification were addressed.

7.2.1. Independent Test

The evaluator's test suite was judged to be quite complete and comprehensive, and thus the evaluator needed to design relatively few additional tests. However, some additional test cases were developed and executed to verify the interoperability capability of the TOE. These interoperability tests were intended to demonstrate that the TOE correctly implemented the referenced X.509 standards. These interoperability tests primarily focused on the three key areas of the TOE security functionality: certification path processing, signature generation and verification, and data encryption and encryption, which were tested against the certificates, signed messages and the encrypted messages generated by a third party product that claims to be conformant to the same set of standards, for example, X.509 certificate and CMS format.

The following tests in the independent test suite were environment-specific, and thus the testing environment affected the test results:

OCSP response re-play test. This test verifies basic OCSP response functionality, and verifies that the nonce support correctly prevents replay of previously sent responses. To simulate a replay attack, the OCSP responder from the developer is modified to allow specifying a hard-coded nonce value to use in responses to override the nonce value from the request. The resulting OCSP responder is called Bogus-OCSP responder in the evaluator team's independent test.

System time modification test. This test is to demonstrate the system time modification has an impact on certificate validation. An expired invalid certificate could turn to be valid if the system time is altered. During the course of the test, the system date and time was change to a point that was priori to the certificate expiration time.

7.2.2. Penetration Test

The selection of the test subset for penetration testing was drawn primarily from the TOE's vulnerability analysis presented by the developer and the Vulnerability Assessment ETR. The evaluator chose the following areas to be explored during penetration testing:

- Buffer overflows in retrieving ill-formed certificates
- Use of non-FIPS mode crypto algorithms
- OCSP response replay attack
- Altering the system time to have an impact on Time Of Interest
- Using the extended PKIF Time Stamp functionality

Tests were devised for each of the above areas and upon execution, no vulnerabilities were discovered.

8. EVALUATED CONFIGURATION

- Microsoft Windows Server 2003, SP1, 32-bit, running on Intel x86 architectures
- Red Hat Enterprise Linux WS, Update 1, 32-bit, running on Intel x86 architectures

9. RESULTS OF THE EVALUATION

The evaluation team determined the product to be **CC** v2.3 **Part 2 extended, CC v2.3 Part 3 conformant, PKE version 2.77 conformant,** and to meet the requirements of **EAL 4 augmented by ALC_FLR.2**. In short, the product satisfies the security technical requirements specified in *Public Key Infrastructure Framework Security Target,* Version 1.8, 2008-01-8.

10. VALIDATOR COMMENTS

The validation team suggested that usage restrictions of the TOE be specified in the user guidance documentation. The following restrictions are now included in the guidance documentation:

- It is the responsibility of the TOE user to check the size of the objects passed to a Decode operation.
- Users and administrators of the TOE are advised to verify the correct system time periodically.
- The TOE does not verify an invalid pointer to an object provided by an application using the TOE.

- CRLs retrieved from LDAP or HTTP servers to be processed by the TOE may exceed 100 Megabytes.
- Certificates retrieved from HTTP servers to be processed by the TOE may not exceed 7 Megabytes, and certificates retrieved from LDAP directories to be processed by the TOE may not exceed 20 Kilobytes.

11. SECURITY TARGET

The ST, Public Key Infrastructure Framework Security Target, Version 1.8, 2008-01-8 is included here by reference.

12. LIST OF ACRONYMS

AKID	Authority Key Identifier
API	Application Programming Interface
CA	Certificate Authority
CAPI	Cryptographic Application Programming Interface
CC	Common Criteria
CCEVS	Common Criteria Evaluation and Validation Scheme
CCTL	Common Evaluation Testing Laboratory
CEM	Common Evaluation Methodology
CMS	Cryptographic Message Syntax
CPV	Certification Path Validation
CRL	Certificate Revocation List
CSP	Cryptographic Service Provider
DN	Distinguished Name
EAL	Evaluation Assurance Level
ETR	Evaluation Technical Report
LDAP	Lightweight Directory Access Protocol
NIAP	National Information Assurance Partnership

NIST	National Institute of Standards & Technology
NSA	National Security Agency
NSS	Network Security Services
OCSP	Online Certificate Status Protocol
PP	Protection Profile
PKITS	Public Key Interoperability Test Suite
PKE PP	Public Key-Enabled Family of Protection Profile
PKIF	Public Key Infrastructure Framework
SKID	Subject Key Identifier
ST	Security Target
TA	Trust Anchor
TOE	Target of Evaluation
TSF	TOE Security Function
TSFI	TOE Security Function Interface
VR	Validation Report

13. BIBLIOGRAPHY

[1] Common Criteria for Information Technology Security Evaluation – Part 1: Introduction and general model, Version 2.3.

[2] Common Criteria for Information Technology Security Evaluation – Part 2: Security functional requirements, Version 2.3.

[3] Common Criteria for Information Technology Security Evaluation – Part 3: Security assurance requirements, Version 2.3.

[4] Common Evaluation Methodology for Information Technology Security: Evaluation methodology, Version 2.3.

[5] Evaluation Technical Report, for the Public Key Infrastructure Framework Version 2.1, Version 1.0, October 25, 2007.

[6] Public Key Infrastructure Framework Version 2.1 (PKIFv2), PKIFv2 Security Target Version 1.8, January 8, 2008.

[7] Public Key Infrastructure Framework Version 2.1 (PKIFv2) Test Set-up and Execution, Version 1.1, October 18, 2007.

[8] Public Key Infrastructure Framework Version 2.1 CC Test Plan (atsec Proprietary), Version 2.0, October 11, 2007.

[9] Developer Test Logs, September 26, 2007.

[10] Developer IT Environment Test Logs, October 10, 2007.