

Software-Based Protection Is Moving to the Mainstream

Yuan Xiang Gu, Brecht Wyseur, and Bart Preneel

Software security is facing new challenges because traditional perimeter defenses and hardware-based protections are often too constrained and costly.

WHAT'S THE MOST important security challenge for current application systems? The fact that security is a moving target! Digital content consumed via commodity devices is penetrating every aspect of life, along with other advanced Internet-based and wireless technologies. But as the value of content and services deployed on many real-life and modern server-client delivery systems grows, so does the attraction to attackers. Modern security is facing new challenges because traditional perimeter defenses against man-in-the-middle attacks are inadequate protection against the man-at-the-end white-box attacks favored by many attackers.

Increasingly, companies rely on security technologies to protect their business model and assets, while users expect their assets to remain protected. Accordingly, security of application systems must be dynamically developed, deployed, maintained, and updated. We have no choice but to make security agile and rapidly deployable, and to employ dynamically

and flexibly renewable protection technologies. Software defenses can go far toward meeting these requirements. Like their hardware counterparts, software defenses must achieve two objectives: resist outcomes that provide the information or capabilities the attacker desires, and resist the disabling of protections. In the hardware world, such defenses include logic gates with specific protection measures to hobble information leakage and chips that self-destruct under examination. In the software world, they include compilation tools and utility libraries that operate on obscured data to resist information capture, with subtle interdependencies to produce chaotic, useless results to the attacker.

But in a rapidly changing world where digital content relies on software for its creation, storage, distribution, and consumption, we can't expect any fixed protection to provide ongoing security. This is true even when hardware is employed to harden against attacks.¹ Hence, it's becoming a fundamental requirement

that we protect digital assets by continuously upgrading the protections in their associated software. If legitimate development suffers from moving security targets, we must ensure that the black hats face moving attack targets as well.

With a sufficient amount of time at their disposal, adversaries can attack both hardware and software. Hardware protection techniques can't simultaneously be both cheap enough for commodity devices and comprehensive enough to provide strong protection; smart cards, smartphones, and other portable devices are not now, nor can they ever be, fortresses. Software has the disadvantage that it needs to fight against the learning curve: adversaries can gain direct access to the implementation, learn even more information during its life cycle, and have a large diversity of tools at their disposal. Hardware has a much higher learning curve, but it needs to aim for full, long enduring security from the start and has difficulty coping with a rapidly changing hostile reality. Today's life cycle for hardware can be very long, whereas software can be much more agile during its development and deployment phase. Therefore, even with ongoing security upgrades,

...continued on p. 58

COUNTERPOINT

Only Hardware-Assisted Protection Can Deliver Durable Secure Foundations

Jean-Daniel Aussel and Reiner Sailer

Trying to create software protections without a strong hardware foundation is like trying to build a fortress on sand.

APPLICATIONS ARE BECOMING increasingly complex to cater to the flexibility and usability requirements of our Internet-based society. Simultaneously, software is becoming the nervous system of utilities, with user mobility forcing enterprises to distribute more functionality and data onto their users' mobile devices to foster an effective and flexible work environment. Thus, the dependency of society on the correct (as expected) operation of software-driven devices and systems has increased dramatically over the past decade, and existing subversive threats against software systems originating from hackers, criminal organizations, and nation states require much stronger protection.

In hardware-based software protection, security is achieved using additional hardware, such as the Trusted Platform Module (TPM), secure coprocessors, smart cards, or a trusted processor mode. This additional hardware is generally tamper-resistant and certified, accessible only through a well-

defined protocol. Hardware protection can be considered as a computing black box, in which the protection code and data are securely hidden. Consequently, the most relevant attacks to be considered are based on the communication protocol, side channels, and physical environment.

One objective of hardware protection is to ensure a secure runtime environment for operating systems or applications. Secure boot restricts the software that can be loaded onto a system to trusted signed software. Trusted boot provides an attestation that securely identifies the layers of the software stack so its properties can be validated; a trusted environment provides the application with secure access to peripherals. This security is limited to a subset of the operating system, mostly due to the large size and dynamic nature of current operating systems. In the case of trusted boot, hardware acts as a root of trust that extends upward, with all layers measuring the next layer

and protecting the measurement before the next layer is activated. In this manner, a chain of measurements can be attested securely, and users or peer systems can at least validate the loaded software and derive its properties.

