LAWRENCE
LIVERMORE
NATIONAL
LABORATORY

LLNL-CONF-674264

Trends in Hardware Authentication

G. K. White

July 7, 2015

INMM
Palm Springs, CA, United States
July 12, 2015 through July 16, 2015

Disclaimer

This document was prepared as an account of work sponsored by an agency of the United States government. Neither the United States government nor Lawrence Livermore National Security, LLC, nor any of their employees makes any warranty, expressed or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States government or Lawrence Livermore National Security, LLC. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States government or Lawrence Livermore National Security, LLC, and shall not be used for advertising or product endorsement purposes.

Trends in Hardware Authentication

Greg White
Lawrence Livermore National Laboratory
July 2015

Abstract

Recent projects intended to create open source hardware designs have principles and practices that can be applied to hardware design for non-proliferation and arms control regimes which increase authentication. This includes the Novena laptop, littleBit's Cloudbit, and OLimex's OLinuXino Nano. Raspberry Pi will be used as a vehicle for both good and bad authentication features. Recent revelations regarding SD card development will be discussed. It will start with a technical discussion and applicable market forces. These market forces lead to counterfeiting and additional functionality which makes authentication harder. Characteristics of an optimal processor for non-proliferation regimes will be discussed. Finally, we will propose using littleBit's Cloudbit as a starting point for the community to design a new CPU board which could be used across a wide range of non-proliferation and arms control regime purposes, where ease of authentication is maximized. We will discuss what design decisions and choices will be made as part of this design.

1 – Introduction

As we make progress toward the deployment of monitoring systems for nuclear material, two important goals must be observed: protection of the host country's sensitive information and assurance to the monitoring party that the nuclear material is what the host country has declared it to be. These goals are met by *certification* in the host country and *authentication* by the monitoring party. During both certification and authentication, each side needs to understand all of the operating parameters of the hardware and software in the deployed system. This paper concentrates on hardware authentication, but similar principles apply to software authentication, as well as to software and hardware certification.

Authentication is the process of gaining assurance that a system is performing robustly and precisely as intended. The simpler the system, the easier it is to authenticate. It is important to limit functionality to only what is needed to satisfy the requirements of the task. Each design decision makes authentication easier, or harder. Simpler hardware, expressed in the number of gates/transistors, chips, or boards, is easier to authenticate than more complex hardware. The same can be said for application and development software.

Other industries have a similar need for authentication. Computers that perform electronic voting and gambling are disparate examples. In previous INMM papers,[1,2,3,4] we have discussed a hypothetical perfect system for authentication, with transparent (to both parties) hardware and software development, and advocated "open source" hardware and software solutions. We advocated software language choices that lower authentication costs, specifically comparing procedural languages with object-oriented languages. In particular, we examined the C and C++ languages, comparing language features, code generation, implementation details, and executable image size, and demonstrated how these attributes aid or hinder

[1] White, G., Increasing Inspectability of Hardware and Software for Arms Control and Nonproliferation Regimes, *Proceedings of the INMM 2001 Annual Meeting*, Indian Wells, California
[2] White, G., Computer Language Choices in Arms Control and Nonproliferation Regimes, *Proceedings of the INMM 2005 Annual Meeting*, Phoenix, Arizona
[3] White, G., Strengthening Software Authentication with the ROSE Software Suite, *Proceedings of the INMM 2006 Annual Meeting*, Nashville, Tennessee
[4] White, G., Tools for Authentication, *Proceedings of the INMM 2008 Annual Meeting*, Nashville, Tennessee

authentication. We showed that programs in lower level, procedural languages are more easily authenticated than object-oriented ones. We suggested some possible ways to mitigate the use of object-oriented programming languages. We described the scope of the software authentication process and the five methods of software authentication. We then concentrated on different types of source code analysis, introducing LLNL's ROSE software tool for automating the authentication of source code. Finally, we discussed how authentication of binaries is complementary to source code authentication.

