Reconfigurable Cryptographic Processor

Ricardo Chaves[1,2], Georgi Kuzmanov[2], Stamatis Vassiliadis[2], and Leonel Sousa[1]

[1]Instituto Superior Técnico/INESC-ID [2]Computer Engineering Lab, EEMCS, TUDelft
http://sips.inesc-id.pt/ http://ce.et.tudelft.nl/
{ricardo.chaves, las}@inesc-id.pt {G.Kuzmanov, s.vassiliadis}@ewi.tudelft.nl

Abstract— **This paper presents the Trusted Computing problem and proposes a reconfigurable approach to the existing solution, the Trusted Computing module chip. The new Trusted Computing approach to computation provides additional safety to the user's data and a higher protection against malicious attacks to the systems. The features proposed by the Trusted Computing Group include secure I/O, memory curtaining, sealed storage and remote attestation, which have a strong dependency on the existing algorithms. A overview to the performance of the already implemented encryption algorithms required for the Trusted Computing is also presented. As a test and prototype platform the Molen [19] computational paradigm has been used. The results suggest that a significant throughput can be achieved with a reduce device occupation. The most significant symmetrical and hashing function are the AES [11] algorithm and the SHA [10] hashing functions. The usage of these cores with a polymorphic approach, suggest that processing throughputs above 1Gbit/s can be achieved with a low device occupation. Preliminary figures for the whirlpool hashing function, recently proposed, is also presented. Future research direction are also presented.**

Keywords— **Trusted Computing, Hardware/Software implementation, AES, SHA, whirlpool, FPGA, Polymorphic processor**

I. Trusted computing

The definition of Trust states: firm reliance on the integrity; or the condition and resulting obligation of having confidence placed in one.

According to the definition of trust, trusted computing can be interpreted as the ability of having computational systems that are reliable and can maintain computational integrity, even when hardware degrading occurs. This may be an important issue since according to some studies the existing technology will reach a point where system degradation will occur with a significantly higher probability [6,7,13]. These failures can be minimized by the usage of self checking designs and the use o redundant logic capable of functionally replacing the damaged part of the circuit.

The work presented in this paper, is however, more focused on the other definition of trusted computing. The capability of assuring confidence on the computational system that is being used. With the increase and proliferation of communication systems, the users privacy and is data coherence is constantly at risk. Either remotely with the use of programs developed to examine or modify the existing data and the systems usage (e.g. virus and worms), or locally through the monitoring of the systems behavior (e.g. printing a document form an unauthorized computer) or through physical attacks (e.g. observation of the power consumption, reading the data stored in memory).

A significant part of these security issues are resolve with use of encryption algorithms. However these algorithms have significant computational requirements and different computational characteristics, so even if hardware accelerators exist to speed up these calculations they can not efficiently improve all the existing algorithms. With this in mind the major software and hardware manufactures created the Trusted Computing Alliance Platform in order to normalize and to catalyze the use of security systems in order to achieve more trustworthy computational systems.

Trusted Computing Group: This Trusted Computing Alliance Platform (TCAP), a consortium formed by Microsoft [9], Intel [4,5], IBM, AMD [9], Sun Microsystems, HP among many other, also designated by Trusted Computing Group (TCG), have established a set of features that may eventually be used in future generation of computers providing new standard for trusted computing [14, 16, 17]. These new capabilities are to be integrated in the hardware and in the software application.

This group developed the Trusted Platform chip (TPM), which provide the hardware acceleration for the proposed features, namely: Secure I/O; Memory curtaining; Sealed storage; and Remote attestation.

Secure input and output: The secure Input and Output (I/O) feature consists on the validation of the received data via using checksums to verify that the software used to do the I/O has not been tampered with. For example a virus trying to snoop the communication between the computer and a credit card reading device.

Memory curtaining: Memory curtaining consists on al-

lowing access to a memory region only to the corresponding software application, thus preventing other applications (e.g. virus) of accessing to critical data that can be miss used, even if the malicious application took control of operating system. Even though the TCG proposes the implementation of this feature in hardware, it can also be implemented in software, but doing it in hardware requires less code to be rewritten.

Sealed storage: Sealed storage consists in storing encrypted data into memory. The key used to encrypt the data is generated as a combination of the software application and the machines hardware, this means that only a given combination of software and hardware is capable of correctly accessing the data stored in memory. This mechanism protects the users information of being read by a different application (or an adulterated of the original software) or from being read in an unauthorized machine.

