HW protections to counter Cyber SW attacks

Jean-Luc DANGER
D.E. Télécom ParisTech
Expert scientifique secure-IC
How SW attacks work

- **Step 1: Identifying vulnerabilities**
  - Human errors, incorrect configurations, bugs, latent problems in software (lack of arguments verification, untested code branches, race conditions, etc.)
  - Methodology
    - statistical analysis
    - “fuzzing” used to crash the system.

- **Step 2: Exploiting vulnerabilities**
  - Execute remote code
    - Give more rights (privilege escalation),
    - Have the system execute arbitrary codes, etc.
  - Methodology
    - Use a debugger and see if injected data can create an exploitable state
Exploitation by modifying the control flow

- **Code injection**
  - If the stack or heap overflowed: the return address is corrupted to jump to the payload

- **Code reuse**
  - Ret2libC: Consecutive return address to the LibC functions
  - Return Oriented Programming (ROP): Consecutive return addresses to any executable sequences (Gadgets)
What does exploitation look like?

- Expected Control Flow Graph
Cyber Attack
What does exploitation look like?

- “Illegal" program behavior – exploitation
  - Code injection and control hijaking

Diagram:

- Code section
- Data section
Cyber Attack
What does exploitation look like?

- “Illegal" program behavior – exploitation
  - Code reuse
State-of-the-art protections : SW

- Prevention and/or detection tools
  - Antivirus
  - Obfuscation
  - Integrity check of the computation (Control Flow Integrity)
  - Integrity check of the stack (canaris)
  - Address Space Layout Randomization (ASLR)
  - Virtualization
  - Honeypot
  - Tainting
  - …

A SW protection cannot guarantee a 100% security level
State-of-the-art protections : SW+HW

- **Insulation**
  - Memory Management Unit (MMU)
  - Support for virtualization (VT-x, AMD-V)
  - Insulation zones : NX bit, XD (eXecute Disable), W xor X
  - Trusted Execution Environment (TEE)
    - ARM TrustZone
    - Intel Software Guard Extensions (SGX) enclaves

All these protections need care at SW configuration!
Protection 100 % HW

- **Advantage:**
  - Root-of-Trust: a priori not flawed – and anyway cannot be exploited
  - Can be very fast
  - Can detect 0-day attack

- **Example of potential full HW protection:**
  - Control Flow Integrity performed by HW
  - Shadow stack: check the return address has not been flawed

**Questions:**
- Level of intrusivity ?
- Complexity ? extra code ?
- Robust against fault injection attack ?
CFI is not enough: Shadow stack necessity

• Shadow stack will be in future Intel CPUs
• Principle described in Article « Defending Embedded Systems Against Control Flow Attacks », by Aurélien Francillon, Daniele Perito and Claude Castelluccia @ SecuCode ’09

The CFI is OK but not the return address
Fault injection attack

The integrity of each basic block needs to be verified

Basic block

This attack can be remote "cyber": Row Hammer attack

Code section
HCODE : Control Flow + code integrity

Basic Block  ➔ Hash values
CFI  ➔ Valid destinations

CPU

BB

Hash codes
Valid destinations
metadata

HCIDE HW module

alarm
Architectural modification

Processor

Interrupt

PC

 Opcode

HCODE Hardware Module

data Cache

Inst. Cache

AMBA Ctrl

AMBA BUS

code

Metadata
Conclusions

- Many SW attacks can be twarted by HW "Root of Trust"
- HW protections
  - Cyber attacks :
    - Shadow stack
    - CFI
  - Fault Attack :
    - BB Integrity Check
- Few Performance degradation
  - Depends on cache miss
- x2 code size max
THANK YOU FOR YOUR ATTENTION!