Designing hardware for authentication is a complex task. Using older and simpler computers (i.e. retro) is often a good choice. Embedded computers often share the same goals as designing for authentication. Processors should be chosen that are just fast enough for the task, use older chip packaging, have less integrated extras (DSP, video, etc.) and have open documentation. Software should have a small image footprint and should be open source. Storage should be only as large as needed. However, we need hardware that has a long lifespan. This lifespan is shorter for older hardware. We can manage this shorter lifetime by large purchases during the active manufacturing cycle.

2 – Raspberry Pi

The Raspberry Pi[5,6] is a credit card sized Linux computer built for education. For this paper, we will use it as a vehicle for talking about designing hardware for authentication. It has adequate processing power for use in past attribute measurement systems[7], but not enough for imaging. The Raspberry Pi runs a wide range of open source operating systems[8] and has the ability to expand its input/output ports via expansion boards. Its low cost ($25-35) supports random selection. There are also no developer ties back to either the host or the monitor. Over 5 million Raspberry Pi computers have been sold[9]. The processor has a small initializing code to boot the operating system over the SD card. There is no writable non-volatile memory. Depending on the model, it only has 2 or 3 integrated circuits. There are also multiple manufacturers and distributors of the assembled boards.

However, the Raspberry Pi has a number of design choices which decrease authentication. The processor is not available for purchase as an individual integrated circuit. This is intended to maintain the quality and compatibility of Raspberry Pi computers. There is also unneeded additional functionality. There is a Digital Signal Processor on the CPU and it is only documented under a non-disclosure agreement. The Raspberry Pi also has a Graphical Processing Unit (GPU), but it is becoming more open as time passes. This has happened as the number of Raspberry Pi computers has been sold. The documentation for the GPU was released publically in February 2014. The PC board is mostly open source. The schematic is 4 pages and publically available. The PC board is a 6 layer board, which would be harder to authenticate and the layout is not publically available (probably to deter the clone market). Finally, the CPU and the memory chips are stacked on top of each other using Package on Package technology. Two chips stacked on top of each other invite something to be placed between them. Each of the two chips uses Ball Grid Arrays which are harder to authenticate.

Chip packaging can make authentication easier or harder. Dual Inline Package and Quad Flat Packages are easier to authenticate, because the leads are on the edge of the chip. Pin and Ball Grid Arrays are harder to authenticate. This is because it requires traces on the PC board to run under the chip and usually requires more PC board layers.

---

[5] https://www.raspberrypi.org/
[6] https://en.wikipedia.org/wiki/Raspberry_Pi
[7] White, G., Review of Prior U.S. Attribute Measurement Systems *Proceedings of the INMM 2012 Annual Meeting*, Orlando, Florida
[8] https://www.raspberrypi.org/downloads/
[9] https://www.raspberrypi.org/five-million-sold/

3 – SD Cards[10]

Secure Digital[11] (SD) cards are often used as local storage on embedded computers. They are postage stamp sized and inexpensive. They have a write protect switch, but it is enforced by software on the host computer. The SD card market is a cutthroat industry. There is almost no profit margin. This leads manufactures to sell every chip produced; a chip with 80% bad blocks will be sold as a chip with less capacity. The market has a severe problem with counterfeiting; imitating the packaging and labels of a high-quality brand increases the profit of low quality SD card manufacturers.

The common believe is that SD cards only contain flash memory. However, many manufacturers have found an advantage in adding a small microprocessor. Manufacturers can use it to perform bad block testing and remapping, capacity reporting, write and wear leveling, and error correction. These microprocessors are often ARM or 32-bit 8051 variants and have their own application memory and are externally programmable. This microprocessor can run at up to 100 MHz. This decreases authentication. It allows the microprocessor to act as a man-in-the-middle. The microprocessor can perform eavesdropping, have two switchable banks of memory, pretend to write protect memory, insecure erase of contents, and scan for specific memory patterns (such as crypto keys). Although primarily used in low-end SD cards, it is also used in high-performance flash memory devices (such as eMMC by both Samsung and Micron).

