T-DRE: A Hardware Trusted Computing Base for Direct Recording Electronic Vote Machines

Roberto Gallo[*]
University of Campinas
Campinas, SP, Brazil
gallo@ic.unicamp.br
gallo@kryptus.com

Henrique Kawakami
KRYPTUS Cryptographic Engineering Ltd.
Campinas, SP, Brazil
kawakami@kryptus.com

Ricardo Dahab[†]
University of Campinas
Campinas, SP, Brazil
dahab@ic.unicamp.br

Rafael Azevedo
Tribunal Superior Eleitoral
Brasilia, DF, Brazil
rafael@tse.gov.br

Saulo Lima
Tribunal Superior Eleitoral
Brasilia, DF, Brazil
saulo@tse.gov.br

Guido Araujo[‡]
University of Campinas
Campinas, SP, Brazil
guido@ic.unicamp.br

ABSTRACT

We present a hardware trusted computing base (TCB) aimed at Direct Recording Voting Machines (T-DRE), with novel design features concerning vote privacy, device verifiability, signed-code execution and device resilience. Our proposal is largely compliant with the VVSG (Voluntary Voting System Guidelines), while also strengthening some of its reccomendations. To the best of our knowledge, T-DRE is the first architecture to employ multi-level, certification-based, hardware-enforced privileges to the running software. T-DRE also makes a solid case for the feasibility of strong security systems: it is the basis of 165,000 voting machines, set to be used in a large upcoming national election. In short, our contribution is a viable computational trusted base for both modern and classical voting protocols.

1. INTRODUCTION

Electronic voting systems (EVSs) are a very interesting subject, as they are comprised of system components which interact within an complex environment with boundary conditions of different nature, legal, cultural, logistical and financial. Several countries have adopted EVSs, tailoring them to meet their specificities.

The Brazilian voting system currently has over 135 million registered voters [2], with variable literacy degree. Thus, electronic voting is a very simple procedure, which consists of typing candidates' numbers on a reduced keyboard, guided by simple instructions on a small screen. Brazil adopted Direct Recording Electronic voting machines (DREs from now on) in 1996. In 2009 a decision was made to replace part of the aging hardware base with a newly designed version, while maintaining backward compatibility.

Voting Systems Fundamental Goals

In spite of local constraints, EVSs share six common, fundamental, goals (Sastry [24]):

Goal 1. One voter/one vote. The cast ballots should exactly represent the votes cast by legitimate voters. Malicious parties should not be able to add, duplicate, or delete ballots.

Goal 2. Cast-as-intended. Voters should be able to reliably and easily cast the ballots that they intend to cast.

Goal 3. Counted-as-cast. The final tally should be an accurate count of the ballots that have been cast.

Goal 4. Verifiability. It should be possible for participants in the voting process to prove that the voting system obeys certain properties.

Goal 5. Privacy. Ballots and certain events during the voting process should remain secret.

Goal 6. Coercion resistance. A voter should not be able to prove how she voted, to a third party not present in the voting booth.

These goals are related (e.g. a voting system that does not satisfy goal 5 will hardly satisfy goal 6) and potentially conflicting (e.g. it is not trivial to build a voting system that is totally verifiable while preserving voters' privacy). Third-party end-to-end verifiability has been a recurrent subject [20]. Usually, verifiability is linked to the concept of (statistical) confidence level. Different cultures, and thus electoral laws, have different thresholds for the level of confidence they consider adequate for the electoral process.

[*]Partially funded by KRYPTUS and SERASA Experian research grants

[†]Partially funded by FAPESP (2007/56052-8), CNPq (309491/2008-8), and SERASA Experian research grants

[‡]Partially funded by FAPESP (2010/14492-4) and CNPq (305371/2009-6) research grant

Software independence is not enough. Different voting protocols [3, 17, 5] have been proposed to meet the above goals, with variable degrees of success and effectiveness. Unfortunately, most of them can be defeated by compromised software or hardware running in the underlying computing base. In order to mitigate such threats, software-independent systems were proposed by Rivest and Wack [21]: *A voting system is* software-independent (SI) *if an undetected change or error in its software cannot cause an undetectable change or error in an election outcome.* However strong, this concept ensures most of the above requirements but not all.

For instance, coercion resistance and vote privacy are especially susceptible to attacks based on tampered hardware and software, as vote input devices themselves can leak information [12, 22, 24]. Hardware protection and verification is thus an essential aspect, regardless of whether SI systems are employed or not. While some effort has been done towards the specification of hardware functionalities in order to provide sufficient device accreditation and tamper resistance [19, 8, 24], there is much room for improvement on the path to feasible implementations. Here we follow that path, presenting a hardware trusted computing base (TCB) for direct recording electronic voting architecture, T-DRE in short, suitable for a variety of existing voting protocols and systems.

