

How to brick my Wii

Homebrew auf der Wii

Alexander Paßfall
<alex@tty23.net>

UnFUG WS 09/10
Hochschule Furtwangen

5. November 2009

Content

- Wii Basics
- Hacks
- Homebrew
- Demo?



Wii Basics



Hardware

- “Overclocked Gamecube”
- IBM Power PC 750CL “Broadway” @ 729Mhz
- ATI “Hollywood” GPU+DSP @ 243Mhz
- 24MB 1T-SRAM (MEM1) + 64MB GDDR3 DRAM (MEM2)
- 512MB NAND Flash
- Modified DVD reader (DL)



Security System

- 2 processors
- Broadway (PPC): Fast + insecure
 - No OS! Games run on “bare metal”
- Hollywood (ATI): Graphics, peripherals, memory, “IO Bridge”
 - IO Bridge: NEC ARM926 SoC: “Starlet”
 - Custom microkernel OS (“IOS”) by BroadOn
 - Drivers and stuff:
 - Security & Software DRM
 - DVD, SD, WiFi, USB, ...
 - HTTP, SMTP, SSL, ...
 - “Always on”
 - All code is signed & authenticated by IOS
 - IOS hidden behind APIs



Boot process

- boot0: 1.5k bootloader mask ROM in Hollywood
 - Reads first 48 pages of Flash (boot1)
 - Decrypt + hash (SHA1)
 - Compare hash with value in OTP memory
 - Run boot1
- boot1: 2nd-stage bootloader
 - Runs in Mem1
 - initializes Mem2
 - loads, decrypts, verifies RSA signature of boot2
- boot2: 3rd-stage (main) bootloader (mini IOS)
 - Verifies & runs IOS
- IOS
 - read from flash filesystem
 - ARM code running on starlet
- Menu: PPC code read from filesystem, pushed to Broadway



Boot process

- boot0: 1.5k bootloader mask ROM in Hollywood
 - Reads first 48 pages of Flash (boot1)
 - Decrypt + hash (SHA1)
 - Compare hash with value in OTP memory
 - Run boot1
- boot1: 2nd-stage bootloader
 - Runs in Mem1
 - initializes Mem2
 - loads, decrypts, verifies RSA signature of boot2
- boot2: 3rd-stage (main) bootloader (mini IOS)
 - Verifies & runs IOS
- IOS
 - read from flash filesystem
 - ARM code running on starlet
- Menu: PPC code read from filesystem, pushed to Broadway



Boot process

- boot0: 1.5k bootloader mask ROM in Hollywood
 - Reads first 48 pages of Flash (boot1)
 - Decrypt + hash (SHA1)
 - Compare hash with value in OTP memory
 - Run boot1
- boot1: 2nd-stage bootloader
 - Runs in Mem1
 - initializes Mem2
 - loads, decrypts, verifies RSA signature of boot2
- boot2: 3rd-stage (main) bootloader (mini IOS)
 - Verifies & runs IOS
- IOS
 - read from flash filesystem
 - ARM code running on starlet
- Menu: PPC code read from filesystem, pushed to Broadway



Boot process

- boot0: 1.5k bootloader mask ROM in Hollywood
 - Reads first 48 pages of Flash (boot1)
 - Decrypt + hash (SHA1)
 - Compare hash with value in OTP memory
 - Run boot1
- boot1: 2nd-stage bootloader
 - Runs in Mem1
 - initializes Mem2
 - loads, decrypts, verifies RSA signature of boot2
- boot2: 3rd-stage (main) bootloader (mini IOS)
 - Verifies & runs IOS
- IOS
 - read from flash filesystem
 - ARM code running on starlet
- Menu: PPC code read from filesystem, pushed to Broadway



