Integrated Security for Embedded IoT Systems

Paul Fortier, Patrick DaSilva, Benjamin Viall
Electrical and Computer Engineering Department
University of Massachusetts Dartmouth
North Dartmouth, Massachusetts USA
email: pfortier@umassd.edu, pdasilva@umassd.edu, u_bviall@umassd.edu

Abstract— **Embedded systems within the evolving Internet of Things (IoT) space are becoming ubiquitous. The problem lies in their weak to non-existing security. Embedded IoT systems can be built as Systems on a Chip (SoC) using IP-cores and Field Programmable Gate Array (FPGA) technology, as components within a Printed Circuit Board (PCB) or as systems of systems. In each case there are different levels of design as well as security vulnerabilities and therefore solutions. This paper looks into hardware, firmware and software security issues as well as techniques to improve an embedded systems overall security for a subsurface roadway IoT sensing system.**

Keywords-Embedded systems; security; IP-Cores; SoC.

I. INTRODUCTION

Embedded computer systems are found in just about every object engaged in smart and connected cities activities. Embedded systems typically operate within real-time processing constraints and must process input data in a timely manner to drive output data or control physical actions. Embedded systems are designed to operate typically without human interaction. In some instances embedded systems may respond to human inputs to steer the actions of one or more deeply embedded systems. When designing embedded systems, the typical mode of operation, is to determine what physical parameters are needed to meet systems goals and design elements to support these requirements.

This paper provides a descriptive overview of an architecture and design for a wireless underground smart sensor system, data collection and Internet of Things (IoT) transmission system. The subsurface sensing system under test at the University of Massachusetts is presently being modified to investigate and include security management through all levels of the architecture where deemed necessary. The paper will examine issues encountered during the development of hardware security elements.

Section II defines the application used to motivate security needs for embedded systems. Section III introduces security concepts and how they apply to embedded systems. Section IV describes the proposed conceptual solution both hardware and software. Followed by Section V where the papers conclusions are drawn.

II. ROADWAY SENSING SYSTEM APPLICATION

Several million miles of secondary paved and unpaved roads in the United States lie in seasonal frost areas and are highly susceptible to damage during the winter freeze and spring thaw periods. To understand roadway structural conditions during these cyclic periods requires knowledge concerning subsurface temperature and moisture. Though, acquiring roadway subsurface information in real-time is costly, difficult and in some cases, impossible given currently available technologies.

Roadway management policies such as seasonal load restriction (SLR), limits loads of heavy trucks during the spring thaw period. Roadway restrictions may cause trucks to take costly detours requiring additional driving time and lighter loads resulting in more trips. The challenge is to protect the transportation infrastructure and minimize roadway maintenance costs, but also to allow commerce to flow as unrestricted as possible during spring thaw and roadway strength recovery periods.

Present methods for imposing SLR's are not real-time nor data driven. Most rely on using cumulative thawing index [1] based on computed degree day measurement. Additional studies [2]-[4] looked to reduce the time SLR is in place using manual collection methods and models. State Department of Transportation (DOT's) have performed studies looking at methods to remove SLR's in a timely fashion [5]-[7] using measured data.

One system design consideration not addressed in the initial study was the issue of sensor site and communications node security. To address those deficiencies, a redesign of the sensor node and communications node to include hardware based mechanisms for detection, response and recovery from malicious attacks has been initiated.

A. Roadway Sensing System Architecture

Researchers [9]-[11] examined using wireless sensors to monitor subsurface environmental conditions. The University of Massachusetts Dartmouth (UMD) SLR forecasting tool builds upon this research using real-time data feeds from wireless sensors embedded in managed roadways using semi-automated techniques [12]-[14]. The hardware for the UMD SLR system consists of embedded subsurface roadway sensors and IoT communications nodes linked with a backend Decision Support System (DSS) (Fig. 1).

Copyright (c) IARIA, 2018. ISBN: 978-1-61208-659-0

Wireless sensors are embedded into roadways to a depth of up to nine feet. Sensors are recharged using a multi-source recharging subsystem. Collected data is packaged and transmitted to the UMD decision support system (Fig. 2).

