RC 24219 (W0703-091), 20 March 2007
Computer Science

IBM Research Report

Processor Requirements for a High Security Smart Card Operating System

Paul A. Karger, David C. Toll, and Suzanne K. McIntosh
IBM Research Division
Thomas J. Watson Research Center
P. O. Box 704
Yorktown Heights, NY 10598, USA

Research Division
Almaden – Austin – Beijing – Delhi – Haifa – T.J. Watson – Tokyo – Zurich

This paper has been submitted to the e-Smart 2007 Research Stream.

Processor Requirements for a High Security Smart Card Operating System

Paul A. Karger, David C. Toll, Suzanne K. McIntosh. IBM T. J. Watson Research Center, Hawthorne, NY

Abstract

This paper presents the basic processor requirements to support a high security operating system on a smart card or small embedded processor. In addition to the standard physical protections and cryptographic side channel protections, the basic CPU architecture must meet requirements laid out in the 1970s for virtual memory, I/O access control, execution domains, and process control. The paper shows how CPUs for desktops and servers meet these requirements, many smart card and embedded processors still do not.

1.0 Introduction

Historically, smart card processors have been simple and cheap, and many of them lack the protection facilities that we have all come to expect on conventional computers, such a supervisor and user modes and a memory management unit (MMU). Starting in 1998 with the introduction of the Philips Smart*XA*, some smart card processors have included at least some protection. However, the protection provided varies from processors where the protection works only if the code does not attempt to circumvent it, up to systems where there are true user and supervisor modes and memory virtualization, sufficient to prevent attacks, improper access or covert channels.

The packaging of smart cards, as specified in ISO 7816 parts 1 [8] and 2 [9], leads to well-known security weaknesses from physical and side channel attacks. See section 8.2 of [41]. While in no way minimizing the importance of protection against physical and side channel attacks, the purpose of this paper is to identify the processor security features needed to address other types of software attacks that arise when you allow untrusted applications to run on the card. While these requirements are well known in other parts of the computer industry, the smart card industry has generally not met these requirements, even in chips that are intended to run multiple diverse applications.

A high security operating system, designed to be evaluated under the Common Criteria [16][17][18] at medium (EAL5) and high levels (EAL6 and above), needs adequate protection and virtualization facilities to ensure separation of the operating system from applications, and of applications from each other. This is true, regardless of whether the high security operating system is being implemented on a large server computer or on a smart card. Further, a high security system requires that the processors should have all possible protection against physical and side channel attacks.

Smart cards frequently are called upon to perform cryptographic operations. The communication into and out of the card may optionally be encrypted, according to the secure messaging standards specified in ISO 7816 part 4 [11]. In addition, cryptography is often required for authentication and other operations. Meeting these requirements at adequate performance levels implies that a smart card processor must provide adequate cryptographic hardware; in addition, such hardware must provide adequate protection against side channel attacks and can often more easily provide such protection than can software cryptographic implementations. Cryptographic hardware requirements will be discussed in section 4.0 below.

This document lists the generic requirements on the smart card processor for a high security operating system. It is possible that a chip could meet those requirements and still have some other security problem that is not explicitly covered by these requirements. It is essential that the security engineers, designers, and authors of the operating system examine the chip specifications in detail, to verify that the processor really does provide the required degree of protection.

This paper uses examples of processor architectures, some of which have good security properties and some of which do not. However the examples quoted generally are not smart card processors, because most smart card vendors treat the details of their processor architectures as extremely confidential. This confidentiality frequently includes such items as the instruction set, the details of the processor protection and the specification of the memory management units. To avoid violating any non-disclosure agreements, the examples cited will in general not be smart card processors, because most other processor architectures are not confidential.

