Smart Cards: The Open Platform Protection Profile (OP3)

Marc Kekicheff, Forough Kashef

Visa International Services Association
Post Office Box 8999
San Francisco, CA 94128-8999

David Brewer

Gamma Secure Systems Limited
Diamond House, 149 Frimley Rd
Camberley, Surrey GU15 2PS, UK

Abstract

Global Platform's "Open Platform Specification" sets a new cross-industry standard for smart cards, governing the loading, installation and removal of applications at any time that the card is on-line during the card lifecycle prior to card termination. The Open Platform Protection Profile (OP3) recasts the Open Platform (OP) security requirements into the language of the Common Criteria (CC) to facilitate the formal evaluation of OP smart cards. In doing so, OP3 stretches the CC to new limits. There are four areas of innovation:

- The use of "packages" to deal with the optional components within the OP Card Specification.
- The use of the CC to evaluate Java Card ™ byte code verification algorithms.
- The integration of the Target of Evaluation (TOE) with the "Card/Chip Operating Environment (COE)" on which the OP software sits.
- The definition of an Application Programming Interface (API), so that applications may use a variety of OP security services without the need to re-evaluate OP each time an application is evaluated.

The paper also discusses other practical aspects of the application of the CC to the OP Program. It is the authors' intention that this paper will help people to apply the CC, particularly those who face similar challenges, and to create awareness of the security needs of smart card technology.

Key Words: Application Code Verification, Common Criteria, Java[1], OP3, Open Platform, Protection Profile, SCSUG-SCPP, Smart Card, Windows for Smart Cards[2].

[1] Sun ™, Sun Microsystems ™, Java ™ are trademarks of Sun Microsystems, Inc.
[2] Windows for Smart Cards ® is a registered trademark of Microsoft Corporation.

Introduction

The paper examines the security requirements of the *OP Card Specification* (OPCS) [1] for reconfigurable smart cards, and how those requirements have been translated into the language of the *CC* [2] to produce the *Open Platform Protection Profile (OP3)* [3]. The work is part of an overall program to establish the trustworthiness of the OP technology and project that trustworthiness to the market place.

The development of the CC is deeply predicated, for historical reasons, on the *"Orange Book"* [4] specification of a "trusted operating system". It is therefore refreshing to apply an established methodology to a new paradigm. The publications of the *Smart Card Security Users' Group Smart Card Protection Profile (SCSUG-SCPP)* [5] has heightened interest in the application of the CC to smart cards, and in so doing explores the application of the CC to chip technology. The present work, OP3, explores the security requirements of reconfigurable smart cards, evaluation by parts, the use of packages and security APIs, and the use of the CC to describe innovative security features such as "application code verification[3]".

The utility of the Common Criteria

In the payment industry it has become usual for payment associations to independently test each issue of smart card technology prior to authorizing its use by members of that association where the cards will carry the brand name of the association and bear its applications. From a vendor's perspective this means having their technology tested by every payment association. There may also be regional requirements, meaning that the technology may also have to be tested in different countries as well before it can be used.

The CC provides an internationally accepted means to specify the scope and nature of security testing for IT products and a means, called the Mutual Recognition Arrangement (MRA), whereby the certified results obtained in one country will be accepted by another. This presents an attractive proposition as it means that vendors would not have to subject their products to so many independent tests. In principle, just one will do.

An important aspect of this work has therefore been to ensure that OP3 stays within the requirements of the MRA.

A new version of the OP Card Specification

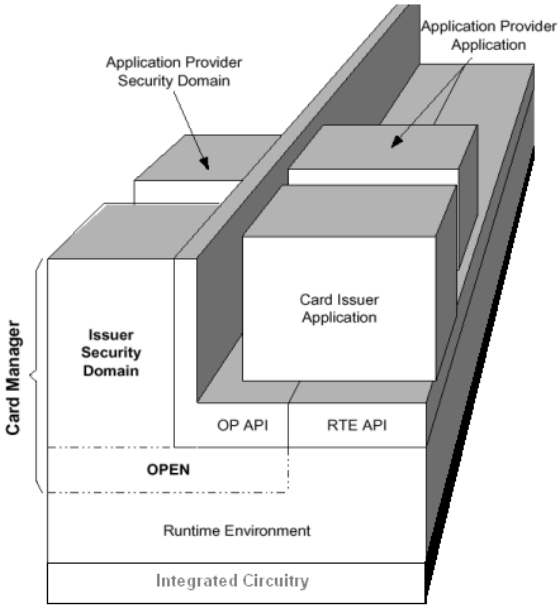
At the time of writing (June 2001), a new version of the Card Specification has just been published and the authors are in the process of updating OP3. In addition to the challenges that producing OP3 that already presented us with, this paper discusses the new challenges that we now face with the update. This paper presents an update of the paper "OP3 – Taking the CC to the Outer Limits" [7], published in October 2000.

[3] In a Java Card ™ context the essential application code verification algorithms are referred to as "byte code verification".