Summary of our Contributions

Our contributions are present both in the novelty of the T-DRE components and in their composition. Namely, we propose a trusted hardware architecture that extensively employs signed code execution with hardware-enforced access control to peripherals in order to prevent a number of attacks. Further advancements include human-computable device integrity verification mechanisms, strong accountability, and improved signed-code execution assurance, all supported by a certification hierarchy which takes advantage of the proposed hardware.

The T-DRE architecture described herein was adopted by the Brazilian National Election Authority (Tribunal Superior Eleitoral - TSE). In order to fully validate the specification, we first implemented a prototype evaluation platform. Subsequently, the specification was realized by a vendor under TSE's control, using another hardware platform, and taking into account additional costs and stringent field, legal, and resilience restrictions, while maintaining backward compatibility with the deployed base. This endeavor, which resulted in 165,000 produced units, further supports our claims on the feasibility of the architecture.

Our proposal is not an airtight solution to electronic voting; we discuss its limitations in Section 5. However, we do claim that it provides a layer of security to SI and non-SI systems alike, whose strength is degrees above that of voting systems currently deployed around the world, by making it extremely difficult and costly for a fraud attempt to go undetected. Also, although we target centralized elections, in Section 4.2 we discuss how T-DRE can be naturally extended to decentralized environments such as in the USA.

This paper is organized as follows: Section 2 gives practical goals and boundary conditions of voting systems; Section 3 discusses related work; Section 4 details our proposal; Section 5 reports implementation efforts; Section 6 concludes, with ideas for future work.

2. VOTING SYSTEMS PRACTICAL GOALS AND BOUNDARY CONDITIONS

Attaining the fundamental goals is subject to practical boundary conditions, especially in large elections. Three important constraints are:

Availability. Voting systems must be available during the critical periods (election day, tallying, etc.) and resist denial of service attempts. DRE machines must resist tampering;

Credibility. An aspect of utmost importance, it is at the basis of fair representativity. Accordingly, implementations of voting systems should minimize the chance of operational errors and resist tampering. Here, again, DRE hardware security and verifiability plays an important role;

Resource Rationalization. The practical realization of voting systems should take into account various cost-related variables, such as auditing and hardware cost and maintenance. When security is considered, a clear budget trade-off exists between built-in security mechanisms and the security procedures employed by the Electoral Authority (EA). While the first is typically a one-time expenditure which is multiplied by the number of DRE machines, the second is recurrent, flexible, and proportional to the number of polls. The security targets for DRE machines must take this into account.

Security Targets

The specification of security targets should make provisions for many different variables (Common Criteria [27]). In face of the current Brazilian Electoral Laws, the following variables demand special attention:

Window of opportunity. Our implementation should take into account that attacks on DRE machines can occur at any time, but more easily in the interstices between elections. Pre-election time is the most vulnerable due to transportation of DRE machines across huge distances.

Surface and scope of attacks. Voting machines are subject to different levels of adversarial exposure between procedural checkpoints established by the EA: during election interstice, an adversary can have physical access to the DREs; in the pre-election (setup) phase, adversaries may have media (logical) access to the DREs; at election day, adversaries typically have only operational access to DREs, as all non-HID I/O are sealed and the machines operate offline. Our security target take these conditions into account. It provides tampering resistance and tampering evidence on the Critical Security Parameters (CSP) such as keys and key counters, with a physical security target of FIPS 140-2 level 3 [18] (passive resistance). Moreover, a successful attack must have limited scope - breaking one DRE should not increase the chances of an adversary of breaking another.

Level of adversarial expertise. Attacks on a DRE, especially those which adulterate or recover key material or CSPs, must demand multiple experts, considerable time (impossible to execute during election day) and removal to a laboratory with special equipment.

Audit control points, mechanisms and equipments. Audit points shall be precise, clear and accessible. There should be an audit point aggregator that simply expresses the DRE's state (fully operational, in error, in service). The interpretation of this audit point should not require additional equipment nor complex procedures, being accessible to all parties involved in the electoral process: voter, electoral authority, poll worker, and party advocates.

3. RELATED WORK

In this section we discuss related work regarding T-DRE's features.

3.1 Signed Code Execution

Signed code execution [4, 1] is an important tool in voting systems [23, 28]. Many security issues faced by EVSs can be directly mitigated by the proper use of signed code execution. Benefits include:

- ensuring that only official voting software is executed in DREs, enhancing resilience against deliberate adulteration and operational errors which may violate EVS fundamental goals such as vote secrecy and coercion resistance;
- tracing and accountability of incidents, enabling security through legal means;
- simple verification of binaries' integrity in pre, intra and post-election phases, which facilitates auditing by parties, voters, and the Electoral Authority.

Hardware-based signed code execution can be achieved by various means, the *de facto* standard being the Trusted Computing Group (TCG, now ISO/IEC 11889) Personal Computer Trusted Platform Module (TPM) [11], a companion chip to the main system CPU, usually connected via LPC bus. The TPM has functional characteristics similar to a smart card. In cryptographic terms, the TPM performs several operations: key generation, storage and use of cryptographic keys, protected by a key that represents the system's root of trust. Moreover, unlike typical smart cards, the TPM has mechanisms for software attestation, which allows certain running application parameters to be anonymously verified and certified as not tampered. The module is recommended by the VVSG ([28], Section 5.5.1) for protection of the DRE software stack.