Remote attestation: With remote attestation the software or a combination of software and hardware can be authenticated, generating a digital signature for the software being used and in which machine. This digital signature is used to assure a remote recipient that the data was constructed by a non forged, cryptographically identified trusted application.

Remote attestation is usually used with public-key encryption, in order to guarantee that only the application that requested the authentication can read the digital signature, other wise, other applications or users could be able to identify which applications the user has been using.

Drawbacks: The use of the TPM chip is capable of supping additional security mechanisms to the current computational systems, however it possesses some drawbacks. The users can not modify the software he is using, since that would invalidate its specific digital signature, making it unusable when interacting with other applications that require a valid a valid signature or when trying to access previously saved data with the Sealed Storage mechanism.

With the evolution of the encryption algorithms the system will became obsolete, since the TPM has no adaptation capability. For example only in the recent revision of the trusted computing group as the AES encryption been included, becoming a mandatory algorithm [11, 15]. With such a static system older versions of software will became unusable and new software will not be able to access data stored by older application, that used different encryption algorithms.

The machine owner is obligated to use the trusted platform module has a black box, having no knowledge on how the module is implemented, if it is properly implemented, or if there are any backdoors to the system.

Trusted computing in reconfigurable devices: Some of the drawbacks of the Trusted Computing Module can be solved with the use of reconfigurable systems. Current reconfigurable systems are capable of achieving a computational capability, which allows them to be used instead of dedicated hardware structures.

Such an implementation will allow this protocol to by used in a big variety of reconfigurable computational systems, such as soft-cores, polymorphic systems [19]. This will allow the system to be easily updated whenever a new version of the protocol is specified. It will give control to the hardware and system designers, and will no longer become a black box as the TCM chips, increasing the trust on the system since the user will be have the information on how the system is design. Specific systems that require some of the features of the trusted computing module, will also be able to efficiently use the proposed processor, by selecting and using only those features required for there systems.

This paper is organized as follows. The next section presents an overview of the computational organization of the proposed processor. Section 3 presents the cryptography cores already developed and its performances. Section 4 concludes this paper with some final remarks and future research directions.

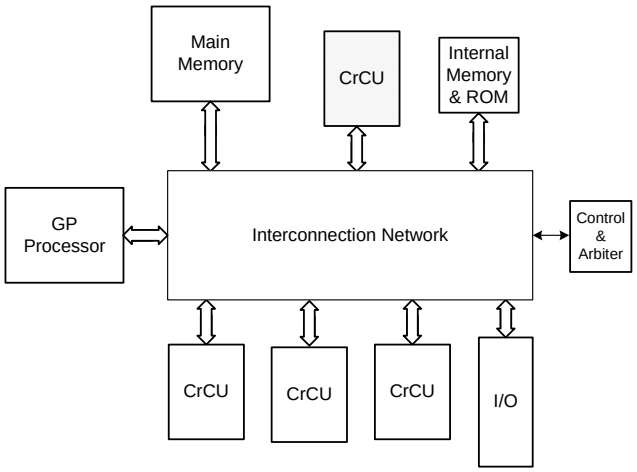
II. Proposed Cryptographic Processor

The proposed approach is user oriented, in other words, the user has more access to the system unlike the strictly closed system of the TCM. This work is focused on the functionality and the comparability of the system with new algorithms and standards while maintaining a strong backward comparability. The proposed processor is manly focused on the security provided by the cryptographic algorithms and in providing a computational performance, that allows these algorithms to be properly used while maintaining the systems performance.

Some of the functionalities of the TCM are not to be implemented in hardware in the first version of this processor. The most significant one is the memory curtaining functionality. However, the memory curtaining can be implemented in software without significant performance degradation, also, some processors already have similar mechanisms. The concept of memory curtaining is not algorithm dependant, thus no significant changes are expected, unlike the encryption algorithms.

Figure 1 depicts a general organization of the processor. This structure is composed by: a General Purpose Processor (GPP); A internal memory to store critical values. Part of this memory has to be non volatile; Several reconfigurable Cryptographical Computational Units (CrCU); A control unit; An I/O Interface for dedicated peripheral de-

vices; A main data memory.



1: General cryptographic processor organization.