The UMD SLR DSS system consists of data extraction, fusion, visualization and infrastructure forecasting tools. Data extraction tools retrieve data feeds, translate and package raw data for decision support tool use. Extracted data include site specific weather data, embedded sensor data and site soil composition data.

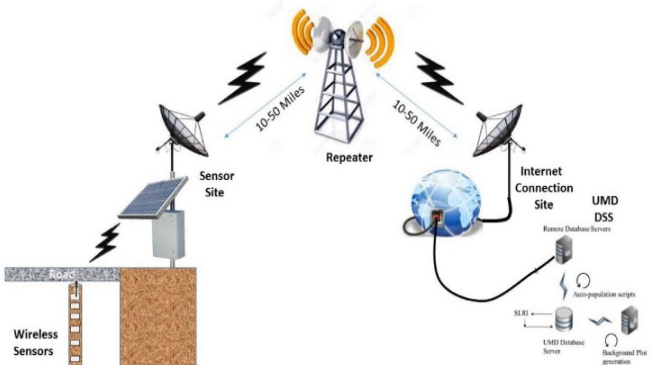

Figure 1. UMD SLR Systems Architecture

B. *Subsurface Roadway Sensor Design*

The wireless sensor nodes (Fig. 2) are constructed using three custom Printed Circuit Boards (PCB) with custom Field Programmable Gate Arrays (FPGA) supporting required data collection, processing and transmission. One PCB is used to read and manage sensor data access, a second to manage power generation and a third to manage system interactions.

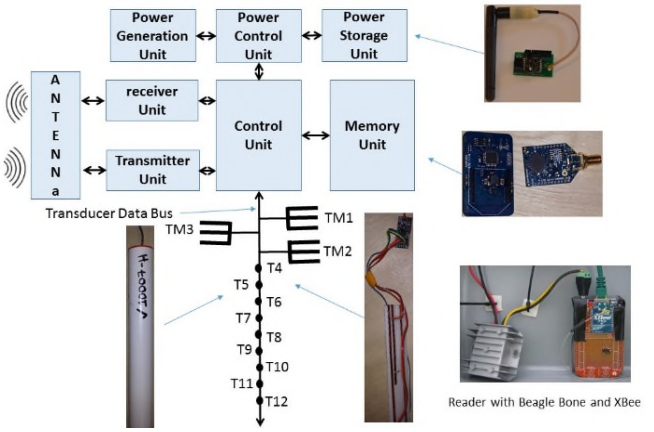

Figure 2. Wireless Smart Sensor Architecture

The sensor communications node operates at 900 MHz with a data rate of 200 kbps. The communications component is connected to an embedded FPGA based controller responsible for interacting with the sensors to extract measurements, convert information and package for transmission to the external reader on demand. The last component is the power management board using power harvesting elements to maintain battery power. Supporting the sensor system is a roadside solar battery powered reader. The reader uses a commercial radio frequency communications module and a Linux processor to implement IoT connectivity to the DSS (Fig. 2).

III. EMBEDDED SYSTEMS SECURITY ISSUES

Security has been talked about for quite some time as more IoT devices spread throughout the ecosystem of the smart and connected cities concept. Even prior to the notion of inter-connection of IoT devices for anywhere / any time access, security was a concern from system developers, though not often for the single IoT sensor developed by small vendors. As IoT devices become ubiquitous, security flaws have begun to be exposed and now thought of as being important to consider. To examine what a security flaw looks like as well as the reasons for an attack an existing security taxonomy [15] was refined to focus embedded systems vulnerabilities.

The taxonomy defines what a security incident is, what the attack was and how the attack was carried out. The taxonomy defines, the attacker, tools used, form of access to the target, what vulnerability was used, what action was taken by the attacker, what the target of the attack was, the objectives and result of the attack, and the actual harm inflicted. The taxonomy has helped in focusing understanding of vulnerabilities and to work towards developing solutions.