The Open Platform Specification

OP smart cards are reconfigurable multi-application smart cards intended for use by customers of card issuing institutions (referred to as Card Issuers) such as banks. The Card Issuer will issue cards most likely with at least one installed application, such as the Visa Smart Debit/Credit application. However, the Card Issuer may allow other organizations (referred to as Application Providers), such as retailers, to load and install their own applications (for example loyalty programs). Applications are written in a hardware independent programming language such as Java™. OP smart cards will therefore significantly shorten the "time-to-market", allowing new applications to be rapidly developed, piloted, rolled out in vast quantities and easily upgraded in response to market demands. The ability to house several applications on a single card and take advantage of new Internet and wireless technologies also has an enormous appeal to the merchant and consumer alike. Indeed, it is anticipated that the ability to reconfigure the application content of a smart card during its lifetime will be recognized as one of the greatest technological contributions to e-business.

The primary purpose of OP is to manage the loading, installation and deletion of applications for consumer cards. Its secondary purpose, in the form of "specialist cards[4]", is to perform a variety of "off-card" security tasks such as the generation of various Data Authentication Patterns[5] (DAPs) and cryptographic tokens.



The components of the OP architecture (see Figure 1) are the *Card Manager*, the *Security Domains* and the *OP API*. The Card Manager is the on-card representative of the Card Issuer and is the central administrator of the entire card. Security Domains are the on-card representatives of Application Providers and may manage the loading and installation of applications pre-approved by the Card Issuer. Applications may invoke the OP services via the OP API and the services of the COE via the RTE API. OP shares a runtime environment and hardware neutral API (termed the *RTE API*) with the applications. Beneath this is the runtime environment itself (e.g. JavaCard™ or Windows for Smart Cards®) and the integrated circuitry (collectively referred to in OP3 as the *Card/Chip Operating Environment (COE)*). The multiplicity of Security Domains allows each Application Provider's security data (such as cryptographic keys) to be kept separate and private from that of other Application Providers and the Card Issuer. The Card Manager contains a Security Domain (referred to as the *Issuer Security Domain*) for the sole use of the Card Issuer in its role as an Application Provider. In the banking world this duality of role is the

[4] Called "Verification, Security and Authorization Modules".

[5] The term "Data Authentication Pattern" (DAP), used in OP3 and OPCS, represents an additional value associated with a message or block of data. The purpose of a DAP is to provide a method of verifying the transmission, source and/or integrity of either a message or a particular block of data within a message or spread over a number of messages.

norm, rather than the exception. The "OPEN" in Figure 1 is the OP Environment and represents extensions to the RTE to support OP.

The smart card takes power from *a Card Acceptance Device (CAD)* once inserted and is powered down when removed. Communication with the card is via the CAD and an on-card *Application Protocol Data Unit (APDU) interface*[6]. OP authenticates the host computer (e.g. the Card Issuer's card management system) via the CAD and, if appropriate from a business perspective, the host may authenticate the card. The users of OP are the on-card applications and the Card Issuer and Application Providers' host systems. If a Card Issuer offers the cardholder a choice of whether to load an application, the negotiation would be conducted via the CAD and the host system. If this results in a request to load an application, the host machine would establish a secure communication channel (termed the *Secure Channel*) and command OP to load and install the application. OP does not require cardholder authentication. Applications may use the *Secure Channel Protocol* for their own purpose as a means, for example, to achieve mutual authentication and preserving the confidentiality and integrity of APDUs. Indeed, the new version of OPCS offers a choice of Secure Channel Protocols. OP also provides an optional authentication service to applications called the *Cardholder Verification Method (CVM)*, which currently takes the form of a traditional *Personal Identification Number (PIN)*.

Chip card technology is available today in a range of product and price configurations, based on the capabilities and security of card hardware and software. This is reflected in the OP Card Specification by a variety of options that allow Card Issuers to choose products that match their business and security requirements. A minimal OP configuration (that omits the Security Domains) restricts the use of the smart card to the Card Issuer. At the other extreme, a different OP configuration allows Application Providers, with pre-authorization from the Card Issuer, to manage the loading and installation of their own applications. In another configuration, the Card Issuer manages the loading and installation of all applications and therefore acts on behalf of the Application Providers.

Security Requirements

Security Assumptions

It is important to recognize that the OP is merely a component of a much larger system that includes people, organizations and other computer systems. Some of these components are untrusted, whereas others form an integral part of overall system security. For example Card Acceptance Devices (CADs) are untrusted, whereas the buildings that house the back-end systems or hosts would be expected to be secure. In deriving the security requirements for the OP it was therefore necessary to make assumptions about the security that will be provided by the other components of the wider system. A general property of these assumptions is that any failure to meet them would expose an exploitable vulnerability that OP could not possibly protect itself against. The assumptions are fully detailed in the OP3 and concern, for example, roles and authorizations, the COE, the back-end systems and cryptographic key management. Guidelines on how to meet these assumptions, including the security of the development environment, are the subject of other Visa and Global Platform documents.