One of the drawbacks of PC TPM modules is that they work passively, in hardware terms, with respect to the main system CPU. TPMs, by design, can be completely bypassed by the system's boot sequence if the BIOS (especifically, the "Core Root of Trust for Measurement", CRTM) is tampered with, and thus "deceived" when used in application verification tests. Extensions to the TPM as the TEM from Costan et al [6], being also passive with respect to the CPU, represent no improvement in this regard.

To overcome this master-slave problem, one can consider the sole use of secure processors as the main component of a TCB aimed at DREs. However, even state-of-the-art processors with security features, such as AEGIS [26], USIP-PRO [13] and Cell [25], suffer from impeditive shortcomings. While the AEGIS specification is completely open, to the best of our knowledge there are no commercially available realizations of it. The USIP-PRO, in turn, has limited processing power, its architecture is proprietary and the vendor makes no assertions regarding memory protection against data modification. Finally, the Cell processor is proprietary, not allowing full access to hardware features from independent software vendors, thus adding undesirable obscurity to the design.

3.2 Key Management and Certification

Entertainment platforms have guided the industry regarding the execution of signed code for DRM purposes. Microsoft's Xbox [10] and Sony Playstation 3 execute only code signed by keys directly under vendors' root CAs. With the Cell Processor [25], Sony advances further: unsigned code running on PS3 has limited access to the device's peripherals, notably the GPU. Only signed code has full access to hardware features. The VVSG (Section 5.5.1) forbids non-signed code from running on DRE hardware, similarly to console platforms. The VVSG also recommends a TPM-like component for controlling software execution.

In addition to certifying (signing) the voting machine software stack, cryptographic key material is extensively used in many voting systems [28, 23, 3, 16, 17] for other reasons, from voting, to producing closeout records, audit log signature and verification, to encryption/decryption of votes and other sensitive material.

Although key management and storage could be handled in software by the DRE, cryptographic tamper-resistant hardware is preferred. The VVSG recommends the existence of a hardware tamper-proof signature module (SM) in DREs, whose primary function is to manage the life cycle of two asymmetric key pairs: i) the Election Signature Key (ESK), a unique per-election/per-device key used to sign votes and closeout records; and b) a per-device DRE Signature Key (DSK), which identifies the device and is used to produce certificates for the ESK. The usage of DSK and ESK is strictly controlled by the SM by means of two counters: CountESK and CountDSK. CountDSK counts the number of generated ESK certificates ever signed by DSK. CountESK counts the number of ESK usages. When the closeout record is produced, ESK is erased by the MSM and both counters are included in the resulting record.

3.3 DRE System Verification

Easy auditing is a paramount requirement for voting systems as it is central to the establishment of trust on the DREs' integrity and correct operation. The concerns with integrity verification of the entire DRE system stack (hardware, firmware, and software) are not new. Although auxiliary devices (software or hardware) can be used, ideally solutions should provide effective user-computable verification mechanisms of the DRE integrity, so that less, not more, hardware and software components are used to verify the main system. In this sense, device integrity verification itself should be also software-independent.

Sastry [24] describes a handful of desired DRE verifying properties, mainly aiming at software insulation, by constructing a proof-of-concept DRE with multiple (seven) processors. Gennaro et al [9] establish a condition for tamper-proofness of general hardware and give some clues on how to check device integrity by means of cryptographic challenges. Öksüzoglu and Wallach [19] present, in VoteBox Nano, an elegant human-verifiable software and firmware (FPGA bitstreams) checking mechanism based on random "session identifiers", which change every time the DRE is rebooted. Gallo et al [8] generalize Gennaro et al's conditions, prototyping a human-readable, cryptographically-strong system verification method called Time-Base One-Time Verification (TOTV), which allows for multiple device verification in a trust amplifying fashion, making humans part of the verification protocol. Although both [19, 8] can be used by poll workers and party advocates to assert DRE integrity, they are not practical for large-scale verification by voters, as they require comparison of multiple digit verification numbers, a hindrance when illiterate voters are considered.

4. OUR PROPOSALS

4.1 The T-DRE Architecture and the Master Security Module

The T-DRE architecture was devised to meet security and availability requirements, as well as cost restrictions. Some key requirements are:

- **(R1)** Run solely signed code, even if the opponent has operational access to the DRE media.
- **(R2)** Enforce the verification of the entire software stack, from the BIOS to the voting application, establishing an effective software trust chain;
- **(R3)** Allow the system state (integrity) to be widely attested by any user. Voters, party advocates and the electoral authority (EA) should be able to verify the integrity of the DRE without additional electronic devices;
- **(R4)** Resist physical and logical attacks, preventing unauthorized access to key material and application tampering;
- **(R5)** Contain only fully auditable components, enabling thorough system verification by the EA and the society;
- **(R6)** Allow the use of low cost, widely available hardware components, with reasonable computing power and fully open source development chain;
- **(R7)** Allow maintenance of the DRE machine and upgrade of its cryptographic mechanisms during its long expected lifetime (10 years);
- **(R8)** Enable and ease software and firmware development cycle, including field testing and simulations; allow faithful simulations which are clearly verifiable as such, which includes the production of non-valid results only.