2.0 Generic Processor Architecture Requirements

The minimum requirements for a CPU architecture to support a high-security operating system were identified in the 1970s by the MITRE Corporation as part of a larger US Air Force security project. These requirements were broken down into minimum essential features, and other highly desirable features. In 1975, Smith [45] proposed the requirements and evaluated five mainframe computers (the Honeywell 6180, the DEC KI-10, the IBM System/370, the Xerox Sigma 9, and the Burroughs 6700) to see if they met the minimum requirements. In 1978, Tangney [46] did the same for 10 different minicomputers (the Modcomp IV/35, the Prime 400/500, the General Automation GA-16/440, the IBM Series 1/Model 5, the Varian V70, the Interdata 8/32, the Honeywell SCOMP, the DEC PDP-11/45, the Data General Eclipse, and the HP 3000), The Honeywell 6180, DEC KI-10, Honeywell SCOMP, Prime 400/500, and PDP-11/45 were considered the best choices, and only the Burroughs 6700 and HP 3000 did not meet the minimum essential requirements. However, these studies excluded many earlier machines that did not come close to meeting the requirements.

Many of today's smart card processors and low-end embedded processors still do not meet the minimum essential requirements for high-security operating systems, even though 30 years have passed since those requirements were first published. This section of the paper will identify (and in some cases update) those minimum essential features, as well as the highly desirable additional features to assist both smart card operating system designers and chip designers in building better security into smart cards and small embedded processors.

Smith and Tangney identified four major functional areas that are important for security in a CPU architecture, with specific minimum requirements in each area. After summarizing the four areas, they will be discussed individually in some detail in sections 2.1 - 2.4.

1. Virtual Memory
   a. fully virtualized (paged or segmented) address space
   b. null, read-execute, full access rights
2. I/O Access Control
   a. access to I/O devices is controlled
3. Execution Domains
   a. two hierarchically structured domains or rings
   b. controlled transfer into and out of the privileged domain
4. Process Control
   a. multiple processes with distinct address spaces

Smith and Tangney also identified a number of additional desirable features in all four areas that could further improve both the security and the performance of a secure system. These included more fine grained access permissions, ability to securely perform I/O from unprivileged code, DMA controls, multiple protection domains, argument validation, hardware support for domain crossings, support for inter-process communication, etc.

2.1 Fully Virtualized Memory

The basic requirements of a fully virtualized memory is that the application program should not know anything about real physical addresses and that the memory have minimal access control. These requirements are important both for security and non-security reasons and can be met by a variety of memory management units that support paging, segmentation or paging combined with segmentation. Merely providing protection on pages of memory is insufficient. If the physical address of a block of memory is visible to

untrusted software (even if properly protected against unauthorized access), and the operating system offers any type of dynamic memory allocation, then the physical addresses can be exploited as a covert channel. The precise exploitation algorithm would depend on the precise memory allocation algorithm used by the OS. Even if the OS's algorithm is kept secret, it can be reverse engineered by running a sufficient number of test cases.

Fully virtualized memory provides benefits to smart cards, not just for security, but also in application program link editing and loading for multi-application cards. Unless all programs are compiled to position-independent code (PIC), then loading the programs could be very difficult. Either the program would always have to be loaded to its own unique pre-defined address, which could cause problems for programs downloaded in the field, or the smart card OS would have to devote scarce memory space to a linking loader. Since PIC code is often larger than non-PIC code, using an “all PIC code” option may not be desirable on a memory-limited smart card.

We examine three memory architectures below to make the full virtualization issues clearer.

2.1.1 IBM System/360

The original IBM System/360 [5] mainframe memory was not fully virtualized, and used a memory protection scheme similar to that used on some smart cards today. Each 2,048-byte block of memory had a storage protection key to determine who could write that block of memory. Also, some processors (but not all) had optional fetch protection to control read access. Only the System/360 model 67 [6] and some of the later System/370 models[1] supported virtual memory. Of course all the later mainframe models up to and including today's zSeries [7] also have fully virtualized memory. Smart card manufacturers need to make the same transition that IBM did from the weak protection mechanisms of the System/360 to the fully virtualized memory of the System/370 and later models.

2.1.2 ARM Architecture

The ARM Architecture [43] has two different memory systems - the Memory Management Unit (MMU) and the Protection Unit. The MMU provides a fully virtualized address space. However, the Protection Unit, which is designed for simpler implementations, does not support virtual to real address translation, and thus is not suitable for a high security system. This analysis of the ARM protection unit is based on the publicly available specification [43] for embedded processors, and is not based on the analysis of any ARM-based smart card. A smart card processor based on the ARM architecture may use the MMU, the Protection Unit or a completely different design. If based on the Protection Unit, then such an ARM-based smart card does not have adequate security. If based on the MMU or an equivalent fully virtualized design, then an ARM-based smart card would meet the security requirements.