[6] In the card world, a smart card always plays a passive role, waiting for a command from a host to which it sends a response. The protocol is known as the Application Protocol Data Unit (APDU) protocol and is defined by [ISO 7816-4].

In order to understand the OP security requirements it is particularly important to understand the assumptions made about the COE, which assert that the COE has the following properties:

- It is tamper resistant, making it practically very difficult for an attacker to extract or modify security data directly from the chip by using techniques commonly employed in IC failure analysis and IC reverse engineering efforts.
- It is resistant to differential power analysis.
- Following power loss or smart card withdrawal prior to completion, it will allow OP (or any other SELECTED application[7]), on the next power-up, to complete an interrupted operation successfully, or recover to a consistent and secure state.
- It will handle exceptions raised by applications and report the nature of the exception and the identity of the offending application to OP.
- It prevents the OP security functions from being bypassed, deactivated, corrupted or otherwise circumvented.
- It enforces separation between applications (including OP components) so that one cannot interfere with another or even the COE itself.
- It prevents the contents of programmable non-volatile memory[8] from being accessed when that memory is reused (e.g. so that data belonging to a physically deleted application cannot be accessed).

These requirements share much in common with an Orange Book B3 operating system and, from the perspective of OP, provide a very similar secure operating environment.

Threats to Security

The threats that are endemic to a smart card in general (see SCSUG-SCPP [5], for example) and those that are specific to OP fall into seven groups:

- Group 1 threats concern direct attacks on the chip circuitry using techniques that are usually reserved for testing and debugging chips.
- Group 2 threats concern more sophisticated attacks that monitor the external effects of the chip operation, such as power consumption.
- Group 3 concerns attacks using cards that have yet to be issued, cards from previous issue generations and clones of current cards.
- Group 4 threats concern the card's usual interface to the outside world via the CAD and deals with problems such as those arising from the premature removal of the card from the CAD (often known as "tearing").
- Group 5 deals with attacks on the RTE and OP that are made through the card's interface to the CAD.

[7] The SELECTED application in this case would be the application that was being executed at the time when the failure occurred. SELECT is a standard APDU command

[8] Smart cards will usually contain between 1K – 24K of programmable non-volatile memory (EEPROM) and a similarly limited amount of permanent memory (ROM).

- Group 6 deals with the threats concerning the post-issuance loading of applications and their subsequent need to share resources.

- Group 7 deals with the threats concerning implementation issues in the RTE.

The COE deals with those in Groups 1 through 5. OP deals with those in Groups 4 through 7.

A Group 4 attack would be a power failure, which could arise from the premature removal of the card from the CAD.

OP has an interface to its users. The first point of an attack in Group 5 would therefore be an attempt to *impersonate an authorized user*. If the attack was successful then OP might be fooled into loading unauthorized applications, divulging security data and making unapproved management commands, such as terminating the card. Alternatively, a bone fide user may read, modify, execute or delete applications, information or other resources without having permission from the authority that owns or is responsible for the application, information or resources. For example, without proper security control, an Application Provider might accidentally delete the Card Issuer's applications. Thus the second type of attack is a user, even if properly authenticated, *may attempt to do things that are outside of their intended authorization*. The third form is that a user, even if properly authenticated and authorized, *may systematically experiment with different forms of input in attempt to violate OP security*. This attack is based on the "black box" software engineering technique of establishing the nature of algorithms and predicates ("IF" statements). If carried out exhaustively it could facilitate the reverse engineering of OP as well as the extraction of operational and security related information. A fourth type of attack is that *an attacker might eavesdrop on OP communications with a host*. They might do this as a precursor to masquerading as a Card Issuer or Application Provider, to steal confidential information (including application code) in transit or to record APDU sequences for a subsequent replay attack. The fifth type of attack is the *replay attack* itself. In this case intercepted APDU commands may be replayed, possibly with modification or in a different sequence, to facilitate a successful attack. These five classes of attack are common to all forms of applications.

The first type of a Group 6 attack of relevance to OP concerns the *malicious generation of errors in the set-up sequence* prior to card issue. During the stages of card issuance that involve loading the OP with cryptographic keys, lifecycle states etc, the data itself may be changed from the intended information or may be corrupted. Either event could be an attempt to penetrate the OP security functions or to expose the security in an unauthorized manner. Errors could be generated through simple errors, or through failure of some part of the transfer mechanisms. These errors could occur during loading of programs and/or loading of data. For example, memory usage limits could be exceeded, both at application load and when an application requests data memory. Similar attacks could be attempted when applications are loaded, installed and personalized post issuance. An attacker may utilize *unauthorized applications* to penetrate or modify the OP security functions. Such applications could include copies of authorized applications that had been infected with a virus or a Trojan horse. An attacker might attack an application based on an *analysis of the same application in a different environment*, such as a PC. An attacker might *delete applications without authorization*. Finally, an attacker may *force OP into a non-secure state* through inappropriate termination of selected operations, e.g. through premature termination of transactions or communications between OP and the CAD, insertion of interrupts, or by selecting related applications that may leave files open.

