Google TV Or: How I Learned to Stop Worrying and Exploit Secure Boot
GTVHacker: The Team

- GTVHacker is a group of 6 hackers with individual skill sets who work together to unlock Google TV devices.
- Our primary goal is to bypass hardware and software restrictions to allow for unsigned kernels to be loaded and used.
- To date the team has released multiple methods for unlocking Google TV devices.
- The GTVHacker team won a $500 bounty for being the first to root the Google TV.
- We hack things because we believe in open and free hardware. Our current target just happens to be the Google TV.

http://DC21.GTVHacker.com
Members

Mike Baker ([mbm])- Firmware developer and co-founder of OpenWRT

Hans Nielsen (AgentHH)– Senior Security Consultant at Matasano

CJ Heres (cj_000) – IT Systems Manager

gynophase – He's running that big ole DEFCON CTF right now

Tom Dwenger (tdweng)– Excellent with APK reversing and anything Java

Amir Etemadieh (Zenofex) – Research Scientist at Accuvant LABS, founded GTVHacker

http://DC21.GTVHacker.com
What's the Google TV?

- Google TV is a platform that:
  - Bridges the gap between your TV and an Android device.
  - Creates an overlay on television stream and also contains an IR transmitter to transmit to media center devices (cable box, TV, sound system).
  - Receives over-the-air updates automatically from OEM manufacturers.
  - Contains a forked version of Chrome with all plugins and extensions disabled.
  - Was originally released without the Android Market available but was eventually updated to include it.
  - Provides a built-in Flash Player, however most content providers block the Google TV.

http://DC21.GTVHacker.com
Why We Hack It

Just a few reasons why we targeted the platform:

• Locked bootloader
• Heavily restricted kernel preventing user modifications
• Generation 1 EOL
• Crippled Flash Player

In short, the Google TV devices are locked down and crippled by their limitations. Our goal is to change that.
Last Year

They released devices...  We hacked them all.
Let's make this quick so we can get to the exploits!

http://DC21.GTVHacker.com
Generation 1 Hardware

Logitech Revue

NSZ-GT1

NSZ-[24-46]GT1

Extremely limited number of devices compared to second generation.
First generation has been discontinued.

http://DC21.GTVHacker.com
Recap of Generation 1 Exploits

- Logitech Revue
  - Root UART
  - /dev/devmem (Dan Rosenberg)

- Sony NS[X|Z]-[24-46]GT1
  - Downgrade nodev bug
  - Recovery LCE
  - kexec as module
  - Unsigned Kernels

http://DC21.GTVHacker.com
Along the way: Chrome Flash Player Modification

Hulu and other sites check the Flash Player version string on the box, preventing access.

From:

```
00969F5269 6E 3A 00 47 54 56 20 31 30 2C 31 2C 31 30 37 2C 31 39 31 00 50 6C 75 67 49 6E 00 35 2E 31 00 25 32
```

To:

```
00969F5269 6E 3A 00 41 54 56 20 31 30 2C 32 2C 31 30 38 2C 31 39 31 00 50 6C 75 67 49 6E 00 35 2E 31 00 25 32
```

This simple change teamed with modifying the browser user-agent results in a Content Block Bypass on all blocked sites.

http://DC21.GTVHacker.com
Logitech's Secret Message to Us

“@gtvhackers congratulations if your [sic] reading this please post a note on your forum to let me know ;)

http://DC21.GTVHacker.com
Boxee Box

We disclosed an exploit for Boxee at last year's DefCon

- Software LCE
- Hardware Root UART (under some VIA's)
- Spawned Boxee+ Community
- Modifications based off our root that extend the life and functionality of the Boxee Box
- 308,128 Views since December, 2012
- STILL VULNERABLE :)

TL;DR We dropped an exploit at DEFCON 20, the community responded. Keep up the awesome work Boxee community.

http://DC21.GTVHacker.com
The Next Generation...

http://DC21.GTVHacker.com
Generation 2 Hardware

- LG U+
- Asus Cube
- LG 47/55G2 & G3
- Netgear Prime
- Sony NSZ-GS7/GS8
- Hisense Pulse
- Vizio Co-Star

Similar hardware design throughout most of the generation

http://DC21.GTVHacker.com
Generation 2

- Marvell 88DE3100 based
- ARM – Dual 1.2GHz processors
- Dubbed the “Armada 1500”
- On-die Crypto processor, separate memory
- Secure Boot from ROM via RSA and AES

http://DC21.GTVHacker.com
Marvell Armada 1500 (88DE3100)