2.1.3 Infineon SLE 88 Smart Card Chip

The memory management security of the Infineon SLE 88 smart card chip has been described by von Oheimb, et.al. [48][47][2]. The SLE 88's 32-bit virtual memory address space is divided into 256 packages of equal size, where the high order 8 bits of the virtual address specify the package number. Packages 0-15 are reserved for system code and future use, while ordinary applications can be loaded into packages 16-255. The use of the high-order address bits to identify protection domains is similar to the DEC VAX-11 architecture [23] and the Data General MV/8000 [38][2]. The VAX-11 architecture used the high-order address bit identify process versus system space, and the Data General MV/8000 used the high-order 3 bits to designate protection rings. All of these schemes are secure and covert-channel free. However, incorporating a protection domain number into the virtual address requires that applications software for the

1 The initial System/370 models did not support virtual memory, but the System/370 models 158 and 168 did. Also the models 135 and 145 were re-announced to have actually supported virtual memory all along. However, the original models 155 and 165 did not support virtual memory without the addition of a dynamic address translation (DAT) box.

2 Infineon is to be commended as the only smart card chip vendor to have published details of their secure memory management approach. We discuss the SLE 88 memory management in some depth, because it is the only smart card memory management unit that is not subject to strict non-disclosure requirements. Other smart card vendors do have fully virtualized memory management, but confidentiality precludes a similar level of discussion.

machine must either be specially linked for the intended protection domain and no other, or the software must be compiled as totally position-independent code (PIC). The VAX-11 had no problem, because a program was either part of the operating system or it was not, and the process space was completely re-mapped whenever a new process was scheduled. The MV/8000 reserved all rings except ring 7 for specially trusted code, so ordinary applications were always mapped into ring 7, and that ring was re-mapped on every process switch. However, trusted applications had to be specially linked for the particular ring number in which they would run.

The SLE 88 views packages 16-255 as equally unprivileged. Any particular smart card will not require a large number of packages, because smart cards have only a limited amount of memory. However, a particular smart card issuer might well need more than 240 distinct packages, even if any particular smart card will only ever load a small fraction of them. For example, an airline might issue a smart card for its frequent flyer program, but need separate packages for all of its partners. One large US-based airline has 21 airline partners, 68 hotel partners, 8 rental car partners, 4 phone partners, 18 bank and financial partners, and 37 other miscellaneous partners for a total of 156 partners. The list of partners changes frequently as some partners may leave and others join, and re-using package numbers could be very dangerous, since there is no easy way to ensure that a particular partner has been removed from all issued smart cards. Furthermore, the software applications need to be downloaded on demand when the card holder wishes to do business with a new partner, rather than being pre-loaded at card-issuing time. Thus, 240 distinct packages is insufficient.

This problem of insufficient package numbers is not insurmountable, but does cause difficulties. Rather than pre-assigning package numbers, each smart card would have to assign package numbers as applications are downloaded. This means that the smart card operating system would have to include a relocating loader which would increase complexity and operating system memory consumption. Some smart card operating systems may include a feature to check digital signatures of applications whenever they are started to ensure that the card has not been tampered. Such checks would be problematic if the application had to relocated (and therefore modified) at download time, as the digital signature applied by the application developer would no longer be valid.

A better approach would be to load all applications into the same package number and re-map them whenever switching applications. However, that would make inter-application sharing difficult and would waste the package number logic incorporated into the chip. A simpler scheme, such as the PDP-11/45 segmentation [21], would seem preferable. The PDP-11/45 has two separate 16-bit virtual address spaces, one for instructions and one for data. Each address space is defined by 8 segmentation registers which provide both protection and address relocation.

It is very important to note that the problems described for the SLE 88 do not make it insecure, unlike the problems described above for the IBM System/360 and the ARM protection unit. The SLE 88 memory management is difficult to use securely for large and diverse multi-application smart cards, but *not* impossible. Similarly, the IBM System/370 (and later models) and the ARM MMU resolve the problems for those architectures.

