Hardware-centric Solutions for Security in Embedded Systems

Alexander J. Rich, Colorado State University

***Abstract*—Embedded Systems are becoming ubiquitous in today's environment. This paper describes how security, in the realm of Embedded Systems, is playing an ever increasing and important role. I focus on why security in Embedded Systems is especially important, what is currently being done to use Hardware as a form of security and what future efforts to provide and improve security might look like.**

I. Introduction

Embedded Systems are becoming one of the most seen electronics device. Currently it is a $10 Billion industry, taking on many different tasks in several different facets of the economy [1]. They are most commonly used in industrial controls, consumer electronics and communications and networking. As the miniaturization of electronics continues Embedded Systems are becoming a huge component in day-to-day life. Furthermore, as the Internet of Things (IoT) sweeps our devices Embedded Systems are becoming more and more connected to us and hence all of our devices.

As result of increased productivity and increased connectivity security is becoming more important and crucial than ever. Security is critical to all computing but specifically to Embedded Systems, and because of the nature of these systems there is a lot of thought that must go into security design, very different to that of desktop and server design. Probably one of the most famous attacks, known as Stuxnet [2], which infiltrated several SCADA systems that was capable of stealing data and even creating control commands to the control systems (found even in one Iranian nuclear power plant). Another well-known attack infected numerous POS devices in several Target stores located in the United States, exposing millions to potential fraud [3]. Hackers have even been able to gain access to previous printing jobs found in the memory of discarded printing systems [4]. Security threats are always present and Hackers are always going to be creative. Although nothing is 100% secure, steps can be taken to ensure certain levels of security. The science behind security is all about balancing cost, performance, power, reliability and area (practicality). A good system will take into account whom the possible attackers are (Class I, Class II, Class III), what information is at stake and why it needs to be protected. With these core principles in mind systems can be designed to offer the best possible combination of all variables.

Embedded System security is different than traditional security designs. The nature of Embedded Systems means attackers have constant physical access to your device which in itself presents many challenges, methods such as Ion Beams, X-Ray's, acids etc. can all be used to break down, probe and reverse engineer almost any device (with the right resources). This also means attackers can manipulate and replace any components on the device. Another consideration is the fact that products, which have been deployed, could be in operation for 10-15 years depending on the service. Not only does this create software maintenance concerns but gives attackers essentially unlimited amount of time to find and exploit any weaknesses within the system.

There are a couple different types of attacks, Non-Invasive and Invasive. Within these are passive and active attacks. Non-Invasive passive attacks breach side-channel pathways whereas active attacks focus on fault-attacks such as stack overflows. Invasive passive attacks are categorized as probing, where reverse engineering is usually the end goal. Invasive active attacks mean physical tampering of the device which can accomplish a number of different things, for example current source power analysis can actually be used to extract information about what kind of operations are being run and where they are being stored on the device.

Non-Invasive attacks are usually aimed at what connects the device to the outside world. Ports like USB, JTAG, RS232, and Ethernet are susceptible to attacks and are very common across all platforms. Intentionally malformed packets can be sent to cause a fault (when firmware does not handle it correctly). Fault injection attacks, which make hardware fail, can come in many different forms. A few popular ways include changing/lowering the voltage to slow signals and miss deadlines, increasing temperature to affect signal propagation delay, overclocking to shorten the allowed time for traversing the logic cloud and even introducing natural particles (e.g. alpha-particles) to create a change of internal signal values.

Invasive attacks, which are particularly difficult to defend against in Embedded Systems, also come in many different shapes and sizes. Chip removal and optical microscope probing can reveal the contents of the ROM. Probes on bus lines can actually figure out what the data is as well as crack crypto algorithms and keys.

Another major concern is supply chain vulnerabilities. Both

IP's and IC's are designed and fabricated all over the world, which increases the probability as well as ease for Hardware Trojans. Hardware Trojans are either intentional or unintentional modifications to existing circuit elements, which can change the functionality and lead to information being leaked as well as an unreliable product. Reports of counterfeit parts from 2009 to 2011 alone more than quadrupled [5].

