Every Cloud has a Silver Lining

Industry Standards, Best Practices, and Recommendations for Embedded System Security

Joe Grand (@joegrand)
Every Cloud has a Silver Lining

- Embedded Security Concepts
- Standards / Guidelines
- Best Practices
- Product / Vendor Resources
Embedded Security Concepts
Security Overview

• Security is a process
  – Constantly changing to reflect "state of the art"
  – The attacker usually has the advantage
  – Treat security as an integral part of system design, continue to evaluate during development and revisions

• Given enough time, resources, & motivation, an attacker can break any system
  – Reduce risk to an acceptable level
  – Costs of a successful attack should outweigh potential rewards
Threat Model / Risk Analysis

• You must understand your risk before you can protect yourself
  – What needs to be protected
  – Why it is being protected
  – Who you are protecting against (define your adversary)

• What features are absolutely necessary for system functionality?
  – Each new feature increases attack landscape

• Identify single points of failure across the lifecycle
  – Design, fabrication, integration/assembly, distribution, in-the-field
## Types of Hackers / Attackers

<table>
<thead>
<tr>
<th>Resource</th>
<th>Curious Hacker</th>
<th>Academic</th>
<th>Organized Crime</th>
<th>Government</th>
</tr>
</thead>
<tbody>
<tr>
<td>Time</td>
<td>Limited</td>
<td>Moderate</td>
<td>Large</td>
<td>Large</td>
</tr>
<tr>
<td>Budget ($)</td>
<td>&lt; $1000</td>
<td>$10k - $100k</td>
<td>&gt; $100k</td>
<td>Unknown</td>
</tr>
<tr>
<td>Creativity</td>
<td>Varies</td>
<td>High</td>
<td>Varies</td>
<td>Varies</td>
</tr>
<tr>
<td>Detectability</td>
<td>High</td>
<td>High</td>
<td>Low</td>
<td>Low</td>
</tr>
<tr>
<td>Target/Goal</td>
<td>Challenge</td>
<td>Publicity</td>
<td>Money</td>
<td>Varies</td>
</tr>
<tr>
<td>Number</td>
<td>Many</td>
<td>Moderate</td>
<td>Few</td>
<td>Unknown</td>
</tr>
<tr>
<td>Organized?</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Release info?</td>
<td>Yes</td>
<td>Yes</td>
<td>Varies</td>
<td>No</td>
</tr>
</tbody>
</table>

P. Kocher, Crypto Due Diligence, RSA Conference 2002
Common Attack Surfaces

MEMORY
- Read (dump object code)
- Replace
- Re-flash

BUS
- Monitor
- Man-in-the-middle

ACTIVE SIDE CHANNEL
For example, glitch attack

NETWORK ATTACK
- Protocol attack
- Attack bad implementation
- Gain foothold
- Inject persistent malware

OPEN PORT
- UART
- JTAG
- Default password

Source: Microsemi Corp.

PASSIVE SIDE CHANNEL
- Power analysis (DPA)
- Electromagnetic analysis

The Biggest Security Threats Facing Embedded Designers, Electronic Design, June 2016
Security Concerns

Preparation for the Internet of Things

Importance of security in products

- Not very: 16%
- Somewhat: 33%
- Very: 51%

How important will security be in future products?

- Same: 43%
- Less: 3%
- More: 54%

Companies that will produce connected products: 43%
The Barr Group's latest survey results (see http://www.barrgroup.com/Embedded-Systems/Webinars/Survey-Results-2016) from embedded programmers reveals some interesting statistics about safety and security in current development. I have taken a closer look at the results (see "Barr Group Survey Results Reveals Embedded Safety and Security Trends" on electronicdesign.com) and a couple stand out. The average years of experience of the responders was 15.9 years.

The results from the question about primary security concerns (Fig. 1) indicate that product tampering, cloning, and theft of IP are high on the corporate priority list. These are important issues, but they do seem to play second fiddle to customer-related concerns highlighted in orange, such as injury and death.