4 – Designing for Authentication vs. Industry Trends

Our needs for authentication are often not aligned with industry trends. Available SD cards are much larger than we need. Most processors are only available on Ball Grid Array packaging. Many of them have extra features we do not need, such as GPUs and CPUs. In addition most of them include large amounts of flash memory, either on the processor, or on the PC board.

5 – Authenticating Programmable Devices

Programmable devices, such as Field Programmable Gate Arrays (FPGA) and Complex Programmable Logic Devices (CPLD), have been proposed in future measurement systems. There are a number of potential problems or obstacles to authentication. Many of them use BGA packaging and/or contain extra features we don't need. This may include microprocessors or complex interfaces. Like SD cards, it can be hard to find a programmable device that is properly sized for our needs; Most are significantly bigger than we need. Most FPGAs are programmed via writable flash memory devices. We would strongly prefer write once, read many devices.

There are a lot of parallels between software development and programmable devices. This is shown below:

	Software	**Programmable Devices**
Source	High level language	VHDL/Verilog source
Binary Form	Executable image	Gate level definition
Tools	Compilers, Loaders, Assemblers	Compilers, placement and routing tools, Device loaders

Just like with software development, it is important that programmable device development tools are open source. However, unlike with software development, the vast majority of programmable device

[10] Andrew "bunnie" Huang and Sean "xobs" Cross, The Exploration and Exploitation of an SD Memory Card, 30th Chaos Computing Congress, 2013, http://media.ccc.de/browse/congress/2013/30C3_-_5294_-_en_-_saal_1_-_201312291400_-_the_exploration_and_exploitation_of_an_sd_memory_card_-_bunnie_-_xobs.html#video http://www.bunniestudios.com/blog/?p=3554

[11] https://en.wikipedia.org/wiki/Secure_Digital

development tools are proprietary and closed source. Here is the list of programmable device development tools that are open source[12]:

	Project
Verilog/VHDL Compiler	Icarus Verilog[13] Yosys Open Synthesis Suite[14]
Placement and Routing	Arachne-pnr[15]
Tools	Project IceStorm[16]

It appears that the only fully open source FPGA development toolchain is for the Lattice Semiconductor iCE40 FPGA family. This family of chips comes in a wide range of sizes, and has some variants that come in QFP packaging. We believe this deserves future research.

6 – Next Steps

We should work towards creating a more authenticatable, certifiable, and inspectable computer for arms control and proliferation regimes. It should be a two layer board, have no traces under devices, and avoid stacked chips or chips with BGAs. As a starting point, we could use the OLinuXino Nano[17] or the Littlebits Cloudbit[18] which is derived from it. Both are based on the Freescale iMX233 ARM processor running at 454 MHz, which is properly sized for our needs. The processor is a QFP package with 64MB memory in a DIP package. Freescale provides 1600 pages of documentation about the processor. With both the Nano and the Cloutbit, the schematic is one page. They boot the operating system off an SD card. The next step would be to relayout one of the boards as a two layer design. We hope to try this in the coming year.

For SD cards, the community should try and find an alternative. It should be much smaller (a small number of megabytes instead of gigabytes), provable hardware write-protect, and not contain any extra functionality like microprocessors.

Prepared by LLNL under Contract DE-AC52-07NA27344.

[12] http://hackaday.com/2015/05/29/an-open-source-toolchain-for-ice40-fpgas/#more-157512
[13] http://iverilog.icarus.com/
[14] http://www.clifford.at/yosys/
[15] https://github.com/cseed/arachne-pnr
[16] http://www.clifford.at/icestorm/
[17] https://www.olimex.com/Products/OLinuXino/iMX233/iMX233-OLinuXino-NANO/open-source-hardware
[18] http://littlebits.cc/bits/cloudbit