Embedded Systems engineers, now more than ever, will need to appreciate all these different types of attacks and considerations to help inform their design and create a device that will meet the needs of the consumer while also protecting the consumer.

II. Security approaches today

Hardware Security Module's (HSM) are hardware based stand-alone subsystems, which are developed mostly for cryptographic key management and processing however the concept behind these designs is based in Hardware Root-of-Trust (RoT). This term essentially implies hardware-based system security capabilities as opposed to a software-centric approach to security.

Some of the services RoT systems can provide include keeping and monitoring an integral platform making sure the system is running as intended. Things like disk encryption help ensure data is being kept safe. Other On-Chip securities include hardware Sandboxing which only allow certain programs to access critical areas rejecting all other calls. This is very similar to the ARM TrustZone which functions in a normal zone and a trust zone [6]. The trust zone is available to pre-authorized applications only, which means less exposure of critical information. This idea of RoT and sandboxing is what I will focus on, and the different approaches to accomplishing this.

A. Architectural Requirements for Supporting Sandboxes

Malicious software in today's world can detect being executed in a virtual machine and then actually change its behavior. When virtualization is moved to the hardware level, however, the malicious software can no longer detect the execution and can hence be monitored in a safe way [7]. One promising technique published by Marcel Eckert at Helmut Schmidt University focused on two ideas, virtualization and FPGAs and dedicated hardware for guest virtual machines.

Due to the adaptability of FPGAs, virtualization in tandem can be used to construct a system virtual machine environment. A process virtual machine is capable of supporting individual processes whereas system virtual machines provide a complete system environment that can actually accommodate the whole OS and processes. Normally, cores in a multiprocessor are used to give each process virtual machine their own core and hence their own security domains to separate the virtual machines via software but in Eckert's white paper they enforce virtual machine separation by additional hardware, assign devices to a virtual machine physically and furthermore allow the execution of guest operating systems (not just processes).

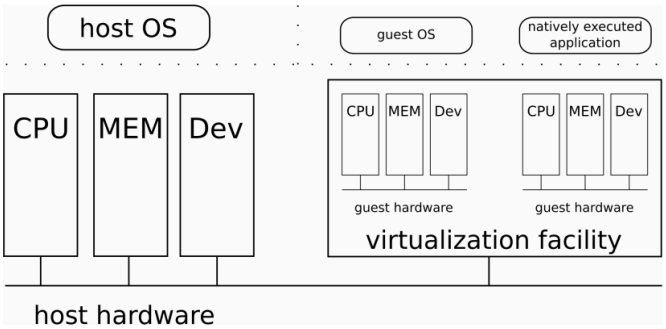

Fig 1. Overall Architecture of a hardware supported virtualization/sandboxing system [7]

Figure 1 demonstrates the overall hardware supported virtualization/sandboxing system and even extends conventional computer architecture by a virtualization facility. Here reconfigurable logic is used to instantiate the entire system (which can be seen as a guest system in terms of system virtual machine concepts). Dedicated sources of the overall system are managed by the host OS, similar to the conventional virtual machine managers.

The major advantages in this system is utilizing the adaptable hardware in FPGAs which allows a system virtual machine environment to execute guest machines on their own physical hardware (instantiated within the FPGA). This translates to stealth malware being unable to detect execution preventing them from getting out of their sandbox.

B. ARM TrustZone and Private Execution Environments

Currently ARM TrustZone is used to provide a Trusted Execution Environment (TEE) (mainly for mobile devices). Use of TrustZone is actually somewhat limited for the very reason it is so helpful, it only is available to a few pre-authorized applications, applications that are pre-approved by the TrustZone OS vendors. To overcome this and extend the usefulness of TrustZone Jinsoo Jang and his team at KAIST developed a PrivateZone framework to enable individual developers to utilize TrustZone resources. PrivateZone developers can execute Security Critical Logics in Private Execution Environments (PrEE).

PrivateZone enables a private execution environment by utilizing ARM virtualization extensions. During a secure boot PrivateZone will generate a two stage-2 page table for the PrEE and the Rich Execution Environment (REE) (REE hosts general OSes such as Linux). The entries for each page table are completed in advance during boot time to create three logical environments REE, Trusted Execution Environment (TEE) and PrEE.