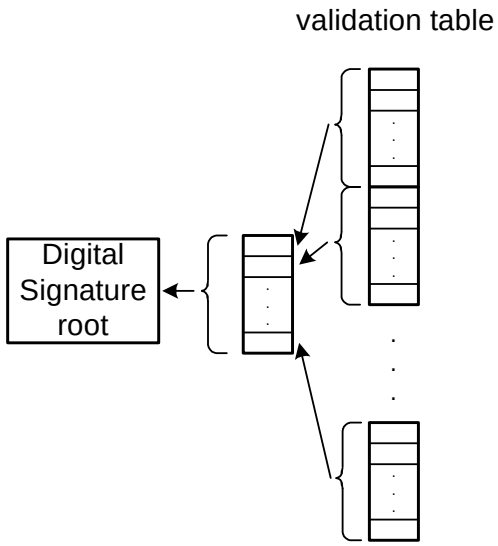
figure

The GPP in this structure is the one responsible for the execution of non critical parts of the execution path as well as for the data flow and which algorithms are to be used. Non critical computation includes key management, key generation and other parts of high level security protocols.

The CrCUs are the ones responsible for the execution of the cryptographical algorithms which can be symmetrical or asymmetrical algorithms and hash functions. These units are configured according to the protocol being used and its respective algorithms. For example one application can be using a protocol that requires the use of 3DES/RSA1024/SHA-128, in which a DES encryption core and a SHA128 hashing core are loaded to the CrCUs, while another one might require AES/RSA1024/Whirlpool, in which case the 3DES and SHA128 cores are replaced by AES and Whirlpool cores.

One of the characteristics of this systems is the attestation of the application that is being executed. In a polymorphic approach to computation an application can be executed either in software or in hardware, thus the test of an application consists not only on the validation of the software being executed but also in the hardware being used. In the software attestation the technique used to perform the attestation is by generating the digest message (DM), or hash value, of the part of code being executed, and compare it with a valid value of the non adulterated DM or Digital Signature (DS). The set of these validation values are from this point on designated by Validation Table. For the hardware part of the application or algorithm the same approach can be taken, taking into account that we are speaking of a reconfigurable structure. This hardware validation can be performed by generating the DS of the bit stream of the respective CrCU, before this is uploaded to the reconfigurable logic.

Due to the physical limitation of the device and efficiency reasons the internal ROM does not store all the digital signatures of all the execution path sections. Instead a Digital Signature of the validation table is stored in this secure internal ROM. Figure 2 depicts a possible example on a two level scheme for such tables, where the initial value is the value stored in the ROM, this value is the DS of table of DS of the validation table. In order to get the DS of a section, e.g. DS2.45, we generate the DS of the code compare it with the entry 45 of the section two of the validation table. If the values match the validation table it self has to checked, this is done by generating a DS of the used section of the validation table and compare it with the level one table. If these values also match the last operation is to check the DS of the first level table against the root value in the ROM. This ROM is used to store the root DS values and critical user passwords.



2: Digital Signatures table hierarchies.

figure

An initialization organization is also stored in this internal ROM. In order to assure that the initial structure is not already compromised, an predefined elementary structure as to be loaded in to the device. Once this is loaded, this structure is responsible for the validation of the next configuration to be loaded and for the assurance that it is an untempered one. Every new configuration is tested by the current existing structure, thus assuring that the existing structure is an untempered one. However, it not efficient to have the DS of every core or code created for every hashing algorithm, thus it becomes necessary to have a permanent CrCU to perform this computation. This is represented by

the grey CrCU in figure 1.

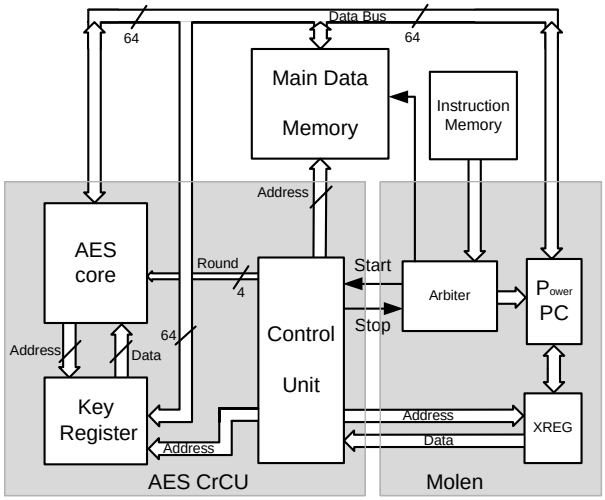
In order to maintain the system performance, of the system with attestation, one can have the dedicated CrCU performing the attestation test to part of the SW/HW that is to be executed next, while the GPP and remaining CrCUs compute the current execution path. In order to have parallel computation it becomes necessary to have a dual port access memory, which might be implemented with several cache systems. The access to interconnection network and to the main data memory by the GPP and the several CrCUs is controlled by a Control and Arbiter unit.

