While hackers had their sights on IT, embedded devices for the most part were spared. Until recently. Hackers have begun targeted more traditional embedded platforms. Ironically like in IT, fame is the initial motivator but before long financial gain and outright destruction will most likely be the end game.

IMPLEMENTING SECURITY IN EMBEDDED LINUX SYSTEMS

MONTAVISTA SOFTWARE, LLC.

Anantvijay Kulkarni
Lead Technical Solutions Engineer

Iisko Lappalainen
Senior Manager, Technical Pre-Sales & Solutions

Jim Gallagher
Senior Marketing Team Lead

July 2016

Contents

Executive Summary

Last summer, wired.com posted a frightening but all too real article about two white hats who hacked into a Jeep wirelessly, taking control of the car's radio, environmental system and even more dangerous the brakes and engine[1]. Some thought it might happen. Could possibly but probably not. At least not "my car". In it, they revealed they could send commands to the car's critical components through the internal CAN bus network. Access to the CAN bus came through a vulnerability they exploited in Chrysler's Uconnect entertainment and navigation system. They accomplished this 10 miles away from the hijacked Jeep, from the cozy confines of their basement. This all sounds like something from a bad science fiction movie but reality is hackers have the ability to attack our embedded systems. Scary. Unthinkable. And now a real threat to embedded systems.

To think, that today embedded systems are prevalently everywhere. From our routers and switches, medical monitoring devices, mobile phones, tablets, wearables, video game boxes, cars, home security systems, embedded is driving this age of Internet of Things (IoT). More importantly, we are just getting started. According to Cisco, it estimates 50 billion devices will connected to the Internet by the year 2020. [2] With that many new devices always on and always connected, it will offer up exponentially more attack surfaces for hackers to target.

The good news is attacks like this Jeep hijack can be avoided and thwarted. There are various security measures available to embedded developers to potentially fend off an attack or breach. Especially when using embedded Linux, the popular choice for IoT and Embedded Operating Systems.[3] Linux offers a choice of proactive and reactive security approaches. In this first in a series of Linux Security Whitepapers, we will review the reason why security is important and a review of these different proactive and reactive security technologies.

[1] Adam Greenberg, "Hackers Remotely Kill A Jeep On The Highway – With Me In It" (https://www.wired.com/2015/07/hackers-remotely-kill-jeep-highway/)
[2] Cisco: http://www.cisco.com/c/en/us/solutions/internet-of-things/overview.html
[3] Eric Brown, "Embedded Linux Keeps Growing Amid IoT Disruption, Says Study" (https://www.linux.com/news/embedded-linux-keeps-growing-amid-iot-disruption-says-study)

Introduction

The Need to Implement Security in Embedded Devices

Network and Cyber Security has been an essential part of daily lives for several decades. The focus for penetrating secure systems started with IT and commercial economy. At first, attacker motivation was for fame and credit in gaining access to a server to alter a website. This evolved quickly to financial gain through ransoming information/access and stealing critical data for resale. Recent incidents at Target, Sony Pictures, and MedStar Health hospitals (crypto-ransomware) show the financial and reputation impact as well as the rapid sophistication attack.

In the past, hackers were highly skilled programmers who understood the details of computer architecture, network communications, software, and how to exploit security vulnerabilities. Hackers now are collaborating with each other working together in "networks" to more efficiently create exploits. Some are even selling or contributing freely baseline exploit "how to" kits and attack programs. Today almost anyone can become a hacker by downloading these tools from the Internet.

While hackers had their sights on IT, embedded devices for the most part were spared. Until recently. Hackers have begun targeted more traditional embedded platforms. Ironically like in IT, fame is the initial motivator but before long financial gain and outright destruction will most likely be the end game. Here are some of the recent real world examples of security compromises in embedded devices that not only potentially impact financially but could also result in loss of human life:

- **Mobile Devices Inc.:** By sending carefully crafted SMS messages to one of those cheap dongles connected to the dashboard of a Corvette car, the researchers were able to transmit commands to the car's CAN bus—the internal network that controls its physical driving components—turning on the Corvette's windshield wipers and even enabling or disabling its brakes.