In order to achieve these objectives, we based our proposal on the fundamentals of secure hardware presented by Gennaro et al [9] and Gallo et al [8]. The latter introduces the concept of *cryptographic identity*, which states conditions for the establishment and verification of a root of trust for general secure hardware. Both suggest the use of their verification schemes in DREs. Here we go further, presenting

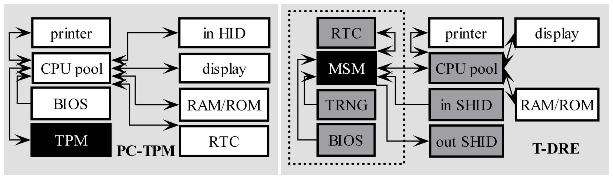

Figure 1: PC-TPM architecture (left) and the T-DRE architecture. The T-DRE components surrounded by the dotted box are under physical protection; BIOS physical protection is optional. Dark-gray components are under MSM direct control.

a DRE system architecture which also brings new control mechanisms and a new verification method (Section 4.3).

Our architecture is depicted in Figure 1, along with a classical PC-TPM system. In both, the CPU pool (one or more main processors) is the main processing unit, which runs the voting application (and software stack). In the PC-TPM design, the CPU pool is the bus master of all peripherals, including the TPM chip, which can be completely bypassed by tampered software at boot time. There is no way for the TPM to prevent CPU access to peripherals, nor to inform users that non-signed code is running.

The T-DRE Architecture, in contrast, is fundamentally different from the PC-TPM: the security is based on the proposed Master Security Module (MSM), which concentrates the DRE's cryptographic mechanisms and controls system peripherals (encrypted voter keypad, poll worker terminal, status lights), BIOS, and CPU pool. This centralization allows for a multi-level certification-based peripherals' access policy which can be enforced on the software running on the CPU pool. This is further explained in Section 4.2. The MSM control over the human interface devices (HID) also plays crucial role in our solution. Its implications are explored in Section 4.3. The MSM is also a CID-enabled device, i.e. a device whose root of trust, represented by a cryptographic key, is bound to the device's physical integrity: crossing the cryptographic boundary is highly likely to cause the device's root key destruction (and thus its identity), preventing the production of valid closeout voting records.

The T-DRE Software Verification, in contrast to PC-TPM, allows for full software stack verification, including BIOS. Prior to the CPU boot, after the DRE hardware power-up, the MSM checks the authenticity (and possibly decrypts) the BIOS contents; only if a valid (signed) BIOS is found, the CPU pool is able to boot. Now the CPU runs signed code from the very beginning of the boot sequence and is able to use the MSM to check the remaining of the software stack (bootloader, O.S., voting applications, scripts, configuration data). The differences between the T-DRE and the PC-TPM boot processes are illustrated in Figure 2. It goes beyond VVSG's required signed code verifier hardware module (VVSG, Section 5.5.1).

Both the T-DRE peripheral architecture and the software verification mechanisms are novel to DREs. Moreover, the MSM also acts as a VVSG Signature Module (VVSG Section 5.1.2). In spite of these advancements, our architecture can be implemented with off-the-shelf electronic components, enabling secure, fully auditable systems and low cost realizations. In Section 5 we describe a prototype using only commodity, general purpose components.

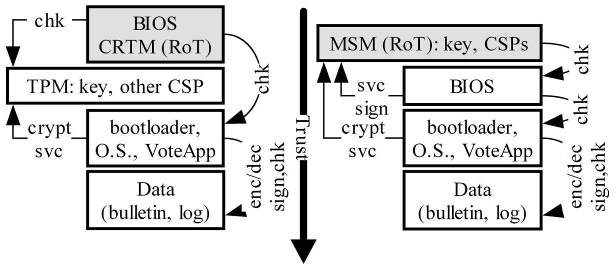

Figure 2: Verification chain for code execution, PC TPM and our proposed MSM

4.2 Hardware-Reinforced Certification-Based Privileges

Satisfying Section 1 goals (in special privacy) and Section 4.1 requisites (in special R3, 5, 7, and 8) requires strict control over the DRE software. Only official (highly audited) voting software must be able to produce valid closeout records. Maintenance (loosely audited) software must be prevented from accessing the DRE's key material (thus preventing production of valid closeout records) and from running an apparently valid, but otherwise fake poll (thus breaking privacy). Also, voting software being developed must be able to exercise all DRE features without being able to produce valid tallies or deceiving voters.