PrivateZone guarantees isolation via Inter-Environments, where stage-2 page tables statically separate REE and PrEE. Due to features like this PrivateZone provides the benefits of the isolated execution to general applications, in contrast the access to TrustZone technologies is currently limited to the pre-authorized ones allowed by the TrustZone vendors. PrivateZone allows for applications to run SCLs securely without crowding the TEE.

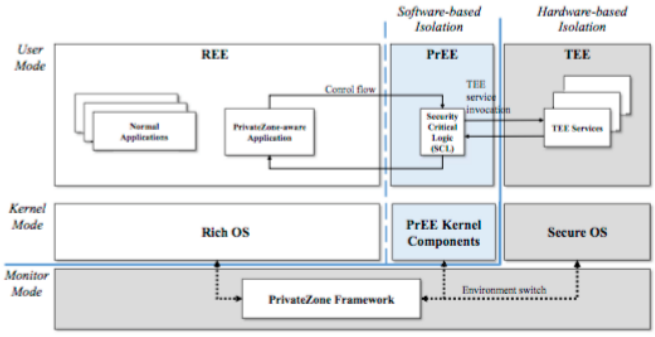

Fig 2. Additional Executional Environment [7]

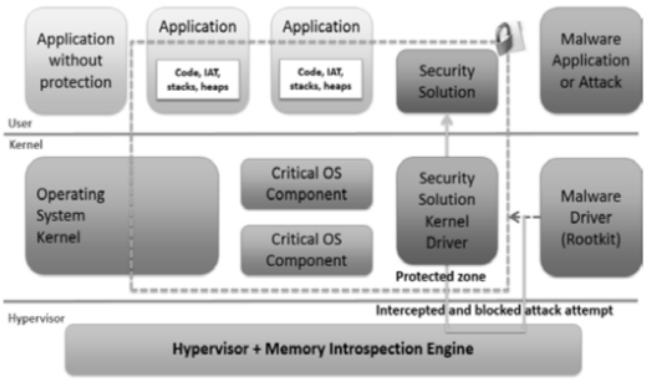

FIG 3. [8]

Figure 2 depicts the PrivateZone interface between the existing TEE and REE environments. With the addition of the PrEE to securely execute SCLs software based isolation, in conjunction with hardware-based isolation allow increased RoT integration. The PrEE is created by leveraging the ARM virtualization and security extensions, the main framework of PrivateZone is in monitor mode to allow managing of switches between different environments.

Due to the fact TrustZone is so widely used I felt the need to include PrivateZone, which would bolster TrustZone, provide more functionality and have the potential to have the same affect TrustZone did on security for Embedded Systems.

C. *High-level approach*

Hypervisors (or Virtual Machine Monitor) is a piece of hardware (can be software too), which creates and runs virtual machines. Hypervisors are interesting because they offer the guest OS with a virtual operating platform and manages the execution of the guest OS systems [8].

This next concept Sandor Lukacs and his team from Romania Technical University took a high-level approach to hardware virtualization based security solutions. Hardware-enforced security solutions are especially suitable for Embedded Systems; their solution involves a thin layer bare-metal hypervisor, a memory introspection engine and is validated on microarchitecture based on Intel x86 processors. This was specifically targeted at providing more security to POS and industrial embedded devices, which is much needed.

A specially tailored hypervisor that incorporates the needed security features with a reduced footprint integrates a Memory Introspection Engine (MIE), which analyses, identifies and protects the running guest operating system. The MIE provides two main functions or types of protections, kernel integrity protection and user mode processes protection. Kernel integrity protection includes the protection of images of the kernel and critical drivers against advanced kernel mode malware. The user mode processes protection involves protecting and enforcing various preprocess security policies.

Figure 3 depicts the hardware virtualization and MIE engine based security solution. Figure 3 shows how the hypervisor is implemented. One special property of the MIE is that it is dynamically adaptable by updating its security policies, and avoids relying on any in-guest agent (coming back to the idea of RoT). The MIE can be easily and moreover independently updated from the in-guest operating system or any in-guest security solution.