A. *Security Flaws in Embedded Systems*

Embedded systems typically interact with some physical system to provide a strict timing response to stimuli. Failure to adhere to timing constraints effects real-time response, performance, and ultimately the safety and security of the embedded application. Typical characteristics of embedded systems such as; limited core processing power, limited available power, physical exposure, remoteness, unmanned operation, and network connectivity represent possible limitations as well as cybersecurity weaknesses. Embedded systems limited resources imply they lack excess capacity to support security services operations adequately [16].

Limited resources within embedded systems also provide the attacker with areas to exploit. For example, many embedded systems utilize FPGA open cores supporting local processing, input, output and storage, which may harbor malicious elements or unobstructed open entry points (Fig. 4). Embedded systems limited resources make them vulnerable to denial of service attacks, power depletion attacks, code reuse attacks and memory hacking attacks to name a few. Limited operating system services, tightly timed applications task segments as well as limited controls on memory access, inputs and outputs may also provide the attacker numerous targets for their interference (Fig. 3). One such attack, referred to as a code reuse attack, causes control flow changes by reusing existing instructions for malicious purposes. Typically code reuse attacks modify non-executable memory by overwriting stacks, function pointers, or set jump buffers forcing the processor to execute instructions in unintended sequences or into regions not

Copyright (c) IARIA, 2018. ISBN: 978-1-61208-659-0

tested during normal system verification and validation. Numerous authors have written about the need for hardware based solutions to embedded System on a Chip (SoC) security vulnerabilities [17].

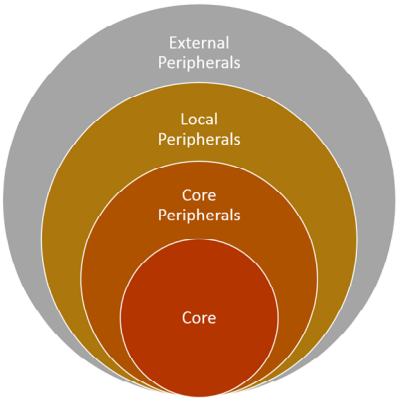

Figure 3. Onion model for system on a chip IoT devices

De Clercq and Verbauwhede published a survey of 21 hardware-based Control Flow Integrity (CFI) architectures [17] that describes the need for hardware-based over software-based CFI solutions. Software-based CFI solutions rely on inserting code into a program to perform CFI checks on indirect branches. When the compiler is unaware of the security aspects concerning the CFI checks, it might cause the optimization step to spill registers holding sensitive CFI data to the stack [17]. The attacker model expects the attacker to always have control of data memory and hence the stack allowing the attacker to circumvent the CFI protections. Hardware-based access control mechanisms can provide strong isolation for runtime data structures and metadata [17]. Hardware-based architectures can protect against attackers that control both code and data memory.

The analysis of the security policies used included a detailed comparison of the policies with respect to their security, limitations, hardware cost, performance, and practicality. In general, the use of a Shadow Call Stack (SCS) is particularly important to protecting the backward edge of control flow against Return-Oriented Programming (ROP). De Clercq and Verbauwhede concluded the forward edge of the control flow still face practical limitations preventing widespread adoption. More than half require software components placed inside the binary causing reduced responsiveness to real time tasking, opening them up to code injection attacks if non-executable memory protections are not enforced. Of the 21 CFI architectures, 75% of them focused on x86, ARM, or SPARC Instruction Set Architecture (ISR) as the target and one focused on the AVR ISR [18]. Of the 17 CFI architectures that contained a forward edge static policy, branch regulation and table policies met the most protection requirements. However, both branch regulation and table policies either require complex control flow graphs or software placed inside the binary.

B. Security Solutions for Embedded systems

A Framework for Improving Critical Infrastructure Cybersecurity [15] version 1.1 released in April 2018 is the outcome of the National Institute of Standards and Technology (NIST) collaborating with private and government entities to provide cybersecurity risk frameworks for voluntary use by critical infrastructure owners and operators. The Framework offers a way to address cybersecurity's effect on physical, cyber, and people. The framework is applicable to organizations relying on technology such as industrial control systems (ICS), Cyber-Physical Systems (CPS), and the Internet of Things (IoT). The core of the Framework consists of five functions or basic cybersecurity activities – Identify, Protect, Detect, Respond, and Recover.

