Embedded Security

Tales from the front lines
About Me

• George Hotz (geohot)
• No formal education
• 2007
• Make it rain?
Ethics/Legality

• Jailbreak
• “Hacker”
• I paid for it, it’s mine
• DMCA applies?
  • Whole system is access-control?
  • If true, that’s just wrong
• “effective measures”
A Quick Primer on “Embedded” Security
“Embedded”?  

- Phones  
- Video Game consoles  
- Routers  
- Car ECUs  
- iPads  
- SIM cards  
- Not “Computers”
Security from the hardware

- Boot chain
- Secure BootROM
  - ROM is ROM
- Signatures
- Root cert in the hardware
Breaking It

• Startup/Run (2 exploits)
• Exploits
  • Buffer overflows (stack and heap)
• Failure to check
How a simple math problem cost me a 6 figure job
Nokia 1661

- GSM Phone -- $20
- Subsidized by T-Mobile
- Big endian ARM ASIC, DCT-4+
- Nokia has non standard security
1661: Initial Code Exec

- FBUS/MBUS flasher (not USB)
- Encryption isn’t security
- CBC cleverness
- 1 instruction
- Runtime code exec
- Halfway there?
1661: Dumping the BootROM

• No data fetch
• Jump into it
• Timer cleverness
• State transform
• THUMB/ARM
• No exploits
1661: Carrier Locking

- Lockstate data is signed with RSA
- Unlock code is salted and SHA-ed
  - And checked on startup
- 12-digits long
- Brute force?
  - GPGPU
Bleichenbacher

- Attack on low exponent RSA (d=3)
- $c^3 \mod n$
- $c^3 < n$
- $m^{1/3} = c$
- Control first $0x80/3 = 0x2A$ bytes
- Used in IPSF
1661: RSA

- First ~0x10 is checked
- Last 0x14 is the SHA1 hash
- No exploit, right?
Wrong
1661: The Math Problem

- Find $c$ such that I control start and end digits of $c^3 = m$
- Start digits is easy. Take $m^{(1/3)}$
- End digits is harder, $n$ time brute
  - I don’t know math
- $(a+b)^3 = a^3 + 3a^2b + 3ab^2 + b^3$
iPhone
A Case Study
iPhone: Boot Chain

- SecureROM
- LLB
- iBoot
- Kernel
- Applications

- BootROM
- Bootloader
- Baseband
iPhone: Exploits 2g/ipt/3g

- “Pwned” for life
- Buffer overflow in bootrom runtime code
- No check on startup!
iPhone: Other exploits

- 0x24000
- iBoot environment heap overflow
- blackra1n
- Unreleased exploit
iPhone: No Downgrading

• iBSS/iBEC/LLB/iBoot/Kernel are unique
• Heard of a replay attack?
iPhone: Baseband

- Hardware exploit (fakeblank)
- RSA exploit
- Various stack buffer overflows
- AT+XEMN heap overflow (blacksn0w)
Theoretical

- Inputs and outputs
- Shorter is better
- Why so generic? (lol @ TIFF)
PLAYSTATION 3
PS3: INITIAL

• Only unhacked console of the 7th generation

• OtherOS!

• Cell processor

• Security is “well done”

• Spent 3 weeks in Cambridge exploring the hypervisor
PS3: BOOTCHAIN

- PPU/SPU
- asecure_loader -> lv0
- metldr -> lv1ldr -> lv1
- metldr -> lv2ldr -> lv2
- metldr -> applldr -> applications
PS3: EXPLOIT

• There isn’t one
  • Well if there is, I’m not clever enough to find it
• Inputs/Outputs
  • So I made one
CODING ASSUMPTIONS

- volatile int i = 1;
- i++;
- printf("%d",i);
CODING ASSUMPTIONS

- volatile int i = 1;
- i++;
- printf("\%d", i);
- Single Threaded and Cacheless
  - Write i = 1
  - Read i
- *Write i = 2
  - Read i
PS3: VIOLATING ASSUMPTIONS

• HTAB

• Allocate/Map/Deallocate

• Glitch!
  
  • Go ahead, encrypt and add ECC to your memory

• Cache writeback

• Strap up
PS3: A WHOLE NEW WORLD

- Dumped the RAM
- In 3 years, no one outside the company had seen this code
- Yet it’s in 33.5 million peoples houses
- Kid in a candy store
CRYPTO ENGINES

- iPhone has one, PSP has one
- Decryption oracles
- AES
- Can’t get it, but can use it
PS3: CRYPTO

- All crypto is done in SPUs
- SPU isolation mode
- Yet the PPU is in charge
- So really, they are oracles
- So much potential for good security
PS3: THE FLAW

• I can use the oracles
• In fact, you don’t even need the exploit
• But metldr is system unique
• Assume your system will be compromised
• Write once registers to “map” out
QUESTIONS
and maybe answers