Another objective of hardware protection is to protect sensitive data, such as keys or credentials, from becoming compromised and to execute appropriately authorized software inside the hardware device. Hardware protection can guarantee those properties even if software fails or in the presence of physical attacks, as is the case of coprocessors and smart cards—for example, smart-card features are unavailable until the user performs two- or three-factor authentication. Similarly, TPMs enable the use of securely stored keys only after a user has supplied valid authentication or a system exhibits a specific software history.

In a distributed world, the ability to attest to protection capabilities is important to enable effective use by third parties. For example, cryptographic coprocessors offer outgoing authentication of the hardware, as well as of active software and its owners, to enable third parties to determine securely

...continued on p. 58



economics favors inclusion of software protection techniques over hardware-only techniques. For the former, a single build, or a single set of patches, can be transmitted electronically to benefit all connected devices, while, for the latter, every hardware enhancement adds cost to every device, and every significant upgrade requires considerable physical deployment logistics.

The emerging reality is that to protect content delivery systems effectively with software-mediated behavior, we must consider the entire security life cycle, not just initial attack resistance: our delivery systems must provide active prevention, monitoring, mitigation, and breach response for the duration of deployment. Rapid detection of and response to breaches in the field is critical. To reduce the likelihood of wide breaches, we must make systems diverse up front, and strongly varying software is much easier and cheaper than strongly varying hardware. Our

immediate response to a breach must be aimed at hobbling or preventing the next breach by the same means, often by modifying existing systems in unpredictable ways to foil an automated attack. Longer-term response includes both ongoing renewal of diversity and device individualization as a proactive defense, thereby minimizing how many devices a particular breach affects and shoring up any new weak spots.

The future security needs of commodity content delivery devices are best served if software and hardware security experts work together much more than is their habit today. Some facilities are best provided in hardware (device-specific behaviors that are hard to shut down or subvert), others are best provided by hardware/software co-design. Further research is needed to determine how our pursuit of security can best combine the agility and electronic communicability of software with the ever-growing throughput and reliable device

individualization that targeted hardware can provide. Currently, hardware and software protection techniques aren't well integrated. How much can we improve content security when hardware and software designers work together? We won't know until we try.

Acknowledgments

We thank Harold Johnson and Michael Wiener for helpful discussion, review, and comments.

Reference

1. J. Halderman et al., "Lest We Remember: Cold-Boot Attacks on Encryption Keys," *Comm. ACM*, vol. 52, no. 5, 2009, pp. 91–98.




YUAN XIANG GU is a chief architect at and cofounder of Cloakware of Irdeto, where he's responsible for technology development and evolution. Prior to joining Cloakware, he worked as a senior scientist and architect at Nortel Networks, a visiting professor at McGill University's School of Computer Science, Canada, and a professor in the Department of Computer Sci-



the overall system's protection properties. TPMs use platform certificates to attest to the platform environment that a third party will rely on when using the TPM. Hardware devices execute trusted code either preloaded or loaded dynamically after issuance, using secure protocols and shared secrets on update servers. The strong hardware integrity and confidentiality protection removes physical disclosure and manipulation threats that exist on software-only protection systems.

Ultimately, hardware protection can provide a certified, tamper-resistant, secure environment to ensure that only secure code is executed, portions of the system software components haven't been modified, sensitive data is protected, access to peripherals is restricted, and users are authen-

ticated. This also provides a means of securely updating the hardware device's software or credentials. All these features can be used independently, as well as provide a secure foundation for software-protection techniques. In today's open environments, where attackers prey on every line of code and commercial off-the-shelf systems include a plenitude of known and unpatched software vulnerabilities,¹ only hardware can create durable security properties upon which software can extend meaningful protection properties. While there are known attacks against some hardware mechanisms and CPUs, their number pales in light of thousands of software vulnerabilities. Additionally, without a strong hardware security foundation, any software security mecha-

nism exposes a "soft underbelly" and is vulnerable to attacks from below.² Until software becomes more reliable and less vulnerable, software should focus on correctness and leave strong, durable protection to hardware mechanisms. 

Acknowledgments

We thank Kenneth Goldman for helpful reviews and comments.

References

1. IBM Security Solutions, "X-Force 2009 Trend and Risk Report, Annual Review of 2009," IBM, Feb. 2010.
2. J. Rutkowska, "Subverting Vista Kernel for Fun and Profit," SyScan'06 & Black Hat Briefings 2006; www.blackhat.com/presentations/bh-usa-06/BH-US-06-Rutkowska.pdf.