- Identify – Develop an organizational understanding to manage cybersecurity risk to systems, people, assets, data, and capabilities.
- Protect – Develop and implement appropriate safeguards to ensure delivery of critical services.
- Detect – Develop and implement appropriate activities to identify the occurrence of a cybersecurity event.
- Respond – Develop and implement appropriate activities to take action regarding a detected cybersecurity incident.
- Recover – Develop and implement appropriate activities to maintain plans for resilience and to restore any capabilities or services that were impaired due to a cybersecurity incident.

Under the Framework, the proposed subsurface roadway sensor system's solution attempts to provide Protect, Detect, Respond, and Recover mechanisms to support an embedded system under cyber-attack while maintaining system integrity and mission essential functionality. The proposed solution seeks to protect an embedded processor by detecting unexpected control flow changes using hardware-based control flow integrity techniques integrated into the Security Unit (SU). The Security Memory (SM) contains a merged file containing meta-data from an off-line control flow analysis merged with the FPGA bit stream to aid in recovery. The response to a malicious detection is to isolate the malicious control flow changes by redirecting them to a SACrificial processor (SAC) to prevent and avoid further infection of the embedded system (Fig. 5). The SAC also has a SAC Data Memory (SACDM) to isolate Data Memory (DM) from malicious alterations. Actions of the SAC are recorded in the data output unit (DOU) for post run analysis.

The conceptual design strives to detect attacks, respond to attacks in a way that prevents future or continuous successful attacks identified through real-time instruction analysis and data storage analysis techniques under development within this project.

Copyright (c) IARIA, 2018. ISBN: 978-1-61208-659-0

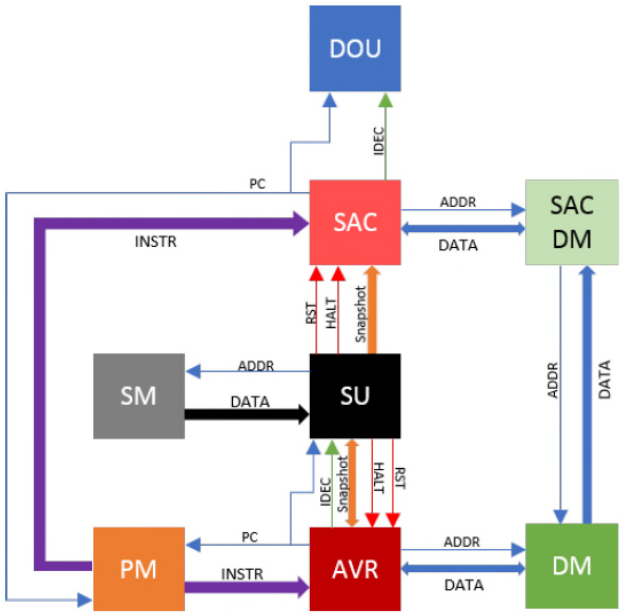

Figure 4. Conceptual Embedded SoC secure architecture

IV. SLR ROADWAY IOT SECURITY SOLUTION METHODS

As depicted in Fig. 4 above, our hardware (FPGA) based solution uses a variety of added hardware elements to monitor activity, detect erroneous activity, respond to the malicious activity and recover embedded systems normal operations. A hardware element, the Security Unit (SU) monitors instruction formats, jump and branching addresses using a novel associative search engine to identify rogue instructions not found in valid code blocks as well as invalid jump and branch addresses injected by malicious means.

One such attack is a Control Flow Attack (CFA). CFA's include Code Reuse Attacks (CRAs) and Code Injection Attacks (CIA). An example of an altered control flow can be seen in Fig. 5. Using a control flow graph, the steps through a process can be visualized as a set of directed edges and nodes. An unaltered running process will follow the correct forward and backward edges of its control flow. Any redirection of an edge would cause a control flow violation.