Boot process

- boot0: 1.5k bootloader mask ROM in Hollywood
 - Reads first 48 pages of Flash (boot1)
 - Decrypt + hash (SHA1)
 - Compare hash with value in OTP memory
 - Run boot1
- boot1: 2nd-stage bootloader
 - Runs in Mem1
 - initializes Mem2
 - loads, decrypts, verifies RSA signature of boot2
- boot2: 3rd-stage (main) bootloader (mini IOS)
 - Verifies & runs IOS
- IOS
 - read from flash filesystem
 - ARM code running on starlet
- Menu: PPC code read from filesystem, pushed to Broadway



Software

- Channels, Games, System software are “titles”
- Identified by TitleID
- TMD: Title MetaData
 - Information about content
 - SHA1 hashes, permissions, group IDs, region locking
- eTicket: Your licence to use the title
 - encrypted AES key (master key in OTP ROM / hard to extract)
 - optional time limits
- TMD + eTicket signed using RSA-2048
- Title content encrypted using AES + hashed using SHA1
 - hash tree structure



IOS

- Custom micro-kernel OS by BroadOn (California)
- talks to Broadway via IPC interface
- high-level network API
- decryption / authentication of Broadway's code
- POSIX-like FS permissions (titles = users / vendors = groups)
 - Hides system files from Broadway
- Modules as isolated userspace processes
- Kernel in MEM1, userspace in top 12MB of MEM2 (no access from Broadway)



Hacks



GameCube Mode

- GameCube software is totally unsigned, but runs in a sandbox
- DVD drive similar to GameCube's
 - Outsourced to Matshita
 - mod chips easy portable to Wii
- GameCube homebrew possible
 - Sandboxed: no IOS, no Wii features
- Wii always boots first into native mode, then reboots into GameCube mode
- GameCube mode uses first 16MB of MEM2



Tweezer Attack

- Upper 48MB not cleared on reboot to GameCube mode
- Protected by hardware register
 - Modify address lines of DRAM chip
 - Move 16MB “window” throughout DRAM
 - Dump entire 64MB
- Content:
 - IOS
 - **Keystore with all the Keys!**

Tweezer Attack

- Upper 48MB not cleared on reboot to GameCube mode
- Protected by hardware register
 - Modify address lines of DRAM chip
 - Move 16MB “window” throughout DRAM
 - Dump entire 64MB
- Content:
 - IOS
 - Keystore with all the Keys!

Tweezer Attack

- Upper 48MB not cleared on reboot to GameCube mode
- Protected by hardware register
 - Modify address lines of DRAM chip
 - Move 16MB “window” throughout DRAM
 - Dump entire 64MB
- Content:
 - IOS
 - **Keystore with all the Keys!**



Keys

- Per-console keys
 - ECC private key
 - ECC public certificate
 - NAND AES key
 - NAND HMAC key
- Global keys
 - Common key 0
 - SD key
 - Root certificate
 - New common key 1 (Korean)

Signatures

- All RSA signature comparison is done by one function
- ES_VerifySign
 - Hardware SHA1
 - Software RSA
- TMD contains SHA1 signed by Nintendo
- Real TMD hash is calculated, then both are compared

Nintendo-RSA

Looks kinda strange..

```

1C 28  ADDS  R0, R5, #0           ; R0 = signature_end
38 14  SUBS  R0, #20              ; R0 -= 20
99 02  LDR  R1, [SP, #SHA1_calc] ; R1 = SHA-1
22 14  MOVS R2, #20             ; R2 = 20
4B 0F  LDR  R3, =(strncmp+1)
47 98  BLX  R3                  ; strncmp(SHA1_sig, SHA1_in, 20)