Normally, an adequate trust relationship will exist between the Application Providers and the Card Issuer, which will ensure that Application Providers are confident in the integrity of their applications when the Card Issuer loads them. Application Providers should also have confidence in the effectiveness of the Card Issuer's security systems that would prevent an attacker from modifying the application code prior

to loading. Under certain circumstances, these assumptions may be invalid. In this case the ability of an attacker to *modify an Application Provider's application code prior to dynamic load* poses an additional threat.

Group 7 attacks concern the implementation of the RTE. The first type of attack is that *an application may attempt to bypass the RTE API or OP API in order to defeat the security of OP or the COE*. The second type of attacks concerns the deliberate omission of run-time checks in the RTE. In conventional technologies such as personal computers and mainframes, the operating system enforces security on applications at run-time. To enforce every aspect of security in this way in a smart card is expensive in terms of memory and processor power. The alternative is to ensure, through formal code verification, that applications would pass these run-time checks. Once this has been established, the run-time checks themselves are unnecessary. This approach is particularly necessary in the case of Java Card ™. Of, course, the omission of run-time security checks, in the absence of formal code verification, exposes vulnerabilities in the operating system that could be exploited by a malevolent application. Thus, the second form of a Group 7 attack is: *an application may attempt to exploit the deliberate omission of run-time checks within the RTE*.

Security Functions

The OP Card Specification details a wide variety of security functions to counter the aforementioned threats. Many of these are cryptographic in nature (see below). There are extensive access control rules, which serve to counter the threat that users might attempt to *delete applications without authorization* and that users *may attempt to do things that are outside of their intended authorization*. Some of these rules are discretionary in nature; the discretion being in the hands of the Card Issuer to approve access rights to Application Providers and applications. Other rules are mandatory in nature and are imposed by the OP Card Specification in the form of state transitions. Thus, the OP Card Specification describes a state machine. It refers to those states as the *Card Manager Lifecycle*, *Executable Load File Lifecycle* and *Application Lifecycle* states. A sample card configuration illustrating the possible lifecycle states and transitions of the Card Manager, Executable Load File and applications is illustrated in Figure 2 (see the OP Card Specification Chapter 5 for details). Note the sequence of Card Manager states, which also reflect the overall state of the card. The sequence starts with the OP_READY state. It is in this state that the OP is ready to accept the Card Issuer's instructions for loading cryptographic keys, and the pre-issuance loading and installation of applications. Applications can be loaded for future installation. The card must only be issued when it is in the SECURED state, a requirement that OP3 covers by an assumption. These discretionary and mandatory rules and state transitions serve to frustrate an attacker who *may systematically experiment with different forms of input in attempt to violate OP security*. This objective is further enhanced by rules that allow applications or even the card itself to

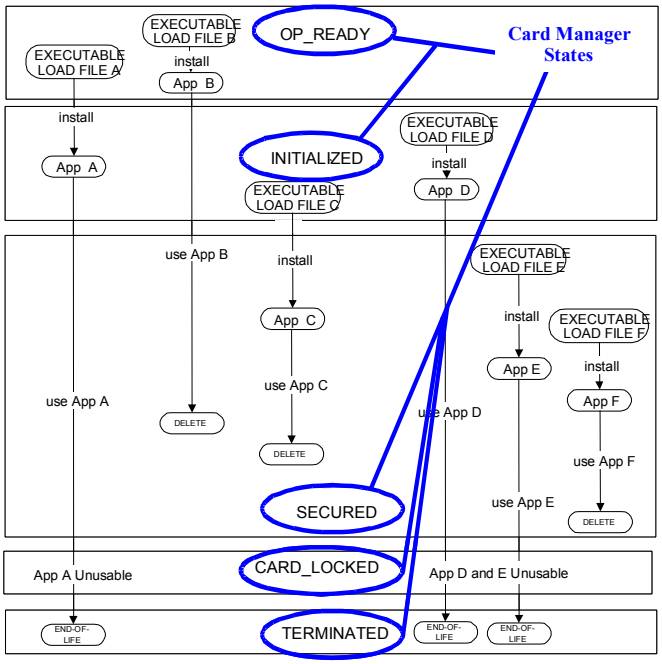


Figure 2: State transitions enforced by the Card Manager

be locked. For example, if an attacker finds a weakness in an application, by *analysis of the same application in a different environment*, the offending application can be locked and subsequently replaced when a remedy has been found. Other security functions check the validity of OP security data and provide a form of intrusion detection. In the latter context OP is empowered to lock applications if it perceives an internal security threat, for example because the application is raising too many runtime exceptions of a particular type. It is also necessary for OP to de-allocate memory in the event of a power failure while loading an application. This provides a defense against an attacker who uses power failure to *force the OP into a non-secure state*.