The problem is that developers have a finite amount of resources and protections against cloning and IP theft, but do not always help improve a product's overall safety and security. The bigger question is whether companies limit their security support to only addressing these types of issues.

The other aspect that jumped out at me was code-standard enforcement (Fig. 2). Only a small fraction utilize fully or partly automated compliance. Code reviews and voluntary compliance made up the bulk of the responses. I do find code reviews useful, but they are better used to finding architectural bugs. Software tends to do a better job at finding compliance issues.

The usual reason for coding standards is to reduce errors due to improper usage of tools. Unfortunately, C remains the dominant embedded tool and C allows a programmer to easily make mistakes that automated tools can catch.

No survey can capture all the nuances of engineers, but it is definitely worth looking at the details of what the Barr Group has come up with.

### Primary Security Concerns

(Respondents were asked to select all that apply.)

- Product tampering
- Theft of data
- Theft of IP
- Customer privacy violation
- Product cloning
- Denial of service
- Injury or death
- Theft of service
- Blackmail or ransom
- Other

Responses: 1,459

Source: Barr Group, 2016
Easier Said Than Done

- Challenge of cost v. security v. convenience
- Implementation is product specific/resource dependent
  - No one-size-fits-all solution
- However, security solutions/techniques/resources becoming more accessible
  - Still requires some level of security competency
  - Be sure to independently verify what you're implementing
Standards / Guidelines
Standards / Guidelines

• Can be used as a checklist/starting point
  – Usually consists of what to do, not how to do it
• Some markets require full compliance to specific standards
  – Arguably a detriment to security if standard is too strict (e.g., only allow a specific process or encryption algorithm)
• Just because a device conforms doesn't make it impenetrable

Standards / Guidelines 2

• International Telecommunication Union (ITU) X.800
  – Security architecture for Open Systems Interconnection for CCITT applications
  – Guidelines/definitions of security-related architectural elements needed for communication between open systems
  – www.itu.int/rec/T-REC-X.800-199103-I/en
Standards / Guidelines 3

• National Institute of Standards and Technology (NIST) Computer Security Resource Center
  – Guidelines/recommendations/references for many aspects of secure systems
  – SP 800: Computer Security
  – SP 1800: Cybersecurity Practice Guides
  – SP 500: Computer Systems Technology
  – http://csrc.nist.gov/publications/PubsSPs.html
Standards / Guidelines 4

• Underwriters Laboratories Cybersecurity Assurance Program
  – UL 2900 Outline of Investigation for Software Cybersecurity for Network-Connectable Products
    • Part 1: General Requirements
    • Part 2-1: Healthcare Systems
    • Part 2-2: Industrial Control Systems
  – Standards only available for purchase
  – http://industries.ul.com/cybersecurity
Standards / Guidelines 5

- Federal Information Processing Standards (FIPS)
  - FIPS 140-2 Security Requirements for Cryptographic Modules

- Common Criteria
  - International standard for computer security certification (ISO/IEC 15408)
  - Verified by independent testing laboratories
  - www.commoncriteriaportal.org
Standards / Guidelines 6

- Trusted Platform Module (TPM 1.2/2.0)
  - Standard/specification for secure cryptographic coprocessor
  - On-chip encryption/decryption/signing/key storage/RNG
  - [www.trustedcomputinggroup.org/tpm-main-specification/](http://www.trustedcomputinggroup.org/tpm-main-specification/)
Standards / Guidelines 7

• Avoiding the Top 10 Security Flaws

• U.S. Dept. of Homeland Security (DHS) Strategic Principles for Securing the IoT
  – High-level guidelines/recommendations
  – www.dhs.gov/securingtheIoT

• Online Trust Alliance (OTA) IoT Trust Framework
  – Guidelines/recommendations for user privacy/security
  – https://otalliance.org/initiatives/internet-things
Global System for Mobile Communications Association (GSMA) IoT Security Guidelines
  - Guidelines/recommendations for endpoint devices/service providers/network operators

Industrial Internet Security Framework (IISF)
  - www.iiconsortium.org/IISF.htm
Standards / Guidelines 9

• FDA Cybersecurity
  – Principles/considerations for managing security in medical devices
  – Also involved in assessing security threats in released products
  – www.fda.gov/MedicalDevices/DigitalHealth/ucm373213.htm

  – Guidelines/recommendations for managing security in automotive electronic systems/communication networks/control algorithms
Best Practices
Best Practices

• Proper design principles can go a long way
  – If implemented correctly...