To attain the desired software control, we combined the MSM's control over the DRE's peripherals and the running software stack, with a custom key hierarchy based on Public Key Infrastructure (PKI) technology (with established procedures and audit controls), thus reducing required audit points. Our proposal centers the confidence of the electoral system on the EA root certification authority (EA-rootCA), which is audited (cryptographically) by the parties and the society. Figure 3 illustrates the PKI architecture with its three intermediate CAs, VoteCA, DevelCA, and ServiceCA, each with distinct purposes and privileges. In common, these CAs are responsible for": a) managing the DSK certificate life cycle; b) signing the DRE's software stack; and c) decrypting any messages coming from the DRE, when the voting protocol so demands. Software signed under each certification branch has different execution privileges and access to different key materials. Each DRE has three DSK certificates (and key pairs), one for each tree branch. *All DRE certificates (and corresponding keys) are stored within the MSM, which controls both the key usage and the signed code execution privileges.*

Vote CA Branch: Binaries signed under this branch have total control over the DRE hardware and are used in the actual election days - they have access to the official voting key material (DSK_{vote}, ESK_{vote}), producing valid election closeout records, controlling the voter's keypad use, the poll worker's keypad use, and the access to the Secure Output HID (Section 4.3). The MSM is responsible for enforcing the privileges of the signed code over the DRE hardware, without any software interference.

Development CA Branch enables the necessary functions for development and election simulation activities , granting restricted access to peripherals and keys: i) the MSM produces signatures only with DSK_{devel}, ESK_{devel}, $Other_{devel}$ keys; and ii) the signed code has no access to the secure output HID which signals valid polls. This prevents in-development code from being used to deceive voters, and easily distinguishes valid signatures on real closeout records from those produced under simulation.

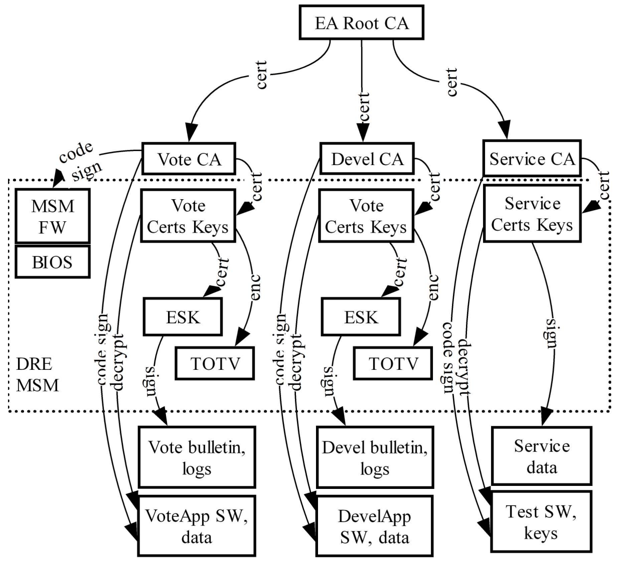

Figure 3: Certification hierarchy, code and data, and key usage

CA Privilegies	VoteCA	DevelCA	ServiceCA
Key Material	$(DSK)_{vote}$ $(ESK)_{vote}$ $(Others)_{vote}$	$(DSK)_{devel}$ $(ESK)_{devel}$ $(Others)_{devel}$	$DSK_{service}$
Input HID access	Full	Full	Restricted
Output HID access	Full	Full	Only test results
Security API (Secure HID)	Full	Restricted	None

Table 1: Signed code execution privileges for our DRE proposal; MSM enforcement

Service CA Branch enables DRE maintenance (memory, battery, peripherals testing and systems components replacement). Servicing operations are highly distributed, thus hard to audit. Under ServiceCA, signed code is not allowed access to keypads nor the secure output HID nor any ESK key material. The allowed operations are: a) re-pairing the input/output of cryptographic devices, and b) signatures of maintenance logs. Table 1 summarizes the privileges enforced by the MSM in each certification branch.

Other Considerations: Although our proposal targets centralized elections, it can be naturally extended to decentralized scenarios, as those in the USA, by adding Local Electoral Authorities CAs (as additional intermediate CAs) to the tree of Figure 3. Then, each local authority would maintain three CAs (VoteCA$_{local}$, DevelCA$_{local}$, ServiceCA$_{local}$). This allows a great deal of independence and flexibility, where local authorities can produce and run their own software without depending on the national authority. Furthermore, DREs can be easily shared by local authorities.