The internal memory can be used to improve the implementation of the sealed storage feature. Whenever the data is to be stored encrypted in memory, the internal memory can be used has a temporary buffer. Thus, the computational units access the decrypted data from this memory, that is only accessible from within the system. The reading and writing to the main memory is performed by an encryption/decryption CrCU. This way the data is only decrypted within the security of system. Identically this internal memory is also used in the secure input and output feature, along with the CrCU units and the I/O hardware of the system.

Current reconfigurable devices do not include non volatile memory components, which would make this systems unsafe to physical attacks. The development by the industry of devices with internal EPROMs will resolve this problem. Additionally to this, it is also important to develop a processor resistent to side channel attacks.

III. Implementation Results

In order to evaluate the proposed processor, several CrCU prototypes have been developed and embedded in the MOLEN polymorphic processor [19]. The MOLEN paradigm [18] is based on the coprocessor architectural paradigm. More specifically, a reconfigurable coprocessor is configured for an application specific operation, while the main program is executed on a General Purpose Processor (GPP). The prototyping platform technology is the XILINX Virtex II Pro, which embeds a PowerPC as a GPP. The prototypes developed so far only include one reconfigurable CrCU. This section depicts an overview of the prototype and the performances analysis of the developed CrCU in the MOLEN processor has well a comparison to the pure software code. In order to use these CrCUs, the pre-existing software applications are compiled by a specialized compiler [12]. Thus, for the software programmer, the usage of the reconfigurable CrCU is transparent, it is used as if the function were implemented in software. This capability allows the CrCU to be used in any existing application with minor modifications to the



3: AES polymorphic processor

figure

already existing software applications.

Figure 3 illustrates the prototype organization with an AES CrCU. While in software functions the parameter passing is done through the stack, in the MOLEN processor these parameter are passed through a dedicated exchange register file, designated by XREG, depicted in Figure 3. Identically to a software function, the data to be ciphered by the CrCU, is accessed directly from the main data memory.

A specialized compiler transfers the function parameters, either from the stack or from the XREG, depending if the functions is implemented in software or hardware. During runtime the arbiter depicted in Figure 3, decides if the instruction is to be executed in the GPP or in the CrCU. The control unit is responsible for retrieving the initialization data and the data blocks from and to the main data and the XREG. This control unit reads the memory pointers from the XREG and addresses the main data memory in order to initialize the CrCU core internal registers. The initialization values are the key addresses, and the begin and end address of the data blocks to be ciphered. The control unit has been described in a fully parameterizable fashion, allowing structural modifications to be performed simply by altering a set of constants. These structural modifications include: the use of a main data memory with different dimensions, influencing the way the data is read; distinct memory organizations.

The proposed AES CrCU implementation, using a fully folded loop version of an AES core [3], capable of encrypting and decrypting data blocks, in both ECB and CBC modes, for all key sizes, i.e., for 10, 12 or 14 rounds. The operation modes as well as the Initialization Vector for the CBC mode are passed as additional parameters in

I: AES CrCU performances table

	Hardware		Software		
Bits	Cycles	(Mbps) ThrPut	Cycles	(Mbps) ThrPut	Kernel SpeedUp
128	646	59	24216	1.59	43
4k	4366	281	738952	1.66	169
128k	31246	1258	23610504	1.67	751

the XREG. This AES CrCU requires 847 Slices and 12 BRAMs. Table I presents the throughput results using a 128-bit key, for the full software and the hybrid software/hardware implementations. The speedup obtained with the AES core, when compared to the software implementation of the AES algorithm is also presented. The obtained speedup is just 43 when only one 128-bit data block is encrypted, due to the overhead to transfer the expanded key (1408 bits). A speedup of 571x is accomplished for a data stream with 16kBytes. Note however, that the overhead of the expanded key transfer is already not significant, especially when considering the encryption of large files. The last column of Table I, contains the calculated speedup for the ciphering kernels. The number of cycles is the same for encryption and decryption, for both the software and hardware implementations. Note that speedups are for the ciphering subroutine (kernel) only and not of the entire application. In a practical application, even if only one data block is ciphered in every function call, the expanded key remains the same. Since the AES core internal registers are not cleared each time a new encryption is executed, they work as static variables. This means that the expanded key has to be transferred only once. In this case, for the ciphering of a single 128-bit data block per function call, a hardware throughput of 89 Mbits/s is obtained, resulting in a local kernel speedup of 56x. Finally, a speedup above 750x is obtained for a data stream of 128$k$bits. A throughput of 1.2Gbit/s can be achieved with this AES CrCU, with small resource utilization, approximately 6% of a Virtex II Pro30 device.

