ELC 2013 Security: Best Practices for Embedded Systems

John Mehaffey

Senior System Architect

Embedded Systems Division

mentor.com/embedded

About Me

- Mobilinux Architect for MontaVista
- Automotive Architect for MontaVista
- Security Lead for GENIVI System Infrastructure Expert Group
- Senior System Architect for Mentor Embedded

Overview

- Embedded system, not IT system
  - Physical security is NOT possible
    - You do not own the device!
  - It’s not Fort Knox
    - If you ARE protecting FK (e.g. ATM), you need to go through MUCH more than is presented here
      - Common Criteria
- Hackers are generally lazy
  - Don’t be “Low Hanging Fruit”

You Don’t Have to Outrun the Bear!

Black Hats

- Script Kiddies
- Opportunists
  - Insiders
  - Bots
  - Grudge hackers
- Organized
  - “Hacktivist”
  - Organized Crime
  - Nation States (Government)

Design Best Practices

- Root of trust
  - Secure boot
  - OTP
  - Hardware Assisted Security
- Harden your hardware
  - Reduce Electromagnetic Emissions
  - Limit access to Hardware
    - Potting
  - Hardware crypto module
  - Be wary of GPLv3
    - Update mechanism may need to accept unverified code
    - OTP “modified” flag inside crypto module

Design Best Practices (continued)

- Linux Security Modules
  - Design it in from the start
  - SELinux
    - Most Complete, hardest to configure
  - AppArmor
    - Path Based
  - SMACK
    - “Simplified”
  - Many more …
- Plan for field updates
  - Push vs Pull

Protect your Data

- Data at rest
  - Ecryptfs
- Data in motion
  - SSL, TLS
  - IPsec

Development Best Practices

- Code Reviews
- Comments!
- Coding Standards
- Simplicity
  - “Constructs in programming languages that are difficult to use properly can be a large source of vulnerabilities”
- Randomize!
  - PAX patches (ASLR, etc)
  - Keep your entropy up
    - Enroll your IRQs
    - Haveged

Static Analysis Tools

- Compiler
  - Wformat, Wformat-security, Wall, Werror, Wpedantic
  - fstack-protector, fstack-protector-all
  - D_FORTIFY_SOURCE=2
- http://en.wikipedia.org/wiki/List_of_tools_for_static_code_analysis
  - Lint, Jlint, Splint, PyLint, …
  - Abash, Boon, Clang, Oink, RATS, UNO, Yasco, …
  - Black Duck, Coverity, Klocwork, Rational, …

Testing Best Practices

- QA Plan
- Test error paths!
- Coverage analysis
  - gcov

Dynamic Analysis Tools

- Dmalloc, Electric Fence, mtrace, mpatrol, …
- Valgrind, …
- Avalanche, Glassbox, gperftools, jrat, …
- Purify, Rational, …

Pre-Deployment Best Practices

- Close your ports
- Limit access as much as possible
- Don’t assume obscurity will protect you
- Run hacker tools
  - nmap
  - Metasploit/Armitage
  - John the Ripper
  - ……..
- Hire a professional for evaluation!
  - Penetration test

Obscurity is not a defense

This month (Feb 2013), security researchers found a critical vulnerability in the Tridium Niagara Industrial control system, allowing hackers to obtain administrator logins and passwords. The system is widely used by the military, hospitals and factories to control surveillance, alarms, door locks, …

Last year, the company said it believed attacks on its systems were
unlikely because the systems were obscure and hackers
.

Post-Deployment Best Practices

- Perfection is not possible
  - Follow the lists
  - Deploy fixes

The Attacks

- Timing attacks
- Fault Attacks
- Power Analysis
- Port Scans
- DoS
- Escalation of privilege
  - rootkits

The Defense

- Timing attacks
  - Randomize
- Fault Attacks
  - Code review, test
- Power Analysis
  - Randomize inside Crypto module
- Port Scans
  - Close them
- DoS
  - Limit resource usage (cgroups)
- Escalation of Privilege
  - MAC