4.3 T-DRE Verification: Secure Human Interface - S-HI

Integrity verification schemes provide variable confidence level in their output. As a rule, the better the scheme the more intrusive an adversary has to be in order to fake a result. From less to more intrusive we list: software modification (SWM), hardware modification (HWM), and key extraction from hardware (KXT). Human verification is especially hard to attain if tampering with the communication channel between the user and the system under verification is a possibility. *We call a human interface secure (S-HI) up to a class of intervention (S-HI-SWM, S-HI-HWM, S-HI-KX) if it does not produce false results even when it is subject to tampering of that class.*

The VoteBox Nano random number display (along with its verification scheme) is S-HI-SWM, i.e., it resists logical (bitstream) attacks, but not S-HI-HWM. In T-DRE we provide users with two interfaces: one S-HI-SWM and one S-HI-HWM. For the S-HI-SWM interface, we employ the MSM (hardware-)controlled 'out SHID' (Figure 1) as a four-state LED which indicates VoteCA, DevelCA, ServiceCA, and non/corrupted signed code. This is a clear improvement over VoteBox nano, as we attain the same security level with a much simpler user verification scheme.

For the S-HI-HWM interface, we employ a modified version of TOTV [8] that does not require the high-stability secure real-time clock (HSSRTC) of Gallo et al's solution. The TOTV protocol is similar to the Time-Base One-Time password (TOTP) described in [15]; TOTP derives, from time to time, an n-digit sequence from a secret key known to the *verified* device and possibly to the *verifier*. It is defined as $TOTP = HOTP(K, T)$ where T represents the number of time steps between the initial counter time T_0 and the current Unix time. K is a key, and $HTOP$ is the HMAC-based One-Time Password Algorithm defined (RFC 4226 [14]) as $HOTP(K, C) = Trunc(HMAC - SHA - 1(K, C))$. The TOTV proposal binds the secret derivation key K to the device's cryptographic identity (CID), so that any attempt to tamper with the device, by construction, should destroy the CID and thus cease the TOTV sequence creation. In our architecture, we maintain two TOTV keys (K_{vote}, K_{devel}) protected by DSK_{vote} and DSK_{devel} keys.

In order to check the integrity of a specific DRE, a user has to access a TOTV sequence produced by the electoral authority. In other to avoid replay attacks, this access must be either i) confidential and prior to the DRE display of the TOTV, or ii) real-time, on-demand, and signed.

In our proposal, we use the same construction as the TOTV, but instead of having a single T representing the number of time steps since Unix epoch, we use two T variables (T_{vote}, T_{devel}). These represent the time steps accumulated during every DRE usage when running in voting mode and development mode, respectively. The time counters necessary for this are made persistent and are protected by the MSM from stalls or decrement. In order to avoid other types of replay attacks, and after signed closeout records are produced by the DRE, it stalls the counter and includes it in the certificate, pausing the timing increments. In the next DRE usage (possibly on the next election), the electoral authority sends the poll workers $TOTV = HTOP(K, T)$, with $T = max(T_{closeout}, T_{user-access})$, which allows for DRE boot-up and counter resumption. The modification from the original TOTV proposal is motivated by the cost of a high stability secure real time clock. The usage of our proposals is further illustrated in Section 5.

5. T-DRE IMPLEMENTATION & RESULTS

The practical realization of our proposals was done in two phases, a prototyping and a mass production phase. In the first, the theoretical, technological, and procedural solutions were tested and validated. In the second, any necessary modifications were implemented.

5.1 Hardware and Firmware Implementation

Prototype Due to the large number of DREs to be produced (165,000), our proposals were thoroughly tested in a prototype prior to the delivery of final specifications to the chosen vendor for mass production. In the prototype (composed by two connected boards: B1 and B2), we instantiated all of the T-DRE main peripherals (Figure 1, namely: MSM, BIOS memory, encrypted voter keyboard (in SHID), output device (serial display), secure output (out SHID), main CPU, among others. The B2 board is a commercial embedded PC, with an AMD Geode LX800 CPU, with 256MB RAM. The B1 is a custom board specifically built for the prototype. It hosts the MSM and other devices, and connects the security module to the bottom board by means of an ISP connection (to BIOS delivery) and a USB connection (for other, cryptographic, services).

Considerable effort was spent on the correct choice of the micro-controller (uC) employed for the MSM as it must conform to many requirements: a) have internal code and data memory (both persistent and volatile); b) the entire memory must be lockable (no read/write access); c) memories must be large enough to handle cryptographic mechanisms (RSA, ECDH, ECDSA, SHA-2, homomorphic DH) and store keys and certificates; and d) reasonable performance, in order to handle quick BIOS verification and cryptographic services.

In our prototype, the MSM was implemented using a NXP LCP2000 (ARM) family uC which meets these requirements: a) up to 1MB internal FLASH memory with code read protection, b) up to 40KB RAM, enough for the implementation of asymmetric algorithms; c) 72MHz, 32-bit core, with 64 DMIPS performance. The voter input device (cryptographic, tamper-resistant physical keyboard) was simulated using a MSP430 uC, connected to the main uC by an SPI bus. The output secure HID is composed by three light emitting diodes (LEDs) which are directly connected to the MSM. In order to provide an onboard source of entropy, we implemented two random number generators using avalanche-effect semiconductor noise.