In order to estimate the usability and performance of a more complete system, other CrCUs have also been designed and tested. Table II presents the comparative results for the polymorphic implementations of the hashing functions that are nowadays most used, the SHA-128 and SHA-256. These functions receive a string of data with an arbitrary size and produce an output value, the Digest Message (DM), with a fixed size [10]. The outputted vale is correlated with input data. No significant information about the input data can be obtained from DM, thus these functions are unidirectional. These functions are used to validate blocks of data and are the core operation of Digital Signature (DS) algorithms.

II: SHA CrCUs performances table

	Hardware		Software		
Bits	Cycles	(Mbps) ThrPut	Cycles	(Mbps) ThrPut	Kernel SpeedUp
	SHA-128				
512	396	389	38280	4.01	97
1024	642	479	76308	4.03	119
128k	63126	623	9766128	4.03	155
	SHA-256				
512	354	434	30402	5.05	85
1024	552	556	60546	5.07	109
128k	50088	785	7718646	5.09	153

The algorithms for these hash functions are based in arithmetic operations and have good software implementations. However, as depicted in Table II, the compression rate is still low. With the proposed implementation a throughput above 600 Mbits is achieved, resulting in a speedup, regarding a pure software implementation, up to 150 times, on a Virtex II Pro device. For the SHA-128 CrCU [2], 813 slices are required representing a 6% occupation on an XC2VP30 FPGA, having a compression rate up to 623 Mbit/s with a global clock cycle of 100MHz. The SHA256 CrCU [1] requires 994 Slices using in total 7% of the available resources of an XC2VP30 FPGA, having a compression rate up to 785 Mbit/s with a global clock cycle of 100MHz.

Foreseing the need for more secure hash functions, the SHA512 and the new Whirlpool hash functions have also been implemented. Both of these hash functions produce a Digest Message of 512 bits.

The SHA-512 CrCU [1] requires 1806 Slices using in total 13% of the available resources logic, achieving a compressing rate of of 1.2 Gbit/s. The algorithms of this hash function is computational identical to the SHA-256 hash function, which may in the future have some security flaws [20]. This algorithm is also susceptible to Differential Power Analysis (DPA) attacks [8]. As an alternative, the new Whirlpool hash function can be used. The Whirlpool CrCU requires 2245 Slices (16%) and 32 BRAMs (24%), achieving a data compression rate up to 2.4Gbit/s with a global clock cycle of 100MHz.

For backward comparability reasons, the DES algorithm has also been implemented. The DES CrCU is capable of achieving a throughput of 1Gbit/s, requiring 757 slices.

IV. Future Work

In order to design the processor proposed in section II additional research has to be developed.

A multi CrCU organization has to be developed in order to efficiently used the reconfigurable hardware. In order not to be constantly reconfiguring the processor, there has to be sufficient CrCUs to fully perform a secure communication protocol, which includes the usage of: an hashing algorithm for data verification and validation, digital signatures, and random number generation; A symmetrical algorithm for the encryption or decryption of the data; An asymmetrical or public key algorithm for key transference and for Digital Signatures.

Asymmetrical algorithms also have to be implemented. The most commonly used one is the RSA algorithm, which usually is used with keys from 512 bits to 2048 bits. Another approach to public-key cryptography is the Elliptic curve cryptography (ECC), that is able to provide identical security with smaller keys. To test functionality of the proposed processor, at lest one of these algorithms has to be implemented.

In order to have a secure system, it is not enough to use secure algorithms, other types of cryptanalysis attacks have to be considered, namely, side-channel attacks. The proposed CrCUs must also have architectural defences to protect the user against this type of attacks, e.g. Differential Power Analysis (DPA).

Finally, an efficient method to reconfigure the CrCUs should also be developed, along with the the validation of the loaded configuration, validating the Digital Signatures of each loaded CrCU.

V. Conclusion

This paper presents the proposed reconfigurable approach to the existing Trusted Computing Module. The proposed structure uses reconfigurable Cryptographical Computational Units along with a General Purpose Processor to provide the secure I/O, sealed storage and remote attestation features of the Trusted Computing, described in section I of this paper. This cryptographic processor is capable of using a variety of encryption algorithms in a limited amount of hardware while achieving a very good performance. The reconfigurability characteristics also allow the processor to be updated with new and more secure algorithms and protocol, as they appear, while maintaining a backward compatibility with old protocols and applications. It is also possible to adapt the processor to meet the specific meads of a particular user or utilization. This adaptability is due the fact that the user can load in the CrCU the algorithms he needs and only those.