2.2 I/O Access Control

As a minimum, access to I/O devices must be restricted to privileged code, for two primary reasons. First, the I/O devices may need to be shared between different security domains. The simplest example of this is a disk file system in which you want to control who has access to which files. Second, I/O devices frequently are allowed to write anywhere in primary memory (for example with DMA). Such access must be controlled to ensure that one user's I/O requests do not allow uncontrolled access to some other user's memory.

Smith and Tangney mention a useful (but non mandatory) feature that would allow untrusted user programs to do their own I/O. This can be implemented using an I/O Memory Management Unit (MMU) that provides a fully virtualized address space for I/O operations. The use of an I/O MMU was first proposed in 1975 for the Multics Secure Front-End Processor (SFEP) [29], and the first practical implementation was for the Honeywell SCOMP processor [31] which received the first-ever A1 security evaluation in 1985. Later implementations were done for the Amdahl Multiple-Domain Architecture [35] and IBM's PR/SM [30]. Examples of modern implementations of I/O MMUs can be found in section 2.19 of [1] and in [28].

2.3 Execution Domains

The minimum requirement for security is that the processor have at least two execution domains, one more privileged than the other. The more privileged domain must be protected from the less privileged. This has a variety of implications. For example, if the machine supports hardware stacks, there must be separate stacks for each domain. If the stacks were shared, the less privileged domain might be able to either read secret information from the more privileged domain or to improperly modify critical information upon which the more privileged domain depends. Additionally, there must be a secure mechanism that allows secure transfer of control between the two domains. In particular, the less privileged domain must be able to enter the more privileged domain only at well defined entry points. If entry is possible at any arbitrary point, then various security attacks are possible, such as described in section 3.3.2 of [37].

More specifically, any special purpose registers, I/O control registers, etc., must be protected against user mode access. The only exceptions permitted are those registers that are explicitly needed in User Mode code (for example the “conditions” register holding flags such as the zero or negative flag), and any register where reading its contents cannot be used to reveal secret information or writing the register truly does not cause security problems. In particular, all special function registers such as those that control I/O or the MMU must be protected against user mode access. In addition, all “sensitive” instructions must be forbidden in user mode.

For example, the Phillips Semiconductors XA processor has two protection domains - user and supervisor [24]. However, some of the protection-critical special function registers were accessible from user mode. This was not a serious problem for the original XA processor which had been designed for automotive applications which needed protection for reliability, but not security. However, when Philips planned to use the XA architecture for smart cards, this became a problem, and IBM assisted Phillips in securing the design of the Smart*XA* smart card processor [20]. The details of these changes are Phillips confidential and cannot be detailed in this paper.[1]

2.4 Process Control

Finally, it is essential that the processor support multiple separately protected address spaces, so that one user-level process can be separated from another. This can be as simple as allowing the operating system (in supervisor mode) to reload the memory management unit when it wishes to switch processes. Hardware support for inter-process communication or secure message passing could be a useful additional feature.

3.0 Physical and Side Channel Attacks

Smart card processors are held in a small package with limited physical protection, and hence are subject to physical attacks [26][27], including:

- power glitching
- clock glitching
- out of range temperature attacks
- simple power analysis
- differential power analysis
- radio frequency leakage
- light attacks

There are various defenses available to chip manufacturers to at least partially protect against such attacks. In general such techniques are highly proprietary, and cannot be disclosed in a public forum such as this. Any processor that is to be used for a high-security smart card system must employ a suitable mix of these defenses.

Many of these attacks are directed against cryptographic operations (see the section “4.0 Cryptographic Hardware” below), where the primary purpose of the attack is to foil the crypto or to break the crypto and obtain the key in use. Some of these attacks are described here: [39][32][33][25][44].

1 Phillips Semiconductors is now a separate company - NXP.

Most smart cards are resistant to a lesser or greater degree to at least some of these attacks. Again, for a high security system, it is imperative that the processor chip have as high a resistance as possible to these attacks.

4.0 Cryptographic Hardware

The operations performed by smart card processors frequently involve cryptographic operations. This varies from simple encryption of the communications between the card and its reader to performing sophisticated Diffie Hellman key exchanges and/or public key operations such as signature generation and verification. For example, FIPS 201 [19] specifies the use of a wide variety of cryptographic algorithms. The processor must include crypto hardware to enable the implementation of the following algorithms in a manner that is highly resistant to side channel attacks: DES and triple DES, DSA and RSA, and Elliptic curve.