- **Boeing and Airbus:** A hacker used the in-flight Wi-Fi connection to hack in the flight control system and managed to control thrust for engines, the oxygen masks, etc.
- **Platforms running Linux:** Researchers revealed in January 2016 an undiscovered flaw in the Linux kernel that allows an attacker to elevate user privileges to root offering the ability to create an exploit to extract cached security data. This vulnerability affects 3.8 and later Linux kernel devices such as IT infrastructure, Telecom Infrastructure, and IoT Gateways.
- **Drones:** A John Hopkins University team demonstrated 3 different ways to send unwanted commands to small unmanned flying devices (drones) causing them to land or spontaneously crash. As use of drones are increasing in agriculture, oil & gas, law enforcement, and commercial business (Amazon, Apple, Walmart), continued exploits such as these could result in privacy violations, stolen goods, and physical harm from drone hijack.

Why target exploits at embedded now? It's all about the exponential increase in attack surfaces and potential data/reputation/operations loss leading to financial gain and more disturbing disruption and discredit of business. Looking at the rise in the recent security breaches, the major factors responsible for this increase are network connectivity, software extensibility, and code complexity.

- **Network connectivity:** Some years ago no public services were really online. Today, almost all of them are available online in several countries. Cloud-based data can be accessed by anyone with the keys, not just the user. Finally, with modern devices such as mobile phones, tablets, video game systems, home cable gateways, and home security systems always on and connected, the available attack vectors have increased exponentially (and growing!) with most devices being accessed from around the world.
- **Extensible services:** Plug-in frameworks, web-apps and cloud concepts make it hard to guard against unknown combinations of components in the end solution. Even while a single component can be secure, then combination can have hard-to-detect flaws. Also the modular designs might allow easier plug-in of malicious modules for example. Previously systems were more static and easier to assure.
- **Complexity:** Bringing up the level of abstraction increases exponentially the complexity of the software from the bottom-up perspective. In general, more code brings in the possibility of more bugs.

With the ever-growing presence of Linux in the various markets like Embedded devices, IoT, Telecommunications, Military/Aerospace, Medical instruments, Enterprise servers, etc., there is an increased demand to create and implement robust security in embedded platforms. Financial implication arising out of the security breach is also one of the major driving forces for implementing security.

Solution Overview

Implementing Embedded Security through Linux

The need to produce robust and highly secure devices is critical as we move into the always on, always connected era. But where do developers start and what security features and technologies are available to them?

There are many ways at implementing security measures in an embedded device. One way is to look at it in terms of reactive and proactive security measures. Reactive security measures are ones that try to minimize the damage if an attacker succeeds in hacking the device whereas proactive security measures are ones that try to secure the device as much as possible from being penetrated. Let's explore these 2 ways in more details.

Reactive Security

Reactive security technologies are mostly usable after an exploit is made or known. For instance, an attacker has developed an exploit taking advantage of an inherent vulnerability (or flaw) to gain unauthorized access to a personal computer, mobile device, or network. Sometimes an exploit is found in research by "White Hats" (i.e. the good guys). Reactive security measures react to these exploits after they have been found, detected, or disclosed. Here are some of the reactive security measures.

Intrusion Detection Systems (IDS)

The purpose of an IDS is to detect if the system is compromised. Knowing that the system is compromised helps in preventing further damage. Apart from minimizing the damage, these tools also provide details on how the system was compromised through auditing and logging functionality. This information can be invaluable in discovering how the hack was accomplished and fixing the security hole to prevent further attacks. MontaVista employs technologies like Samhain, Auditd, and Tripwire that actively monitor/analyze the logs, checks file integrity, and performs port monitoring. Auditd in particular is a powerful tool to monitor details of system operation. While it is a userland package, auditd operates at the kernel level monitoring such activities as login

attempts/failures, user login history, file access, and network traffic. Since auditd gives detailed visibility into a system, reporting information can be used to discover a security attack and potentially how it was infiltrated. Additionally, information from auditd can be used in real-time to analyze and monitor out of the ordinary behavior on the system to potentially block an eminent breach.

Common Vulnerabilities and Exposures (CVEs)

The Common Vulnerabilities and Exposures (CVE) system provides information on publicly known security vulnerabilities and exposures. CVEs were created to bring some commonality in categorizing these threats across various security tools, databases, and services. MITRE Corporation maintains the system, with funding from the National Cyber Security Division of the United States Department of Homeland Security.