• Remove the low-hanging fruit
  – Increase difficulty of attack

• Strive for simplicity
  – Each security feature should support a defined goal
Best Practices 2

• Compartmentalization
  – Distribute design documentation on a need-to-know basis
  – Be aware of where/how documentation appears online (firmware update packages)

• Board-Level
  – Remove all non-necessary information
  – PCB silkscreen (designators, fab markings, logos)
  – Component/IC markings (part numbers, logos)
  – Hide critical signals on inner layers, use buried vias
  – Only obfuscation, but increases reverse engineering time
Best Practices 3

• Security Fuses
  – Prevents full read-out or access to a specific memory area
  – Most commonly used on MCU internal memory
  – Easy to enable during code compilation or device programming
  – May still be exploited via brute force, glitch, die attack, off-shore services

• On-Chip Debug/Program/Diagnostic Interfaces
  – Disable or remove completely for production units
  – Implement password/authentication mechanism (may not be part of standard interface)
  – Possibly inconvenient for legitimate personnel (manufacturing, service/repair)
Best Practices 4

• Coding
  – Take care to handle undefined behavior, memory leaks, buffer overflows/bounds checking, invalid data structures, off-by-one, etc.
  – Remove debug symbols/tables, enable optimization
  – Mechanism to update/patch vulnerable code/OS (if needed)
  – Couple w/ source code review, static analysis

• Network Configuration
  – Don't use default login credentials (username/password)
  – Don't add backdoors for future use
  – Close unused ports/daemons/configuration/management interfaces
  – Learn about common network/OS exploits
Best Practices 5

• Anti-Tamper
  – Prevent/deter/detect physical access or tampering of embedded system
  – Resistance, evidence, detection, response

• Run-Time Diagnostics/Failure Modes
  – Ensure device is fully operational at all times (watchdog, periodic system/memory checks)
  – Detect when system is being operated outside of defined conditions (voltage, timing, thermal, optical glitching)
  – Determine how product handles failure (halt/shutdown system, erase critical memory areas)
Best Practices 6

• Encryption
  – For both data at rest and in motion (including firmware, if possible)
  – Consider key management/storage, cipher type
  – Many vendors offer on-chip support for encrypted memory areas
  – Beware of how unencrypted data could be accessed during operation (chip-to-chip communication, debug interface to RAM)
  – For wireless systems, use available security features (check if protocol has already been broken)
  – Use industry standard, publicly scrutinized/analyzed/proven ciphers
    • Don't roll your own!
Best Practices 7

• Secure Boot Process
  – Each stage verifies subsequent stage
  – Only execute trusted code (verified origin/integrity)
  – Prevents arbitrary code execution (unless defeated, commonly done via glitch or patch of hash compare)

Protecting networked designs from intrusion w/ secure FPGAs, Signal Processing Design, Oct. 2013
Best Practices 8

• Side-Channel Prevention
  – Unintentional leakage from system
  – Consider power, EM/RF, timing, thermal
  – See Rambus DPA Countermeasures
  – Many compilers generate side channels unintentionally
Product / Vendor Resources
Product / Vendor Resources

- No endorsement given!
- Evaluate before implementation
  - Some versions may already have been broken
    - Security Failures in Secure Devices, Tarnovsky, BH DC 2008
    - Hacking the Smartcard Chip (TPM), Tarnovsky, BH DC 2010
- Many vendors require NDA for data sheet
- Just a sampling of what's available for embedded systems
Product / Vendor Resources 2

• Altera (Intel)
  – Secure programmable logic (FPGA, SoC)
  – Root key storage, encrypted bitstream, glitch protection, HW crypto