For the asymmetric algorithms on the MSM and the cryptographic keyboard we used the RELIC library [7]. For our prototype, the implementation of the required MSM functionalities, including DSK and ESK handling, binary code verification, CSR exportation, secure firmware update and cryptographic keyboard handling required about 180Kbyte FLASH (code) memory and 24Kbyte RAM. Employed functions were: signing and verification, asymmetric encryption/decryption (RSA-2048 PKCS#1); hash (FIPS 180-3 SHA-512); block ciphers (FIPS 197 AES 256).

A prototype software stack was also implemented. The bottom board BIOS was modified so that it uses the MSM slave interface to check the bootloader's authenticity. The bootloader was also modified (from GRUB) to test the boot image, rather than files, using the MSM.

5.1.1 *Attacks and Countermeasures*

T-DRE, as PC-TPM, has no effective runtime (after boot) countermeasures against defective software nor buffer overflow attacks (data execution). While the first problem can be traced (and later dealt with) due to the sole use of signed code, the second demands more attention. In Brazil DREs have no data links, so buffer overflow attacks from voters or poll workers keypad is highly unlikely. For further protection, one may consider the "reboot prior to each vote" approach.

Hardware systems are subject to many implementation attacks, in special side-channel analysis (SCA) [12]. SCA use information leaked through side-channels from real systems. More information can be found in [12] and [22]. SCA-aware cryptographic hardware usually resists, to a certain extent, side-channel attacks. However, they typically suffer from lack of transparency on the employed security mechanisms (see Section 3). As we privilege transparency over off-the-shelf solutions, our solution uses a standard uC and added FIPS 140-2 level 3 equivalent physical protection and SCA counter measures:

- The entire top board was immersed in tamper-resistant and -evidencing resin;
- In order to weaken power attacks (SPA, DPA, CPA), we adopted two countermeasures: a) we used decoupling elements in all external communication paths; and b) we filtered and stabilized the power input to prevent energy consumption variation;
- Timing attacks are weakened by using constant-time cryptographic operations.

5.1.2 *Mass Production Versions*

After validation, our architecture was realized in a mass production version, and is set to be used on the 2010 Brazilian national election, with more than 165,000 DREs. This version differs from our prototype in some implementation decisions and functions: a) there is a single board containing all the components required in our architecture; b) the CPU pool was implemented as a single x86 processor; c) the MSM master interface was replaced by an assistive (supervisor) interface; if the MSM perceives any BIOS change, it resets the CPU pool (the main drawback being that BIOS cannot be encrypted). A second mass production version is expected to be manufactured in the fourth quarter of 2010, with more than 200,000 DREs. These will present further side-channel countermeasures and incorporate improvements deemed necessary.

5.2 Usage Procedures

5.2.1 *Pre-Election, Election, and Post-Election Procedures*

Since valid (non-tampered) voting machines run only code signed by the electoral authority, it is easy for a verifier to check whether the voting application is correct and that the voting machines have not been tampered with:

- In the **pre-election** phase, a human verifier must: a) Check for any physical tamper evidences on the DRE; if any are found, stop and report; b) switch on the DRE and enter the "resume TOTV" provided by the electoral authority (Section 4.3); if the DRE fails to continue the boot process, stop (either it is not the correct DRE or the device has been tampered with); c) check for the next TOTV to be shown by the DRE; if it is not the expected one, stop (the DRE has been tampered with); d) perform other verification procedures (e.g. audit procedures).
- On **election day**, human verifiers can, at any time: a) check for software stack integrity, by simply checking a DRE's status S-HID (indicative LED); if the S-HID does not present a valid status, the use of that DRE must be prevented (either it has been tampered with or it is not running the correct voting software stack); b) from time-to-time, electoral judges and voters can check for device integrity by comparing the TOTV produced by the DRE with those from the electoral authority; if any comparison fails, stop that DRE's use (it has been tampered with).
- In the **post-election** phase, a human verifier must check whether the final TOTV present in the closeout record is valid; if not, the device has been tampered with and the produced closeout record is deemed invalid.

5.2.2 *Other Procedures: Development, Testing, and Maintenance*

We chose a PKI model for key management, so that its established practices and procedures can be used. The use of the root CA's and the VoteCA' authorization keys is only granted to the highest rank staff of the EA (in Brazil, Supreme Court judges preside the Supreme Electoral Court), audited (cryptographically) by political parties, Congress and society representatives.

6. CONCLUSION AND FUTURE WORK

In this paper we propose T-DRE, a trusted computing base for direct recording electronic voting machines, which is mostly independent of the voting application and largely VVSG-compliant. T-DRE's novel combination of technologies enable device verifiability by humans, deep PKI integration and simple auditing. Our architecture was prototyped and then reengineered for large scale manufacturing, with 165,000 devices produced. These DREs will be used in the Brazilian 2010 presidential election.

T-DRE's main component, the Master Security Module (MSM), unifies the TPM and SM modules proposed in the VVSG and adds key new features by: a) enforcing, over the entire software stack, a policy of multi-level, certificate-based access to peripherals and key material; and b) taking control of human interface devices, thus amplifying vote privacy and user DRE tamper detection.