The Hypervisor in conjunction with the MIE was able to demonstrate protection mechanisms not only for kernel mode structures but also user-mode processes which inherently better protects the integrity of various applications against many kinds of attacks.

This is especially important in Embedded Systems because attacks can and do come in all shapes and sizes. Protecting from high-level approaches, as well as low, is important and hence worthwhile to consider each approach.

III. THE FUTURE OF SECURITY IN EMBEDDED SYSTEMS

All things considered the future of security in Embedded Systems will likely have to employ several different techniques to best protect against attackers.

Most likely hardware security will work in conjunction with software security as both need to work together for success a otherwise one can negate the other.

Fig 4. Trust pyramid [8]

Figure 4 shows a Trust Pyramid, any one flaw in each layer means the others can be compromised so it is important to have a comprehensive security design that covers all bases otherwise efforts to design one single layer can be futile.

Typical Hardware Security Modules are compromised of secure memory, secure cryptography, secure functions,

interface and control and tamper-protection. Securing memory means have little non-volatile data storage inside the HSM (RoT area) to prevent unauthorized readout, manipulation or deletion of critical information (like cryptographic keys). Securing cryptography algorithms means keeping key generation and key verification within the RoT area. Securing functions comprises all shielded functions within the HSM (such as a protected clock signal). Tamper-protection, which is particularly relevant meaning all blocks of hardware security, needs to be enclosed by a continuous physical boundary (denying data interception). The future of security in Embedded Systems will employ all of these techniques.

IV. Conclusion

Hardware security modules, and hardware-centric solutions to Embedded Systems are becoming increasingly more important. The challenges presented by the nature of Embedded Systems is especially difficult due to accessibility, creative solutions will become necessary to help ensure the protection of and integrity of devices.

There is no one solution to the security issue, instead the best approach is a case by case evaluation of each layer of security, from organizational to software security all working together to create the most secure system problem.

Things to consider when designing security for Embedded Systems come down to the security science aforementioned. Balancing cost, performance, power, reliability, area and time. Knowing who your enemy is or what demographic they most likely fall into is a huge step in being able to classify the types of attacks one can expect to see and hence preemptively safeguard against.

References

[1] J. Jackson and I. Service, "IDC: Embedded Systems Market to Double by 2015", *PCWorld*, 2016. [Online]. Available: http://www.pcworld.com/article/239798/idc_embedded_systems_market_to_double_by_2015.html. [Accessed: 15- Dec- 2016].

[2] R. Langner, "To Kill a Centrifuge: A Technical Analysis of What Stuxnet's Creators Tried to Achieve," The Langner Group, 2013.

[3] B. Krebs, "A First Look at the Target Intrusion, Malware — Krebs on Security," 15-Jan-2014. Available: http://krebsonsecurity.com/2014/01/a-first-look-at-the-target-intrusion-malware/.

[4] A. Ki-Chan and H. Dong-Joo, "Embedded devices, and Antivirus-free safe jideout for malware," presented at the Defcon, 2010.

[5] 2016. [Online]. Available: http://www.erai.com/presentations/General%20Session%202/An%20In-Depth%20Look%20at%20Counterfeits%20from%20a%20Statistical%20Perspective-%20ERAI%20and%20IHS.pdf. [Accessed: 15- Dec- 2016].

[6] "Architectural requirements for constructing hardware supported sandboxes - IEEE Xplore Document", *Ieeexplore.ieee.org*, 2016. [Online]. Available: http://ieeexplore.ieee.org/document/7518532/. [Accessed: 15- Dec- 2016].

[7] "PrivateZone: Providing a Private Execution Environment using ARM TrustZone - IEEE Xplore Document", *Ieeexplore.ieee.org*, 2016. [Online]. Available: http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=7723848. [Accessed: 15- Dec- 2016].

[8] "Hardware virtualization based security solution for embedded systems - IEEE Xplore Document", *Ieeexplore.ieee.org*, 2016. [Online]. Available: http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=6857879&tag=1. [Accessed: 15- Dec- 2016].