Monitoring and fixing CVEs is essential if deploying connected devices. CVEs are increasing dramatically each year. According to research done by Casper Manes using data from the National Vulnerability Database (NVD) the number of discovered CVEs in 2015 increased to 8822, up from roughly 7000 in 2014.[4] That's approximately a 25% increase year over year. And while Apple's OS X surprisingly had the most operating systems vulnerabilities (384), Linux taken as a whole across the number of distributions (Canonical, openSUSE, Debian, Linux Kernel, and Fedora) was even more than Apple at 499 vulnerabilities:

rank	operating system	number of vulnerabilities
1	Apple OS X	384
2	Microsoft Windows Server 2012	155
3	Canonical Ubuntu Linux	152
4	Microsoft Windows 8.1	151
5	Microsoft Windows Server 2008	149
6	Microsoft Windows 7	147
7	Microsoft Windows 8	146
8	Microsoft Windows Vista	135
9	openSUSE	121
10	Debian Linux	111
11	The Linux Kernel	77
12	Microsoft Windows 10	53
13	Fedora Linux	38
14	Microsoft Windows 2003	36
15	Xen OS	34

[5]

This graph supports what we already have surmised; hackers are persistent, executing attacks more frequently (possible as we discussed by collaborating), and targeting more

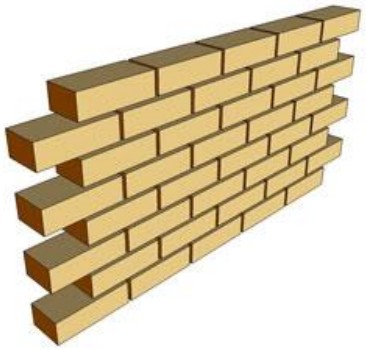
[4] Casper Manes, "2015's MVPs – The most vulnerable players",http://www.gfi.com/blog/2015s-mvps-the-most-vulnerable-players/

[5] Graph reprinted from Casper Manes "2015's MVPs – The most vulnerable players" article, http://www.gfi.com/blog/2015s-mvps-the-most-vulnerable-players/

and more attack surfaces. Thus the need to quickly identify and fix CVEs becomes that more critical and demanding.

MontaVista proactively monitors emerging CVEs. Moreover, we pay for membership into groups that provide early warnings of CVE's before they go public. We research all of the CVE's that might be relevant to our distribution and if we determine that the CVE is applicable to our code base or any of our supported applications, it becomes a top priority for our engineering team to find or create a patch, verify the patch fixes the vulnerability, and validate the new fix does not negatively affect other components. For high severity CVEs that have not been disclosed, MontaVista follows the Security Technical Information Guide (STIG) in handling this sensitive information having a patch ready for our customers when the vulnerability information embargo is lifted. This is an expensive and time consuming process that MontaVista offers on a quarterly basis so our customers can focus on building their valued product offering.

Proactive security



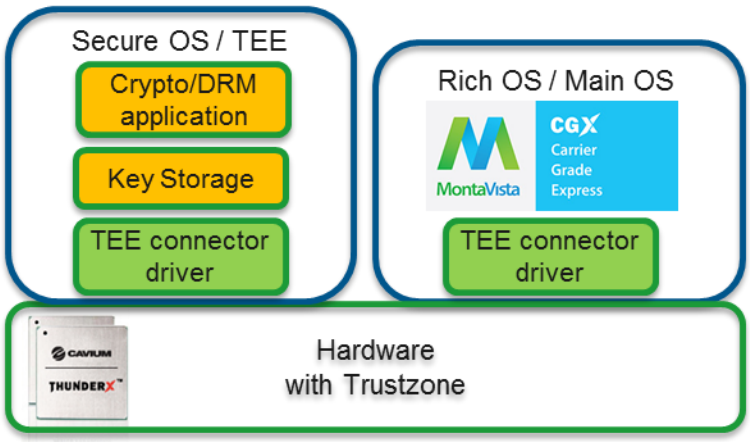
Proactive security methods differ from reactive ones in the goal is preventing a hack whether known or not. In general, they create walls for the attacks involving exploits on vulnerabilities that are not yet known, sometimes referred to as Zero Day attacks. The aim here is to harden the device as much as possible. There are various approaches in applying proactive security measures in embedded systems that we will examine.