The AES algorithm is also required in modern smart cards; this should be easy to implement in software, although protecting the implementation against side channel attacks may be difficult. A better solution would be for the processor to include AES hardware that is resistant to attack.

5.0 Interfaces

Smart cards come in two forms, namely “contact” cards which have electrical contacts on one side, and “contactless” cards, which use a radio frequency signal for connection with the outside world. The smart card processor needs electrical power to run, and on a contact card, two of the contacts provide the required power and ground connections. This means that the processors in such cards have an easy and plentiful source of power. However, a contactless card has only the radio signal - apart from providing the communication medium for this card, this is also used to cause electrical power to be generated in the coils of the antenna, thus powering the card. The amount of power available in a contactless card is markedly less than that available in a contact card: this can restrict the capabilities of the card, for example its processing speed since faster processors typically consume more power, particularly when performing cryptographic operations

Smart cards were originally all of the contact type. However, more recently, with improvements in semiconductor technology and the production of chips that require far less power to operate, contactless cards have become much more common. In many ways, contactless cards are far more convenient to use, since they operate merely by proximity to the reader rather than having to be physically inserted into the reader. Even just inserting the card in the proper direction can be tricky for some users. Furthermore, contactless cards do not have contacts that may cause problems as they wear.

Modern standards, for example FIPS 201 [19], specify the use of both contact and contactless chips. Hence any smart card processor for these modern applications must be dual interface, that is the processor must operate in both contact mode according to ISO 7816 [8][9][10] and in contactless mode according to ISO 14443 [12][13][14][15].

There is a serious security and privacy problem with contactless cards in that there is an ID number field that is included in every message in unencrypted form for conflict resolution. This ID number must either be randomized in hardware for each new session or be able to be randomized in software for each new session. If this ID is hardwired into the chip so any specific chip always uses the same number, there is a serious privacy problem which was described by Schneier in [42]. The ID number could be used to surreptitiously track the movements of the card holder.

6.0 Secure Initialization

It is required, for a smart card chip containing a secure operating system, that there be continuous assurance that the chip can never be tampered with from the initial manufacture of the chip itself to the initialization, personalization and issuance of the smart card to the end user. For example, there must be assurance that an un-initialized chip cannot be initialized and/or personalized by an unauthorized person for some nefarious purpose.

This can be achieved by ensuring that, once the card leaves the production line, it is already secure and requires appropriate authentication before any further operation can be performed on or using the card. This in turn requires that certain card specific items be installed in a secure facility into each card *before* that card leaves the secure chip production facility. These items include, but may not be limited to, the following:

- unique chip serial number
- public/private key pair, for authentication and signing
- public/private key pair, for encryption
- the chip certificate (this includes the public keys above), signed by the IC manufacturer's key

Authentication of the card, for example for initialization or personalization, requires that certain other items must also be installed in the chip at this stage, such as the following:

- IC Manufacturer's certificate
- Root Certification Authority's public key
- Certification Authority's public key
- Diffie Hellman Key parameters
- possibly other public keys

7.0 Manufacturer's Code in ROM

Some smart card processors come with a library of crypto code (and other code to drive the processor hardware) included in an area of ROM. In such cases, it is imperative that one of the following applies:

- the source must be made available to the OS development team. This is most important for the vulnerability analysis of the hardware and software as a complete system, and is necessary for a high-level (e.g. EAL7) Common Criteria evaluation.

- alternatively, it must be possible to either replace the special ROM contents with code written by the OS developers, or the chips must completely omit the special ROM code. Leaving the code unused in the ROM could still introduce security problems, particularly if an attacker could force a branch into that code. In this case, it would be highly desirable that the ROM space that is now unused should be made available as part of the main ROM for use by the operating system or other preloaded applications, as memory space on a smart card is always in short supply.