[9] "Assessment of Hypervisor Vulnerabilities - IEEE Xplore Document", *Ieeexplore.ieee.org*, 2016. [Online]. Available: http://ieeexplore.ieee.org/document/7600180/. [Accessed: 15- Dec- 2016].

[10] "A hardware security solution against scan-based attacks - IEEE Xplore Document", *Ieeexplore.ieee.org*, 2016. [Online]. Available: http://ieeexplore.ieee.org/document/7538894/citations. [Accessed: 15- Dec- 2016].

[11] "Hardware Security Modules for Protecting Embedded Systems", *escrypt*, 2016. [Online]. Available: https://www.escrypt.com/fileadmin/escrypt/pdf/WP-Embedded-HSM.pdf. [Accessed: 15- Dec- 2016].

[12] "Hardware-Assisted Detection of Malicious Software in Embedded Systems - IEEE Xplore Document", *Ieeexplore.ieee.org*, 2016. [Online]. Available: http://ieeexplore.ieee.org/document/6301679/?part=1. [Accessed: 15- Dec- 2016].

[13] "Embedded Security Solution for Digital Safe-Guard Ecosystems - IEEE Xplore Document", *Ieeexplore.ieee.org*, 2016. [Online]. Available: http://ieeexplore.ieee.org/document/4233767/?reload=true. [Accessed: 15- Dec- 2016].

[14] "Practical Secure Hardware Design for Embedded Systems Abstract | TechOnline", *Techonline.com*, 2016. [Online]. Available: http://www.techonline.com/electrical-engineers/education-training/tech-papers/4125495/Practical-Secure-Hardware-Design-for-Embedded-Systems. [Accessed: 15- Dec- 2016].

[15] "Building Trust into Embedded Devices Abstract | TechOnline", *Techonline.com*, 2016. [Online]. Available: http://www.techonline.com/electrical-engineers/education-training/tech-papers/4126211/Building-Trust-into-Embedded-Devices. [Accessed: 15- Dec- 2016].

[16] "University Research in Hardware Security", 2016. [Online]. Available: http://www.hotchips.org/wp-content/uploads/hc_archives/hc26/HC26-10-tutorial-epub/HC26.10-tutorial1-HW-Security-epub/HC26.10.155-6_Lee_UniversityResearch_go.pdf. [Accessed: 15- Dec- 2016].

[17] "Addressing Hardware Security Challenges in Internet of Things: Recent Trends and Possible Solutions - IEEE Xplore Document", *Ieeexplore.ieee.org*, 2016. [Online]. Available: http://ieeexplore.ieee.org/document/7518284/. [Accessed: 15- Dec- 2016].

[18] "A Study on the Hardware-Based Security Solutions for Smart Devices - IEEE Xplore Document", *Ieeexplore.ieee.org*, 2016. [Online]. Available: http://ieeexplore.ieee.org/document/7424208/. [Accessed: 15- Dec- 2016].

[19] "SmashGuard: A Hardware Solution to Prevent Security Attacks on the Function Return Address - IEEE Xplore Document", *Ieeexplore.ieee.org*, 2016. [Online]. Available: http://ieeexplore.ieee.org/document/1683758/. [Accessed: 15- Dec- 2016].

[20] "Hardware-Based Malware Detection Using Low-Level Architectural Features - IEEE Xplore Document", *Ieeexplore.ieee.org*, 2016. [Online]. Available: http://ieeexplore.ieee.org/document/7430287/. [Accessed: 15- Dec- 2016].

[21] "A Designer's Rationale for Nanoelectronic Hardware Security Primitives - IEEE Xplore Document", *Ieeexplore.ieee.org*, 2016. [Online]. Available: http://ieeexplore.ieee.org/document/7560196/. [Accessed: 15- Dec- 2016].

[22] "Hardware security in practice: Challenges and opportunities - IEEE Xplore Document", *Ieeexplore.ieee.org*, 2016. [Online]. Available: http://ieeexplore.ieee.org/document/5955003/. [Accessed: 15- Dec- 2016].