To test the proposed organization, the Molen [19] computational paradigm has been used. With this computational paradigm only the critical parts of the execution path have to be executed in hardware. It also allows the usage of the CrCU in a simplified manner, identically to a pure software implementation.

The most significant symmetrical and hashing function currently used in security protocols have been implemented and tested, namely the AES algorithm and the SHA hashing functions. Experimental results suggest that encryption throughputs above 1Gbit/s and Digest Message generation with compression rates above 600 Mbit/s, can be achieved, with a low device occupation. Preliminary figures for the whirlpool hashing function, are also presented, suggesting compression rates above 2 Gbit/s. To conclude the paper, future research direction for the development of the full cryptographic processor, are presented.

References

[1] R. Chaves, G.K. Kuzmanov, L. A. Sousa, and S. Vassiliadis. Improving SHA-2 hardware implementations. In *Workshop on Cryptographic Hardware and Embedded Systems, CHES 2006*, October 2006.

[2] R. Chaves, G.K. Kuzmanov, L. A. Sousa, and S. Vassiliadis. Rescheduling for optimized SHA-1 calculation. In *SAMOS Workshop on Computer Systems Architectures Modelling and Simulation*, July 2006.

[3] R. Chaves, G.K. Kuzmanov, S. Vassiliadis, and L. A. Sousa. Reconfigurable memory based AES co-processor. In *Proceedings of the 13th Reconfigurable Architectures Workshop (RAW 2006)*, page 192, April 2006.

[4] intel. LaGrande Technology Architecture Overview. September 2003.

[5] intel. Breakthrough Intel Tecnologies Move from the Descktop to Communications. August 2005.

[6] A. Kulkarni, S.S.; Ebnenasir. The Complexity Of Adding Failsafe Fault-Tolerance. pages 337 – 344. Proceedings. 22nd International Conference on Distributed Computing Systems, July 2002.

[7] Jean-Claude Laprie. Dependable Computing And Fault Tolerance :Concepts And Terminology. volume III, pages 2–11. FTCS, 1996.

[8] Lemke, Schramm, and Paar. DPA on n-bit sized boolean and arithmetic operations and its application to IDEA, RC6, and the HMAC-construction. In *CHES: International Workshop on Cryptographic Hardware and Embedded Systems, CHES, LNCS*, 2004.

[9] microsoft. Next Generation Secure Computing Base. 2003.

[10] NIST. Announcing the standard for secure hash standard, FIPS 180-1. Technical report, National Institute of Standards and Technology, April 1995.

[11] NIST. Announcing the advanced encryption standard (AES), FIPS 197. Technical report, National Institute of Standards and Technology, November 2001.

[12] E. Moscu Panainte, K. Bertels, and S. Vassiliadis. The PowerPC Backend Molen Compiler. In *14th International Conference on Field-Programmable Logic and Applications (FPL)*, pages 434–443, September 2004.

[13] B. Thuraisingham. Dependable Computing For National Se-

curity: A Position Paper. pages 333 – 334. The Sixth International Symposium on Autonomous Decentralized Systems, ISADS, April 2003.

[14] Trusted Computing Group. Main Specification Version 1.1b. 2003.

[15] Trusted Computing Group. TPM v1.2 Specification Changes. October 2003.

[16] Trusted Computing Group. TCG Specification Architecture Overview. April 2004.

[17] Trusted Computing Group. TPM Main Part 2 TPM Structures. February 2005.

[18] S. Vassiliadis, S. Wong, and S. D. Cotofana. The Molen $\rho\mu$-coded Processor. In *11th International Conference on Field-Programmable Logic and Applications (FPL), Springer-Verlag Lecture Notes in Computer Science (LNCS) Vol. 2147*, pages 275–285, August 2001.

[19] S. Vassiliadis, S. Wong, G. N. Gaydadjiev, K. Bertels, G.K. Kuzmanov, and E. Moscu Panainte. The Molen polymorphic processor. *IEEE Transactions on Computers*, pages 1363– 1375, November 2004.

[20] Xiaoyun Wang, Yiqun Lisa Yin, and Hongbo Yu. Finding collisions in the full SHA-1. In Victor Shoup, editor, *CRYPTO*, volume 3621 of *Lecture Notes in Computer Science*, pages 17–36. Springer, 2005.