8.0 Common Criteria Chip Evaluation

The smart card chip hardware must be evaluated to EAL5+ at a minimum, either as a separate evaluation to be used for a composite evaluation with the operating system or as part of the operating system evaluation itself. If a composite evaluation is planned, then the full evaluation results (not just the "ETR-lite" - see [3] and [4]) must be made available to avoid the problems discussed extensively by Karger and Kurth [36]. Furthermore, the most commonly used protection profile [22] for smart card chip evaluation is not sufficient, because it assumes that all software code is trusted and installed on the chip prior to delivery of the card to the end user. Since neither of these assumptions is valid for a system supporting the download of potentially hostile code in the field, a revised protection profile or security target is essential. This is extremely important, because the assumptions of all trusted software made in [22] cause the vulnerability analysis to not cover a large variety of serious potential threats.

9.0 Conclusion

We have shown how smart card chips need better security to support untrusted applications, and identified examples from other CPU architectures that should help future smart card chip designers better meet these important security requirements.

Acknowledgements

The authors must acknowledge the help of a variety of people in the writing of this paper, including Elaine Palmer, Sam Weber, David Safford, and others to be added before final publication.

Bibliography

[1] *AMD64 Virtualization Codenamed "Pacifica" Technology: Secure Virtual Machine Architecture Reference Manual*, Publication No. 33047, Revision 3.01, May 2005, Advanced Micro Devices: Sunnyvale, CA. URL: http://www.amd.com/us-en/assets/content_type/white_papers_and_tech_docs/33047.pdf

[2] *ECLIPSE MV/8000 Principles of Operation*, Ordering No. 014-000648, April 1980, Data General Corporation: Westboro, MA. URL: http://bitsavers.org/pdf/dg/mv8000/014-00648_MV8000_PrincOps_Apr80.pdf

[3] *ETR-lite for Composition*, Version 1.1, July 2002, Bundesamt für Sicherheit in der Informationstechnik (BSI): Bonn, Germany. URL: http://www.bsi.de/zertifiz/zert/interpr/etrli11.pdf

[4] *ETR-lite for composition: Annex A Composite smartcard evaluation: Recommended best practice*, Version 1.2, March 2002, Direction Centrale de la Sécurité des Systèmes d'Information (DCSSI): Paris, France. URL: http://www.bsi.de/zertifiz/zert/interpr/etrlia12.pdf

[5] *IBM System/360 Principles of Operation*, A22-6821-0, 1964, IBM Corporation: Poughkeepsie, NY. URL: http://bitsavers.org/pdf/ibm/360/poo/A22-6821-0_360PrincOps.pdf

[6] *IBM System/360 Model 67 Functional Characteristics*, A27-2719-0, 1967, IBM Corporation: Kingston, NY. URL: http://www.bitsavers.org/pdf/ibm/360/funcChar/A27-2719-0_360-67_funcChar.pdf

[7] *z/Architecture Principles of Operation*, SA22-7832-04, September 2005, IBM Corporation: Poughkeepsie, NY. URL: http://publibz.boulder.ibm.com/epubs/pdf/a2278324.pdf

[8] *ISO 7816-1, Identification cards - Integrated circuit(s) with contacts - Part 1: Physical characteristics*, first edition 1987-07-01.

[9] *ISO 7816-2, Identification cards - Integrated circuit(s) with contacts - Part 2: Dimensions and location of the contacts*, first edition 1988-05-15.

[10] *ISO 7816-3, Identification cards - Integrated circuit(s) with contacts - Part 3: Electronic signals and transmission protocols*, first edition 1989-09-15, Amendment 1 1992-12-01.

[11] *ISO 7816-4, Identification cards - Integrated circuit(s) with contacts - Part 4: Interindustry commands for interchange*, first edition 1995-09-01.

[12] *ISO/IEC 14443-1, Identification cards -- Contactless integrated circuit(s) cards -- Proximity cards -- Part 1: Physical characteristics*, first edition 2000.

[13] *ISO/IEC 14443-2, Identification cards -- Contactless integrated circuit(s) cards -- Proximity cards -- Part 2: Radio frequency power and signal interface*, first edition, 2001.

[14] *ISO/IEC 14443-3, Identification cards -- Contactless integrated circuit(s) cards -- Proximity cards -- Part 3: Initialization and anticollision*, first edition, 2001.

[15] *ISO/IEC 14443-4, Identification cards -- Contactless integrated circuit(s) cards -- Proximity cards -- Part 4: Transmission protocol*, first edition, 2001.

[16] *Common Criteria for Information Technology Security Evaluation, Part 1: Introduction and general model, Version 2.3*, August 2005.