Cryptographic Solutions

Most of the remaining threats are addressed by cryptographic means. Mechanisms exist to load cryptographic keys both before and after issuance in a secure manner, to generate session keys and destroy keys. Mechanisms also exist to maintain independent key sets for the Card Manager and each Security Domain. Applications may therefore load their own keys encrypted by their Security Domain's key encrypting key.

The Secure Channel Protocols provide *authentication*, *key confidentiality*, *message encryption*, *message authentication* and *MAC[9] chaining*. Secure Channel authentication is achieved using 3-DES through session key agreement by deriving a common set of session keys from static keys and subsequently using the session keys to calculate and to verify authentication data. Message authentication verifies the integrity as well as the authenticity of an APDU command sequence. The integrity of the sequence of commands being transmitted to the card is achieved by using the MAC from the current command as the Initial Chaining Vector (ICV) for the subsequent command. This ensures the card that all commands in a sequence have been received. These security functions counter the threats of *impersonating an authorized user*, *eavesdropping on OP communications*, *replay attack* and *malicious generation of errors in the set-up sequence*.

Additional cryptographic functions using RSA with at least 1024 bit keys and the secure hash algorithm, SHA-1, are used to counter the threat of loading *unauthorized applications* when the task of loading and installing applications has been delegated to an Application Provider. There are two functions. In the first, the Card Issuer digitally signs the application to indicate that it is an authorized application. OP checks the validity of the signature prior to loading and installation. In the second, OP generates a cryptographic receipt to say that the application has been installed (or for that matter deleted) and returns that to the Card Issuer.

Another cryptographic function, which may use RSA/SHA-1 or DES to generate a data authentication pattern, is used to prove that *an Application Provider's application code has not been modified prior to dynamic load*.

Application Code Verification

The only way to address the threat that *an application may attempt to exploit the deliberate omission of run-time checks within the RTE* is to formally verify that the application would pass those run-time checks. The formal verification process, which OP3 calls "Application Code Verification", can be used to address the remaining Group 7 threat, namely that *an application may attempt to bypass the RTE API or OP API in order to defeat the security of OP or the COE*. Application Code Verification could also be used to implement RTE requirements such as information flow (see, for example, [5], page 34) by verifying that they are satisfied by the applications rather than by enforcing the requirements via the COE.

[9] Message Authentication Code

Application Code Verification can be performed on-card or off-card. In the latter case, the verification algorithms could still be implemented on a smart card, but one that is used by the Card Issuer, or some other organization that acts as the "Verification Authority", in a type of hardware security module called a "Verification, Security and Authorization Module" (VSAM). In this case, the VSAM signs the application to indicate that the verification has been performed and that the application has passed. The Verification Authority's Security Domain checks the signature validity and only allows the loading/installation of the application to proceed if the signature is valid.

The Open Platform Protection Profile (OP3)

OP3 follows the usual structure of a Protection Profile (PP), with an *introduction*, a *description of the Target of Evaluation (TOE)*, a definition of the *security environment* (assumptions, policies and threats), *security objectives, IT security requirements* and *rationale*. To enhance readability application notes are included "in-line" so that in defining a security objective, for example, the reader is immediately told which threats that security objective counters. The Evaluation Assurance Level (EAL) requirements are not repeated except where they have been refined. There are four appendices. The first provides guidance on the creation of OP security targets (STs). The second provides a specification for the COE. The third provides guidance to the authors of application PPs and STs and the fourth details the cross-references to the OP Card Specification.

5.2.1.2 STATIC KEY DISTRIBUTION (FCS_CKM.2+1)

The TSF shall distribute **static keys** in accordance with a specified cryptographic key distribution method **([assignment:** ***the relevant APDU command and in association with the appropriate Security Domain*****])** that meets the following: **OPCS ref *e*.**
FCS_CKM.2+1.1

This component is necessary to support the various cryptographic operations that in turn support O.SECURE_COMMUNICATIONS, O.REPLAY_DEFENSE and O.AUTHORIZED_LOADING.

In contrast to FCS_CKM.2+2 (see section 5.8.1.4), in which keys generated on card are distributed to off-card entities, this component receives off card generated keys and distributes them to a Security Domain as appropriate.

Security API Definition: This component is utilized by the [**select:** ***decryptVerifyKey, decryptData***] security API to decrypt a key received by the application within a Secure Channel. It is available to applications to load their own keys encrypted by their Security Domain's K_{SDC}.

The appropriate Security Domain could be the Issuer Security Domain.

Figure 3: An iterated component

Almost every TOE Security Function (TSF) has been refined leaving the ST author little to do save mapping the TSFs onto the functional components that satisfy them. In this sense OP3 is quite unlike other PPs (see [5], [6] for example), and arises simply because the OP has a very specific job to do and in that sense is very "un-generic". In turn, this explains why certain TSFs are iterated several times. For example there are twelve iterations of FCS_COP.1 (cryptographic operation), as there are twelve distinct cryptographic operations.

OP3 either details the specific requirement in the OP Card Specification or refers to them by reference. Figure 3 presents an example. Note the use of the + sign to indicate that the component is iterated and the in-line application notes. The significance of the "Security API Definition" will be explained later.