```

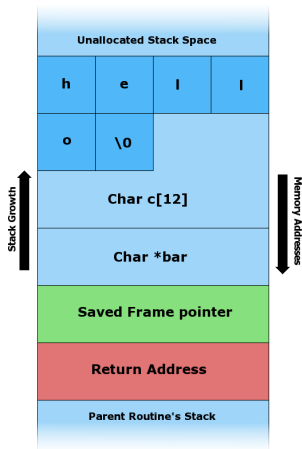
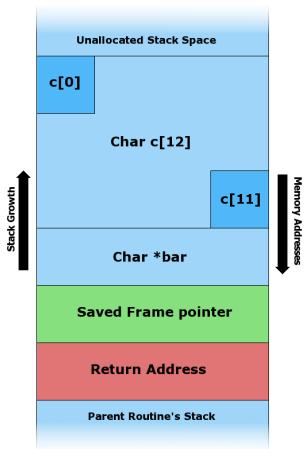
Impact

- We can somehow sign/install everything we want:
- Unsigned games
- Unsigned System Menu
- Unsigned IOSes
- Unsigned boot2 (fixed somewhere in 2008)



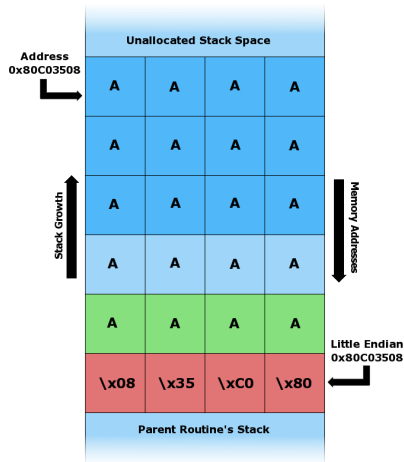
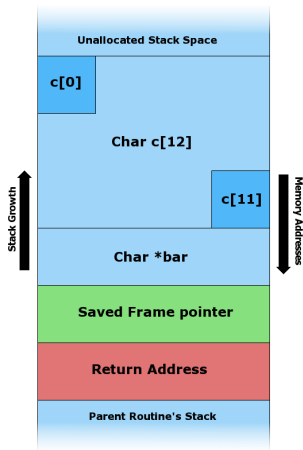
Demo

Stack Smashing





Stack Smashing



Twilight Hack

- Savegames on SD card are signed with the console's private key
- We can extract the keys, so we can sign any savegame
- Exploit a stack buffer overflow in The Legend of Zelda: Twilight Princess (Name of horse)
- Load an ELF-Loader
- Loader reads an ELF-executable from an SD card



Homebrew

HBC

- Home Brew Channel
- Launcher for multiple homebrew apps

BootMii

- Custom boot2
- Recovery System
- Runs as custom IOS, too

DVDx

- Wii normally rejects non-Wii discs
- Drive firmware has hidden DVD Video player functions
- Blocked by IOS..
- unless you set a magic bit in TMD
- Homebrew can play DVD Videos
- DVD-Rs look a lot like DVD Video discs..
- Warez loader

DVDx

- Wii normally rejects non-Wii discs
- Drive firmware has hidden DVD Video player functions
- Blocked by IOS..
- unless you set a magic bit in TMD
- Homebrew can play DVD Videos
- DVD-Rs look a lot like DVD Video discs..
- Warez loader

DVDx

- Wii normally rejects non-Wii discs
- Drive firmware has hidden DVD Video player functions
- Blocked by IOS..
- unless you set a magic bit in TMD
- Homebrew can play DVD Videos
- DVD-Rs look a lot like DVD Video discs..
- Warez loader

DVDx

- Wii normally rejects non-Wii discs
- Drive firmware has hidden DVD Video player functions
- Blocked by IOS..
- unless you set a magic bit in TMD
- Homebrew can play DVD Videos
- DVD-Rs look a lot like DVD Video discs..
- Warez loader

DVDx

- Wii normally rejects non-Wii discs
- Drive firmware has hidden DVD Video player functions
- Blocked by IOS..
- unless you set a magic bit in TMD
- Homebrew can play DVD Videos
- DVD-Rs look a lot like DVD Video discs..
- Warez loader

Other

- mplayer
- ScummVM
- Custom Games
- Linux
- ...



Fragen?