[17] *Common Criteria for Information Technology Security Evaluation, Part 2: Security functional requirements, Version 2.3*, August 2005.

[18] *Common Criteria for Information Technology Security Evaluation, Part3: Security assurance requirements, Version 2.3*, August 2005.

[19] *FIPS PUB 201-1, Personal Identity Verification (PIV) for Federal Employees and Contractors*, Computer Security Division, National Institute of Standards and Technology, March 2006.

[20] *P16WX064 SmartXA-Family: Secure 16-bit Smart Card Controller*, Short Form Specification Revision 1.1, February 2001, Philips Semiconductors. URL: http://www-us.semiconductors.philips.com/acrobat/other/identification/smartxa_ls.pdf

[21] *PDP-11/45 Processor Handbook*, Order Code 67.00473.2743, 1973, Digital Equipment Corporation: Maynard, MA.

[22] *Smartcard IC Platform Protection Profile*, BSI-PP-0002-2001, July 2001, developed by Atmel Smart Card ICs, Hitachi Europe Ltd., Infineon Technologies AG, and Philips Semiconductors, registered and certified by Bundesamt für Sicherheit in der Informationstechnik (BSI): Bonn, Germany. URL: http://www.bsi.bund.de/cc/pplist/ssvgpp01.pdf

[23] *VAX-11 Architecture Reference Manual*, EK-VAXAR-RM-001, Revision 6.1, 20 May 1982, Digital Equipment Corporation: Bedford, MA. URL: http://www.bitsavers.org/pdf/dec/vax/archSpec/EK-VAXAR-RM-001_Arch_May82.pdf

[24] *XA User Guide*, 1998, Philips Electronics North America Corporation. URL: http://www-us.semiconductors.philips.com/acrobat/various/XA_USER_GUIDE_1.pdf

[25] Agrawal, D., B. Archambeault, J.R. Rao, and P. Rohatgi. *The EM Side-Channel(s)*. in **Cryptographic Hardware and Embedded Systems** - CHES 2002. 2002, Redwood Shores, CA:Lecture Notes in Computer Science Vol. 2523. Springer Verlag. p. 29-45.

[26] Anderson, R. and M. Kuhn. *Low Cost Attacks on Tamper Resistant Devices*. in **Security Protocols, 5th International Workshop Proceedings**. April 1997, Paris:Lecture Notes in Computer Science Vol. 1361. Springer-Verlag. p. 125-136.

[27] Anderson, R. and M. Kuhn. *Tamper Resistance - a Cautionary Note*. in **Second USENIX Workshop on Electronic Commerce Proceedings**. 1996, Oakland, CA: USENIX Association. p. 1-11.

[28] Armstrong, W.J., R.L. Amdt, D.C. Boutcher, R.G. Kovacs, D. Larson, K.A. Lucke, N. Nayar, and R.C. Swanberg, *Advanced Virtualization Capabilities of POWER5 Systems.* **IBM Journal of Research and Development**, July/September 2005. **49**(4/5): p. 523-532. URL: http://www.research.ibm.com/journal/rd/494/armstrong.html

[29] Biba, K.J., S.R. Ames, E.L. Burke, P.A. Karger, W.R. Price, R.R. Schell, and W.L. Schiller, *A Preliminary Specification of a Multics Security Kernel*, WP-20119, April 1975, The MITRE Corporation: Bedford, MA.

[30] Borden, T.L., J.P. Hennessy, and J.W. Rymarczyk, *Multiple Operating Systems on One Processor Complex*. **IBM Systems Journal**, 1989. **28**(1): p. 104-123. URL: http://www.research.ibm.com/journal/sj/281/ibmsj2801H.pdf

[31] Broadbridge, R. and J. Mekota, *Secure Communications Processor Specification*, ESD-TR-76-351, Vol. II, June 1976, Honeywell Information Systems, Inc., McLean, VA: HQ Electronic Systems Division, Hanscom AFB, MA.

[32] Chari, S., C. Jutla, J.R. Rao, and P. Rohatgi. *Towards Sound Countermeasures to Counteract Power-Analysis Attacks*. in **Proceedings of Crypto '99**. August 1999, Santa Barbara, CA: Vol. LNCS 1666. Springer Verlag. p. 398-412.