The Challenges

Producing OP3 has presented a number of challenges.

Challenge No. 1 – Dealing with Optional Components

The first challenge concerned how to deal with the optional requirements in the OP Card Specification. These requirements fall into two categories: those concerning a choice of implementation and those concerning a choice of functionality. The former is dealt with by refining the TSF to offer OP3 specific selections to the ST author, or by assignments that cross-refer to the OP Card Specification where the choice of implementation is specified. For example, FCS_COP.1+8 concerns load file verification, but the choice of whether to use DES or RSA/SHA-1 to implement the data authentication pattern rests with the Card Issuer and the ST author. OP3 says:

> The TSF shall perform **Load File verification** in accordance with a specified cryptographic algorithm **([assignment: valid OPCS specified cryptographic algorithm])** and cryptographic key sizes **([assignment: valid OPCS specified key length])** that meet the following: **([assignment: list of OPCS specified standards] and OPCS ref *ab***. FCS_COP.1+8.1

Which may translate in the ST to

> The TSF shall perform **Load File verification** in accordance with a specified cryptographic algorithm **(*3-DES in CBC mode*)** and cryptographic key sizes **(*double key length*)** that meet the following: **(*ANSI X9.52, FIPS 46/3* and OP Card Specification, paragraphs 4.3.1, 11.2.2, 12.1.2.3, 12.2, 12.3, 13.7, 13.7.1 and 13.7.2**. FCS_COP.1+8.1

Note the instantiation of reference "*ab*" in OP3 to become the more specific references in the OP Card Specification. The permissible instantiations are detailed in Appendix D of OP3.

Choices of functionality are dealt with using the CC concept of a package. OP3 defines seven packages: *Basic*, *Delegated Management*, *DAP Verification, CVM, Intrusion Detection, Application Code Verification and DAP/Token Generation.* There is a set of package selection rules that must be followed, as certain combinations are not permitted. Every OP card configuration (i.e. every TOE configuration) must include the *Basic* Package.

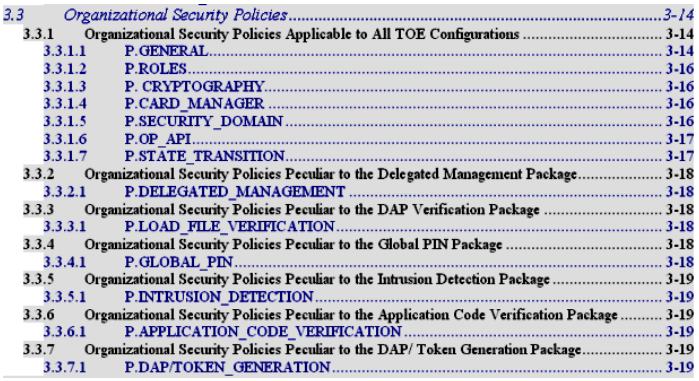


Figure 4: Relationship between packages and policies

The packages are differentiated by their assumptions, organizational security policies, security objectives and TSFs. However, the threats are common to all. Thus there are assumptions, policies and objectives that apply to all TOE configurations and some that only apply to just a particular package, see Figure 4 for an example.

Challenge No. 2 – Specification of the COE

The second challenge concerns how to deal with the COE.

The interface between OP and the COE is either clean or it is not. In the first case, OP can be developed and evaluated separately from the COE. A ST would just comply with OP3, which is the normal practice that is described in the CC. OP3 refers to such a ST as a "Class B" ST.

In the second case, OP has to be considered as an extension to RTE. It has to be developed as part of the RTE and evaluated with it. In the case, the ST has to comply not only with OP3 but it must also include the TSFs (and associated assumptions, policies, threats and objectives) to satisfy the COE requirements[10] and any security functions that must be provided for the applications. Appendix B of OP3 defines the TSFs necessary to satisfy the COE requirements but is silent on the need for any additional security functions to support the applications. This is simply outside the scope of the OP Card Specification and is therefore beyond the scope of OP3. However, other PPs (e.g. [5] and [6]) can be used for this purpose. OP3 refers to these as the "COE PP". The ST (which OP3 refers to as a "Class A" ST) would then be compliant with OP3 and the COE PP. Appendix A of OP3 explains how a Class A ST can be developed using the SCSUG-SCPP [5] as an example. It shows that there are four cases and explains how to deal with them:

- Security Function Requirements (SFRs) that are common to OP3 and the COE PP
- COE PP SFRs that are not used by OP3
- OP3 SFRs not included in the COE PP
- COE PP that are upgraded[11] or iterated by OP3.

Effectively the ST becomes the set-theoretic *union* of OP3 and the COE PP. Where SFRs are upgraded, the ST must include both versions. In other words, the upgraded function does not replace the original COE PP requirement. In this case, we can think of the original function acting across the RTE API and the upgraded function acting solely on, or for the sole use of OP.