Root of Trust and Attestation

The root of trust is a method to build a secure software environment from the hardware up to the applications. It protects the system against software tampering by signing the binaries or using checksums. This allows that only trusted software can run on a device.

MontaVista can enable customers to use secure crypto-processor based on Trusted Platform Module (TPM) standard.

ARM TrustZone and Trusted Execution Environment (TEE)



ARM has included TrustZone-enabled IP in their cores for almost 10 years. TrustZone allows for setting up trusted devices, memory areas, and even trusted interrupts at system boot. It allows running a "secure world" operating system or executive, enabling a degree of virtualization and isolation supported by hardware separation. In case of ARM TrustZone enabled devices, TPM can be replaced by TrustZone which also would result in BOM cost saving. An example of TrustZone setup is Samsung Galaxy S3 (e-commerce, NFC, secure login).

A Trusted Execution Environment (TEE) can run inside TrustZone or in a similar environment that can fence off the secure components from the main system. TEE itself is a Global Platform standard for such operating environments. Available implementations include for example SierraTEE and OP-TEE (supported by Linaro).

Mandatory Access Control (MAC) Frameworks

MAC frameworks limit the user ID-based access control further by allowing role-based access control where the role of the user can change depending on the security context (mostly process) he's running. This allows protecting against more complicated scenarios than only user-based access control. SELinux is one of the popular MAC frameworks. MV has been using SELinux for solution development, for enhancing security in specific customer projects and in some of its products.

Conclusions

In today's world, always on and always connected needs always being secure. Hackers are mobilizing and collaborating with an eye to gain financial windfall, discredit government/company reputation, and disrupt business. IBM estimates in 2016 the average cost for each stolen sensitive or confidential information record will be $158.[6] This does not include any cost associated with discredited reputation or business disruption. Priceless comes to mind.

Embedded devices are now targets of hackers. This puts responsibility of the developers to include security measures when they are building the next great IoT gateway, router, home security system, or heart monitor. Solid proactive and reactive security measures are available today in Linux to thwart these attacks. The challenge is for the community to educate and demand enforcement so that the measures are deployed. Else, embedded security will be reminiscent of hockey great Wayne Gretsky's quote, "You miss 100 percent of the shots you never take." Embedded systems will allow 100 percent of attacks to succeed due to taking ***no shot*** at deploying security measures.

[6] IBM 2016 Ponemon Cost of Data Breach Study (http://www-03.ibm.com/security/data-breach/)

Appendices

- Adam Greenberg, "Hackers Remotely Kill A Jeep On The Highway – With Me In It" (https://www.wired.com/2015/07/hackers-remotely-kill-jeep-highway/)
- Cisco: http://www.cisco.com/c/en/us/solutions/internet-of-things/overview.html
- Eric Brown, "Embedded Linux Keeps Growing Amid IoT Disruption, Says Study" (https://www.linux.com/news/embedded-linux-keeps-growing-amid-iot-disruption-says-study)
- Scott Pack, "A Brief Introduction to auditd"(http://security.blogoverflow.com/2013/01/a-brief-introduction-to-auditd/)
- Casper Manes, "2015's MVPs – The most vulnerable players", http://www.gfi.com/blog/2015s-mvps-the-most-vulnerable-players/
- CVE: https://cve.mitre.org/index.html
- Mandiant Consulting (Fireeye) M-Trends 2016
- Corvette hacked - http://www.wired.com/2015/08/hackers-cut-corvettes-brakes-via-common-car-gadge.
- Boeing and Airbus hacked - http://edition.cnn.com/2015/05/17/us/fbi-hacker-flight-computer-systems
- TrustZone setup is Samsung Galaxy S3 - http://armdevices.net/2012/05/04/samsung-galaxy-s3-may-be-the-first-smartphone-with-full-arm-trustzone-support-for-enabling-100-security-in-everything-online
- IBM 2016 Ponemon Cost of Data Breach Study (http://www-03.ibm.com/security/data-breach/)

MontaVista Software, LLC | 2315 North 1st Street San Jose, CA, 95131 | www.mvista.com

DOC. ID. MVWP-SEC1-070616