[33] Chari, S., J.R. Rao, and P. Rohatgi. *Template Attacks*. in **Cryptographic Hardware and Embedded Systems - CHES 2002**. 2002, Redwood Shores, CA:Lecture Notes in Computer Science Vol. 2523. Springer Verlag. p. 13-28.

[34] Chari, S.N., V.V. Diluoffo, P.A. Karger, E.R. Palmer, T. Rabin, J.R. Rao, P. Rohatgi, H. Scherzer, M. Steiner, and D.C. Toll, *Method, Apparatus and System for Resistance to Side Channel Attacks on Random Number Generators*, United States Patent Application No. US 2006/0104443 A1. Filed 12 November 2004, Application Published 18 May 2006.

[35] Doran, R.W., *Amdahl Multiple-Domain Architecture.* **Computer**, October 1988. **21**(10): p. 20-28.

[36] Karger, P.A. and H. Kurth. *Increased Information Flow Needs for High-Assurance Composite Evaluations*. in **Second IEEE International Information Assurance Workshop**. 8-9 April 2004, Charlotte, NC: IEEE Computer Society. p. 129-140.

[37] Karger, P.A. and R.R. Schell, *Multics Security Evaluation: Vulnerability Analysis*, ESD-TR-74-193, Vol. II, June 1974, HQ Electronic Systems Division: Hanscom AFB, MA. URL: http://csrc.nist.gov/publications/history/karg74.pdf

[38] Kidder, T., *The Soul of a New Machine*. 1981, Boston, MA: Little, Brown and Company.

[39] Kocher, P., J. Jaffe, and B. Jun. *Differential Power Analysis: Leaking Secrets*. in **Proceedings of Crypto '99**. August 1999, Santa Barbara, CA:Lecture Notes in Computer Science Vol. 1666. Springer Verlag. p. 388-397.

[40] Organick, E.I., *The Multics System: An Examination of Its Structure*. 1972, Cambridge, MA: The MIT Press.

[41] Rankl, W. and W. Effing, *Smart Card Handbook: Third Edition*. 2003, Chichester, England: John Wiley & Sons.

[42] Schneier, B., *Fatal Flaw Weakens RFID Passports.* **Wired News**, 3 November 2005. URL: http://www.wired.com/news/privacy/0,1848,69453,00.html

[43] Seal, D., ed. *ARM Architecture Manual*. Second ed. 2001, Pearson Education, Ltd., Addison-Wesley: Harlow, England.

[44] Skorobogatov, S.P. and R.J. Anderson. *Optical Fault Induction Attacks*. in **Cryptographic Hardware and Embedded Systems - CHES 2002**. 2002, Redwood Shores, CA:Lecture Notes in Computer Science Vol. 2523. Springer Verlag. p. 2-12.

[45] Smith, L.A., *Architectures for Secure Computing Systems*, ESD-TR-75-51, MTR-2772, AD A009221, April 1975, The MITRE Corporation: Bedford, MA, HQ Electronic Systems Division, Hanscom AFB, MA. URL: http://stinet.dtic.mil/oai/oai?&verb=getRecord&metadataPrefix=html&identifier=ADA009221

[46] Tangney, J.D., *Minicomputer Architectures for Effective Security Kernel Implementations*, MTR-3531, ESD-TR-78-170, AD A059449, October 1978, The MITRE Corporation: Bedford, MA, HQ Electronic Systems Division, Hanscom AFB, MA. URL: http://stinet.dtic.mil/oai/oai?&verb=getRecord&metadataPrefix=html&identifier=ADA059449

[47] von Oheimb, D., V. Lotz, and G. Walter, *Analyzing SLE 88 memory management security using Interacting State Machines.* **International Journal of Information Security**, June 2005. **4**(3): p. 155-171. URL: http://www.springerlink.com/content/r5neq7jw6m5u34na

[48] von Oheimb, D., G. Walter, and V. Lotz. *A Formal Security Model of the Infineon SLE 88 Smart Card Memory Management*. in **8th European Symposium on Research in Computer Security (ESORICS)**. 13-15 October 2003, Gjøvik, Norway:Lecture Notes in Computer Science Vol. 2808. Springer Verlag. p. 217-234.