• ARM TrustZone
  – Security extensions/kernel added to ARM architecture
  – Hardware-enforced separation
  – Open source reference implementation
  – www.arm.com/products/security-on-arm/trustzone
Product / Vendor Resources 3

- **Atmel (now Microchip)**
  - CryptoAuthentication, TPM, CryptoRF, CryptoMemory
  - ATECC508A AWS IoT Secure Provisioning Platform

- **Broadcom**
  - Secure Applications Processors (ARM + TPM)
  - BCM5880, BCM5882, BCM5892, BCM5830x family

- **Cypress**
  - Secure MCUs/PSoc (HW crypto, WiFi security features)
  - SecureNAND Flash Memory (Block protection capabilities)
Product / Vendor Resources 4

- **Infineon**
  - OPTIGA family (Trust, TPM, Mobile)
  - Authentication, secure MCUs

- **Macronix**
  - Password-protected SPI Flash memory

- **Maxim**
  - Authentication, secure MCUs (DeepCover), secure memory/managers

- **Mentor**
  - Nucleus SafetyCert RTOS (Real Time Operating System)
  - Designed to meet many safety/security/regulatory requirements
Product / Vendor Resources 5

- **Microchip**
  - CEC1302 Crypto Embedded Controller (ARM Cortex-M4)
  - PIC Microcontrollers w/ Cryptographic Engines, CRC Scan
  - [www.microchip.com/design-centers/embedded-security](http://www.microchip.com/design-centers/embedded-security)

- **Microsemi**
  - Secure FPGAs (root of trust, on-chip cryptographic support)
  - SmartFusion2 SoC (ARM Cortex-M3), IGLOO2

- **NXP (Freescale)**
  - Kinetis K8x Secure MCU family (ARM Cortex-M4)
  - On-the-fly AES decryption/execution from external Flash, boot ROM for encrypted FW updates, HW crypto, tamper detection (temperature, voltage, clock)
Product / Vendor Resources 6

- **Qualcomm**
  - Snapdragon
  - Secure boot, trusted execution environment, HW crypto, authentication

- **Renesas**
  - Secure MCUs (RS-4, AE-5)

- **Samsung**
  - NAND Flash w/ serial interface, inline encryption/decryption
  - [http://csrc.nist.gov/groups/STM/cmvp/documents/140-1/140sp/140sp2380.pdf](http://csrc.nist.gov/groups/STM/cmvp/documents/140-1/140sp/140sp2380.pdf)
Product / Vendor Resources 7

• STMicroelectronics
  – ST23, ST31, ST32, ST33 Secure MCU families
  – STSAFE-A authentication

• Swissbit
  – Secure uSD cards w/ encrypted Flash memory
  – swissbit.com/products/security-products/overview/security-products-overview/

• Texas Instruments
  – Secure MCUs, HW crypto, protected memory regions
Product / Vendor Resources 8

• Xilinx
  – Secure programmable logic (FPGA)
  – Root key storage, encrypted bitstream, HW crypto, anti-tamper, DPA countermeasures
  – www.xilinx.com/products/technology/design-security.html

• Zilog (IXYS)
  – eZ80F91 MCU w/ TCP/IP stack & embedded firewall (ZGATE)
  – www.zilog.com/ZGATE
CHIPSEC: Platform Security Assessment Framework
- Test suite for analyzing security of PC platforms (HW, system firmware, platform components)
- https://github.com/chipsec/chipsec

SparkFun CryptoShield
- Open source hardware security reference/experimentation shield for Arduino and compatible
- Real-time clock, TPM, encrypted EEPROM, authentication chips
- www.sparkfun.com/products/13183

CrypTech Alpha
- Open source Hardware Security Module (HSM) reference design
- Cryptographic engine and key storage (ARM + FPGA)
- https://cryptech.is/
What Now?

- Learn from history/prior attacks
- Proactive security means safer products for all
  - Invest in proper design from the beginning
  - Don't wait for legislation before taking responsibility
  - Allocate time for white/black box product security analysis/testing
  - Bug bounty programs, accept/reward outside discoveries
  - Enable security by default
Thanks for your time!