Challenge No. 3 – The OP API

From an application perspective OP adds the following security characteristics to those of the COE:

- It is impossible to load an application onto an OP smart card without the authorization of the Card Issuer.
- Card Issuers and Application Providers can check the authenticity and integrity of their applications when they are loaded and can ensure confidentiality of application code and data.
- It is only possible to remove applications from an OP smart card with the authority of its owner.

An Application PP can quote these characteristics, along side those of the COE, as part of it assumptions for when the application is used in a multi-application reconfigurable smart card environment. This leads us to consider the third challenge, which is the use of the OP API. An Application may use this, for example, to deploy the services of its associated Security Domain to establish a secure communication channel using a Secure Channel Protocol. OP3 declares it existence (e.g. with FTP_ITC.1), which ensures that it is evaluated as part of an OP evaluation. However, how does an Application PP or ST invoke it without requiring the OP SFRs to be re-evaluated? For example, if the Application ST includes

[10] See the section of security assumptions earlier in this paper.
[11] For example, the SCSUG-SCPP specifies FDP_RIP.1 whereas OP3 requires FDP_RIP.2.

FTP_ITC.1, even if it mapped[12] on to OP components rather Application components[13], then the evaluator must evaluate it. This is unnecessary. Instead, we want the evaluator to evaluate the invocation of the function not the function itself, that having been done in an OP evaluation. The CC does not provide an elegant way to do this; i.e. there are no CC components that are specifically aimed at defining security APIs.

OP3 overcomes this problem in two ways:

- It identifies those TSFs that form part of the OP API. An example of the identification method is shown in Figure 3. A refinement to ADV_RCR.1 then forces the evaluator to evaluate the OP API as part of the traceability analysis that confirms that the various representations of the TOE are consistent with one another. In this case, the evaluator would be confirming that a specific OP API method (or procedure call) invokes the required TSF.

- It invites the Application PP/ST to include certain assumptions (defined in Appendix C of OP3) that describe the various OP API methods and what they do from a security perspective.

The inclusion of these assumptions in the Application PP/ST forces the evaluator to treat the OP API methods as axiomatic, and therefore the original OP SFRs are not re-evaluated.

The need to evaluate the invocation of the OP API during the evaluation of the Application is left to the ingenuity and requirements of the Application PP/ST author. To prove that an OP API, e.g. the Secure Channel or a CVM is always invoked or is conditionally invoked can be dealt with using the access control SFRs as explained in the next section.

Challenge No. 4 – Application Code Verification

The fourth challenge concerns how to describe the application code verification process and in particular the algorithms. We found that could be done using the access control components FDP_ACC and FDP_ACF, thereby obviating the need to define any new components.

Traditionally, the access control decision is made using a security attribute whose value is already known, for example, in an Orange Book context, the subject *clearance* and/or the object *classification*. In the application code verification case, the value of that attribute (or attributes) is not known at the instance of invoking the access control TSF but can be determined from the Load File (the "object" of the access control decision) by the application code verification process. The application code verification process is not the "subject" of the access control decision. It forms part of the access control decision-making process (sometimes called a "reference monitor").

The use of FDP_ACC and FDP_ACF unambiguously instructs the evaluator to ensure that the access control policy is enforced and, most importantly, to check that the application code verification process implements every verification rule defined by that policy. If the policy is ineffective, for example because of the absence of a rule, then the combination of run-time checks and application code analyses will either fail to implement the OP3 SFRs or an exploitable vulnerability will otherwise be exposed.

The Application Code Verification Policy (P.APPLICATION_CODE_VERIFICATION) requires:

[12] A ST must map the TSFs on to the hardware and software components of the TOE that implement them.
[13] Which begs the question "how does the Application Developer know what the OP components are?"

- Each application code verification process shall be conducted in accordance with [**assignment:** ***list of {process identifier, reference to the specification of the verification process that is implemented}***]

- Each application code verification process shall be defined as a set of "security attributes". Each security attribute shall be refined in terms of a set of verification tests or rules.

Thus each application code verification algorithm (or test) is declared during the instantiation of the first clause of the policy. When executed it is assigned a true or false value depending on the outcome of the test. The corresponding TSF (FDP_ACF.1+6.3) then states, "All the rules defined by the application code verification processes must prove true".

Challenge No. 5 – Updating the OP Card Specification

The fifth challenge has been to minimize the impact in OP3 of changes to the OP Card Specification. OP3, in the main, is already compatible with OPCS Versions 2.0.1 and 2.1. It is intended that the current round of OP3 updates will complete this. It is then the intention to transfer OP3 to Global Platform, register and evaluate OP3.

The resilience of OP3 to changes in the Card Specification has been achieved by a system of indirection. References to normative documents such as the Card Specification are made first to a letter code (e.g. *a*, *b*, *c*, …*aa*, *ab*), as illustrated in Challenge No. 1 above. Appendix D of OP3 contains a table for each version of a normative document. Thus the upgraded version of OP3 will contain two tables for the Card Specification, one for version 2.0.1 and one for version 2.1. Looking the letter code up in the table gives the actual paragraph reference(s) in the Card Specification. Because OP3 refers out to the Card Specification for details of the lifecycle states, cryptographic algorithms and such like, this indirection allows functional changes to be made in the Card Specification without any substantive change to OP3.