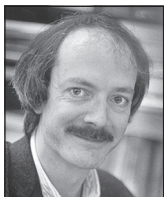
GU, WYSEUR, AND PRENEEL RESPOND

ence of Northwest University of China. Contact him at yuan.gu@irdeto.com.



BRECHT WYSEUR is a cryptography expert at Nagravision S.A., Switzerland, where he works on software security aspects for conditional access systems and digital rights management. His

main research interests are in the area of cryptography, with a particular focus on white-box cryptography. Wyseur has a PhD in electrical engineering from K.U.Leuven, Belgium. Contact him at brecht.wyseur@nagra.com.



BART PRENEEL is a full professor in the COSIC research group at K.U.Leuven, Belgium. His research interests include applied cryptography, system security, and privacy. Preneel has a PhD in electrical engineering

from K.U.Leuven, Belgium; he has been a visiting professor at five universities in Europe and was a research fellow at the University of California, Berkeley. Contact him at bart.preneel@esat.kuleuven.be.

The hardware counterpoint specifies some valuable hardware-assisted protection features that can be exploited to establish a trusted code base. However, in contrast to what Aussen and Sailer claim, we strongly believe that hardware-assisted protection techniques (which are, in fact, only optional, not mandatory, components installed in consumer devices because of cost limitation, availability, and usability) aren't a necessity to obtain a trusted code base on an untrustworthy computing platform.

Gaining trust in the execution of a software application is a matter of establishing a trust chain from a "core of trust" to the application. Instead of relying on a local hardware component, attackers can exploit a network's availability to achieve a trust chain from a remote core of trust (a server) to the application, based solely on pure software protection techniques. The combination of several layers of protection techniques eventually results in an implementation that's too complex and dynamic for analysis.

Aussen and Sailer also state that software is susceptible to vulnerabilities. First, it's a well-known mistake that hardware-assisted protection can replace software-based protection. In a white-box attack environment, one of the biggest and most fundamental vulnerabilities is to execute application software without any self-protection against direct attacks to program flows and assets. Software-based protection is thus essential to ensure end-to-end system security if hardware-assisted protection techniques are used. Moreover, when vulnerabilities are discovered in hardware, they can have a much more devastating effect and can be hard and costly to repair. We believe that the development of modern protection techniques, in combination with a hardware/software co-design, can favor the balance between performance and security.

AUSSEL AND SAILER RESPOND



JEAN-DANIEL

AUSSEL is Tools and Innovation R&D Manager at Gemalto. His research interests include operating system, network, and application security using smart cards. Aussen has a PhD in electrical

engineering from INSA, France. Contact him at jean-daniel.aussen@gemalto.com.



REINER SAILER is a research staff member and manager of the Security Services Team at the IBM Thomas J. Watson Research Center, with a strong focus on cybersecurity. His research

interests include trusted computing, secure runtime environments, and computer forensics. Sailer has a PhD in electronic engineering (Dr.-Ing.) from the University of Stuttgart, Germany. Contact him at sailer@us.ibm.com.

The main claim made in favor of software protection is that software is easier to update than hardware, and consequently that we expect better base protection from obfuscated, diversified, and constantly updated software. However, hardware protection devices aren't immutable—for example, hundreds of millions of smart cards host Java virtual machines and can be securely updated in the field; even cryptographic coprocessors are field-upgradeable. Tipping the scales toward hardware protection, it's estimated that patches exist for only about half of the thousands of known vulnerabilities in popular software. Fast-moving vulnerable software targets easily become a nightmare for the defender.

The key advantage of hardware protection is that the protected code and data, such as keys and algorithms, can't be observed directly but only eventually derived from observable hardware characteristics. Promising efforts focus on minimizing or avoiding such side channels. On the other hand, attackers can analyze statically or dynamically the host software protection to learn protected data and algorithms. The Cold Boot attack doesn't prove a TPM weakness, but rather confirms that keys aren't safe if exposed to software in main memory.

Attackers tend to focus on the weak spots, one of them being host software that can be manipulated to disable or bypass hardware protection. We agree with our software protection colleagues that this promises to be an interesting area of hardware/software co-design, where hardware safely protects data and code, and host software ensures that the host safely uses its platform's hardware protection mechanisms.