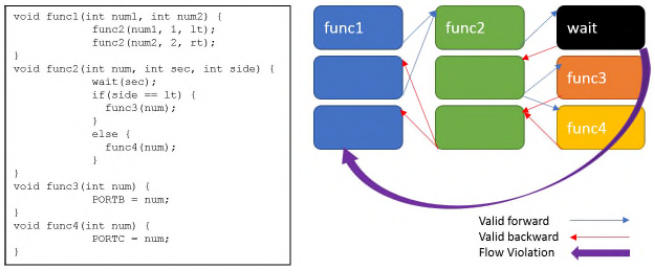

Figure 5. Example control flow graph.

To aid in detection of control flow violations, meta-data from an off-line control flow analysis conducted on the program binary is loaded into the security memory (SM) through a merging process with the FPGA bit stream file. This avoids the need to re-synthesize the FPGA bit stream due to a security recovery scenario. When the FPGA powers on, pre-build control flow meta-data is loaded from the SM into the SU.

Detection is done using a hardware-based CFI solution assuming CRAs. At a minimum the CFI solution will monitor AVR direct, indirect, and relative branch instructions (jmp, call, ret) to determine any violations in the intended control flow at the function block level.

Each time a valid function block is called, a snapshot of the core is taken. Since the PM can't be modified by an attacker, the detected malicious instruction happened in the called function block through a data memory exploit. Upon detection, a previous valid snapshot is loaded into the AVR core. The embedded core continues processing from a state before the attack happened. If the same malicious instruction is encountered, then a reset occurs, clearing out any bad data in the DM.

If a code injection attack is detected, partial or whole portions of the program memory will be reloaded to recover the system to a usable state. To support reloading program memory, the recovery scheme requires an additional copy of the original program binary. The best case scenario would be code injected outside the program binary bounds. In this scenario, only a checkpoint recovery may be needed. The worst case scenario is injected code within the program binary area. This would result in pausing execution to reload program memory and restarting the soft-core from a checkpoint or reset state.

Once malicious code is identified, the system moves active malicious code execution to a sacrificial processor that continues to run the malicious code and collect data, to the Data Output Unit (DOU), to aid in analysis of the malicious code off line. Recovery will be accomplished using clean binaries, stored in the security memory (SM), which are reloaded into the devices primary program memory.

Presently a prototype system on a chip (Fig. 4) implementing an 8-bit Alf and Vegard's RISC (AVR) soft-core processor on a Xilinx Artix 7 series FPGA has been constructed. Further refinement and testing should be completed by the time of the conference.

V. CONCLUSION

In this paper, we described the problem facing federal, state, municipalities and companies in management of roadway assets during times of fluctuating weather conditions to limit damage due to heavy trafficking on their managed roads using load restrictions. The paper describes a system developed to meet the need of placement and removal of SLR's. Described within is the high level systems architecture, sensor, reader and communications architectures used to validate systems operations. The system as presently deployed lacks adequate security from cyber-

Copyright (c) IARIA, 2018. ISBN: 978-1-61208-659-0

criminal activity. To remedy this situation a study is ongoing to redesign the FPGA based components and PCB's in order to integrate security at the core hardware level. The goal being to provide policies and mechanisms to support identification of potential security flaws in the design, develop hardware based detection of malicious activities, to develop hardware solutions supporting response to such attacks and mechanisms to recover to an operational mode after an attach event in near real-time.

ACKNOWLEDGMENT

The project basic sensor development was partially funded by the USDOT Office of the Assistant Secretary for Research and Technology, U.S. Department of Transportation, under grant OASRTRS-14-H-UMDA. The views, opinions, findings and results presented are those of the authors and do not reflect official policy or position of our sponsor, the USDOT/OST-R, or any State or other entity.

REFERENCES

[1] A. H. Bradley, M. A. Ahammed, S. Hilderman and S. Kass, Responding to Climate Change with Rational Approaches for Managing Seasonal Weight Programs in Manitoba. Proceedings of the American Society of Civil Engineers 15th International Conference on Cold Regions Engineering. CD-ROM. Quebec City, Canada, August 19-22, pp. 391 – 401, 2012.