The only type of change that this approach cannot deal with is a change that requires the addition of new TSFs. At the time of writing, the best approach to handling this type of change has not been resolved. It may take the form of introducing package "versions" or it might trigger the need for a new version of OP3 specific to the version of the Card Specification that necessitates the change.

Other Observations

Some CC components allow the initiation of a service to be defined but not its termination. There are two instances in OP3: FTP_ITC.1 [Inter-TSF trusted channel] and FPT_RVM.1 [Non-Bypassability of the TOE]. In the first case there is a variety of ways to close the Secure Channel, but they are the only ways defined in the OP Card Specification. In the second case the COE also needs to ensure that control is passed back to the TOE (specifically the Card Manager) as soon as an application completes. OP3 deals with these termination conditions and essential security requirements via special application notes and uses the same refinement to ADV.RCR.1 (see Challenge 3 above) to ensure that the evaluator takes them into account.

The OP requirement to take action against applications that behave suspiciously is also problematic. OP3 addresses the requirement by specifying functionality within the COE to handle application exceptions and report the nature of the failure to the Card Manager. The OP3 then specifies functionality (using FAU_ARP.1 [Security alarms]) to take appropriate action on the COE furnished information. A similar approach is used to handle roll back. In this case, when the COE detects that there has been a power failure (which given the limitations of current smart card technology will usually be on the next power-up), the COE must inform OP of the failure so that it may take appropriate action. The OP3 links these related OP/COE functions via cross references in the in-line rationale statements and application notes. In

a Class B ST an API would need to be defined to do this. In a Class A ST it is anticipated that no action needs to be taken as the ST will include both the OP3 and COE TSFs.

Conclusions

The production of the OP3 has stretched the CC to its limits. The authors have no doubt that improvements to the CC, such as the introduction of components to deal with security APIs, would have made the task easier. Nevertheless, it has proved possible to use the CC in its current form to recast the OP Card Specification in the form of a PP. Thus the goal of keeping OP3 within the requirements of the MRA has been achieved.

The use of OP, the CC and the approaches taken to address the particular challenges in the production of the OP3 should have a number of business benefits:

- Users will have confidence that OP smart cards are implemented correctly in accordance with the OP Card Specification and relevant cryptographic standards and that there are no ways to bypass, corrupt, deactivate or otherwise circumvent its security features.
- Users and card vendors have a choice of OP configurations to meet different budget, market and security requirements.
- Card vendors can evaluate their cards in the country of their choice and have the certificate recognized in other countries worldwide under the Mutual Recognition Arrangement (MRA).
- For Class A STs, card vendors can minimize evaluation risk by evaluating their OP separately against OP3 first and subsequently with the COE against OP3 plus a suitable COE PP.
- Applications can be developed independently of the internal OP/COE requirements, making use of the OP API.
- The ability to use hardware independent languages, such as Java™, allows the rapid development and rollout of new applications in response to market demands.
- The ability to house several applications on a single card and take advantage of new Internet and wireless technologies also has an enormous appeal to the merchant and consumer alike.

The OP Card Specification and OP3 are publicly available *now* at the Visa website (www.visa.com) and the Global Platform website (www.globalplatform.org) respectively. Work has already begun to develop products that meet the OP Card Specification and should soon be available as CC certified versions.

Acknowledgments

The authors would like to thank Dr. Ken Ayer for his enthusiastic support and most helpful comments and suggestions.

References

[1] *The Open Platform Specification, Version 2.0.1 issued December 2000, and Version 2.1 issued June 2001* http://www.globalplatform.org/

[2] *The Common Criteria for Information Technology Security Evaluation Version 2.1, August 1999 (ISO 15408:1999)*

[3] *The Open Platform Protection Profile, Version 0.7 issued January 2001* http://www.visa.com/nt/suppliers/open/docs.html

[4] *Department of Defense Trusted Computer System Evaluation Criteria (TCSEC), DOD 5200.28-STD, December 1985*

[5] *The Smart Card Security User Group Smart Card Protection Profile, Draft Version 2.1d, 21 March 2001* http://csrc.nist.gov/cc/sc/sclist.htm

[6] *Protection Profile 9806 - Smartcard Integrated Circuit (revision of PP 9704 - Smartcard Integrated Circuit), Protection Profile 9810 - Smartcard Embedded Software, Protection Profile 9911 - Smart Card Integrated Circuit with Embedded Software (supersedes PP9809 - Smart Card Integrated Circuit with Embedded Software),* http://www.eurosmart.com *and* http://www.scssi.gouv.fr

[7] *The Open Platform Protection Profile (OP3): Taking the Common Criteria to the Outer Limits, Brewer, Kekicheff, Kashef, Proceedings of the 23rd National Information Systems Security Conference, Baltimore, October 2000*