We also indicate how the new audit and control mechanisms present in our architecture can be integrated into the usual electoral cycle, the voting itself, election simulation, device testing and servicing, and software development.

Currently, we are working on the design of a fully-auditable secure processor to be used as a CPU-MSM for DREs.

7. REFERENCES

[1] R. Anderson, M. Bond, J. Clulow, and S. Skorobogatov. Cryptographic processors—a survey. *Proceedings of the IEEE*, 94(2):357–369, 2006.

[2] Brazilian Superior Electoral Court (TSE). Election statistics, April 2010.

[3] D. Chaum. Secret-ballot receipts: True voter-verifiable elections. *IEEE Security & Privacy*, 2(1):38–47, 2004.

[4] B. Chen and R. Morris. Certifying program execution with secure processors. In *HOTOS'03: Proceedings of the 9th conference on Hot Topics in Operating Systems*, pages 23–23, Berkeley, CA, USA, 2003. USENIX Association.

[5] M. Clarkson, S. Chong, and A. Myers. Civitas: A secure voting system. 2007.

[6] V. Costan, L. F. Sarmenta, M. van Dijk, and S. Devadas. The Trusted Execution Module: Commodity General-Purpose Trusted Computing. In *CARDIS '08: Proceedings of the 8th IFIP WG 8.8/11.2 International Conference on Smart Card Research and Advanced Applications*, pages 133–148, Berlin, Heidelberg, 2008. Springer-Verlag.

[7] C. G. Diego Aranha. Relic is an efficient library for cryptography. http://code.google.com/p/relic-toolkit/, April 2010.

[8] R. Gallo, H. Kawakami, and R. Dahab. On device identity establishment and verification. In *Proc of EuroPKI'09 Sixth European Workshop on Public Key Services, Applications and Infrastructures*, September 2009.

[9] R. Gennaro, A. Lysyanskaya, T. Malkin, S. Micali, and T. Rabin. Algorithmic Tamper-Proof (ATP) Security: Theoretical Foundations for Security against Hardware Tampering, 2004.

[10] A. Huang. Keeping Secrets in Hardware: The Microsoft XBox TM Case Study. *Cryptographic Hardware and Embedded Systems-CHES 2002*, pages 355–430, 2002.

[11] International Organization for Standardization (ISO). *ISO/IEC 11889:2009 Information technology – Trusted Platform Module*. ISO/IEC, 2009.

[12] M. Joye. *Basics of Side-Channel Analysis*, pages 365–380. Cryptographic Engineering. Springer, 1 edition, 2009.

[13] Maxim Integrated Products Inc. Usip-pro component datasheet, April 2010.

[14] D. M'Raihi, M. Bellare, F. Hoornaert, D. Naccache, and O. Ranen. RFC 4226: HOTP: An HMAC-based one-time password algorithm, December 2005.

[15] D. M'Raihi, S. Machani, M. Pei, and J. Rydell. RFC draft: TOTP: Time-based one-time password algorithm, January 2009.

[16] C. Neff. A verifiable secret shuffle and its application to e-voting. In *Proceedings of the 8th ACM conference on Computer and Communications Security*, page 125. ACM, 2001.

[17] C. A. Neff. Practical high certainty intent verification for encrypted votes, October 2004.

[18] NIST. *Security requirements for cryptographic modules, Federal Information Processing Standards Publication (FIPS PUB) 140-2*, 2002.

[19] E. Oksuzoglu and D. Wallach. VoteBox Nano: A Smaller, Stronger FPGA-based Voting Machine (Short Paper). *usenix.org*, 2009.

[20] E. Rescorla. Understanding the security properties of ballot-based verification techniques. In *Electronic Voting Technology Workshop / Workshop on Trustworthy Elections*, August 2009.

[21] R. L. Rivest and J. P. Wack. On the notion of "software independence" in voting systems. *System*, 2006.

[22] P. Rohatgi. *Improved Techiniques for Side-Channel Analysis*, pages 381–406. Cryptographic Engineering. Springer, 1 edition, 2009.

[23] D. R. Sandler. *VoteBox: A tamper-evident, verifiable voting machine*. PhD thesis, Rice University, April 2009.

[24] N. K. Sastry. *Verifying security properties in electronic voting machines*. PhD thesis, University Of California, Berkeley, 2007.

[25] K. Shimizu, H. P. Hofstee, and J. S. Liberty. Cell broadband engine processor vault security architecture. *IBM J. Res. Dev.*, 51(5):521–528, 2007.

[26] G. E. Suh, C. W. O'Donnell, and S. Devadas. Aegis: A single-chip secure processor. *IEEE Design and Test of Computers*, 24(6):570–580, 2007.

[27] The Common Criteria Recognition Agreement. Common criteria for information technology security evaluation v3.1 revision 3, July 2009.

[28] USA Election Assistance Commission. Recommendations to the EAC voluntary voting system, guidelines recommendations, 2007.