[2] Minnesota Department of Transportation (Mn DOT), "Policy and Process for Seasonal Load Limit Starting and Ending Dates," Minnesota Department of Transportation, Policy, Safety & Strategic Initiatives Division, Technical Memorandum No. 09-09-MAT-02, June 29, 2009.

[3] R. Embacher, "Duration of Spring Thaw Recovery for Aggregate-Surfaced Roads," Transportation Research Record: Journal of the Transportation Research Board No 1967, pp. 27-35, Transportation Research Board of the National Academies, Washington D.C., 2006.

[4] G. L. Hanek, M. A. Truebe and M. A. Kestler, Using Time Domain Reflectometry (TDR) and Radio Frequency (RF) Devices to Monitor Seasonal Moisture Variation in Forest Road Subgrade and Base Materials, U.S. Department of Agriculture, Forest Service, San Dimas Technology and Development Center, San Dimas, CA, 2001.

[5] R. A. Eaton et al., "Spring Thaw Predictor and Development of Real Time Spring Load Restrictions," Proceedings of the American Society of Civil Engineers 14th International Conference on Cold Regions Engineering, Duluth, MN, August 30-September 2, 2009.

[6] R. A. Eaton, R. L. Berg, A. Hall, H. J. Miller and M. A. Kestler, "Initial Analysis of the New Hampshire Spring Load Restriction Procedure," Proceedings of the American Society of Civil Engineers 14th International Conference on Cold Regions Engineering, Duluth, MN, August 30-September 2, 2009.

[7] M. A. Kestler, et al., "Determining When to Place and Remove Spring Load Restrictions on Low Volume Roads: Three Low-Cost Techniques." Low Volume Roads Conference, Austin, TX, Transportation Research Record 1989, pp 219-229, WA, DC. June 2007.

[8] K. Ashton, "That 'Internet of Things' Thing." In: RFID Journal, 22 July 2009. Retrieved 17 December 2012.

[9] K. Dziadak, J. Sommerville and B. Kumar, "RFID based 3D buried assets location system." Journal of Information Technology in Construction, Vol. 13, pp. 155–165, 2008.

[10] F. Faridazar and N. Lajnef, "Intelligent multi-sensor measurements to enhance pavement monitoring and safety." In Passive Wireless Sensor Tag Workshop. NASA, Houston, 2011.

[11] M. Roberti, "RF sensors could optimize crop irrigation." RFID J. Mag. Online. http://www.rfidjournal.com/magazine/article/8172, accessed 12.17.2012.

[12] B. Marquis, "Mechanistic Approach to Determine Spring Load Restrictions in Maine," Technical Report No. 08-1, Maine Department of Transportation, Bangor, Maine, 2008.

[13] J. M. Ovik, J. A. Siekmeier, and D. A. Van Deusen, 'Improved Spring Load Restriction Guidelines Using Mechanistic Analysis," Technical Report, Minnesota Department of Transportation, 2000.

[14] R. A. Eaton, M. A. Kestler, and A. Hall, "Spring Thaw Predictor & Development of Real Time Spring Load Restrictions, First Two-Year Data Report," SP&R Research Project No. 14282K, New Hampshire Department of Transportation, Concord, NH, 2009.

[15] CI Cybersecurity, "Framework for Improving Critical Infrastructure Cybersecurity, Version 1.1," National Institute of Standards and Technology, 2018.

[16] S. Parameswaran and T. Wolf, "Embedded systems security - an overview," Design Automation for Embedded Systems, vol. 12, pp. 173-183, 2008.

[17] R. de Clercq and I. Verbauwhede, "A survey of Hardware-based Control Flow Integrity (CFI)," ACM Computing Surveys, pp: 27, 2017.

[18] A. Francillon, D. Perito and C. Castelluccia, "Defending embedded systems against control flow attacks," in Proceedings of the first ACM workshop on Secure execution of untrusted code, pp: 19-26, 2009.

Copyright (c) IARIA, 2018. ISBN: 978-1-61208-659-0