- Secure Boot ROM
- SPI x2
- SDIO x2
- UART x3
- TWSI x4
- GPIO and PWM
- SATA 3.0
- Fast Ethernet PHY
- Fast Ethernet w/SMII & TMII
- USB 2.0 Host
- USB 2.0 OTG
- NAND/NOR Flash
- 32-bit DDR3 800MHz
- JTAG Debug Port w/ARM Core Sight
- HDMI TX v1.4
- Dual Link LVDS Tx
- 4 Video DACs
- Stereo Audio DACs
- 7.1 I²S and S/PDIF Tx
- 7.1 I²S and S/PDIF Rx
- Digital Video Input
- Serial Transport Input x2

Dual Core PJ4b-SMP
Super-scalar 2.61 DMIPS/MHz/Core
ARM v6/7

32 I/D L1 Cache
L0 Cache
32 I/D L1 Cache
ARM Neon with FPU v3.0
WMM x2

512 KB L2 Cache

Security Engine OTP, RNG, AES/(3), DES, RSA, SHA-1, MDS

TS Processor PID Section Filter De-multiplexing AES/(3) DES, Multi-2, CSS, CSA

Qdeo™ Video Post Processing

VMeta™ Video CODEC
H.264 AVC/MVC VC-1, MPEG2/4 H.263, RV8/9, VP6/8, AVS H.264HP Encode

Audio DSP

Front Panel Controller and Power Management

http://DC21.GTVHacker.com
Chain of Trust

Chain of Trust Placeholder

http://DC21.GTVHacker.com
Platform Information

- Android 3.2
  - No public vulnerabilities work
- Not a Bionic libc
  - No Android native libraries supported*
- Gen 1: Intel CE4150
  - Single Core Atom ~1.2GHz
- Gen 2: Marvell Armada 1500
  - Dual Core ARM ~1.2GHz each
- Android 4.2.2 incoming for Gen 2
  - Adds Native Libraries, Bionic libc

*Native libraries supported only for Gen 2.
Sony NSZ-GS7/GS8

- 8GB EMMC Flash
- Best remote
- Larger form factor
  - Internal PSU
  - Built in IR blasters
- $199

Same box as the GS7, but with a voice search remote

http://DC21.GTVHacker.com
Vizio Co-star

- Small form factor
- No Voice Search
- Custom Launcher
- $99 MSRP
- Updates are encrypted via Update Logic
  - Common in all Vizio devices

http://DC21.GTVHacker.com
Hisense Pulse

- 2nd Best Remote
- Launched with ADB running as root
  - Patched shortly after
- $99 MSRP
Hisense Pulse Root

- Teardown showed a root shell over UART
- `ro.debuggable=1`
- `adb root` was all it needed!
- Released a script that disabled updates and installed our Chrome Flash Modification

We'll have a select number of USB to TTL adapters available at the Q&A

http://DC21.GTVHacker.com
Netgear NeoTV Prime

- Horrible Remote
- $129 MSRP
- Two exploits
  - One real
  - One oversight

http://DC21.GTVHacker.com
Netgear NeoTV Prime Root

Prime auto-spawned a console as the root user over UART regardless of the security setting.

```bash
### force to create a console no matter what ###
on property:ro.secure=0
  start console
on property:ro.secure=1
  start console
```

Factory backdoor in the “testmode” service.

Allowed for execution of code from USB as root.

http://DC21.GTVHacker.com
GTVHacker Netgear NeoTV Prime Root Exploit

Netgear NeoTV Prime Boots

Check persist.radio.testmode.enabled?

1

Extract "test_mode.tgz" from USB drive to /tmp

Does /tmp/test_mode/test_mode.sh exist?

Yes

Run /tmp/test_mode/test_mode.sh as root

ROOT!

Install Superuser, Flash Bypass, Set persist.radio.testmode.enabled=1 and reboot

0

Is there a USB Drive?

Yes

Does ".testmode" exist, and does it contain "testmodemark"?

Yes

Set persist.radio.testmode.enabled=1 and reboot

No

Boot Normally
Asus CUBE

- Same generation 2 hardware
- Bad Remote
- $139 MSRP

http://DC21.GTVHacker.com
CubeRoot

Auto exploits and patches your Asus Cube from an App!

http://DC21.GTVHacker.com
Exploited a helper app (oplayhelper) via a world writable socket
Helper application passed un-sanitized input to the mount command resulting in LCE
  - We triggered the vulnerability from within an Android APK
  - Point, click, pwn
  - Added in Google Play Store

http://DC21.GTVHacker.com
CubeRoot

- Also patches the exploit, to prevent evil apps
- Pulled by Google – get it at GTVHacker.com
  - Downloaded ## boxes
  - Rooted ## boxes (Included 1 eng build)
  - Listed in Google Play store for 6 days
- Patched at the beginning of July
  - Took roughly 2 months
One Root to Rule Them All

http://DC21.GTVHacker.com
Magic USB

- Recall our past exploits with file system nodes and block devices?
  - In the first generation of GoogleTV devices, our original "4 usb recovery exploit" leveraged a USB device improperly not mounted "nodev"
  - That was only two very similar devices.

What about something a bit bigger?
Magic USB

- All Google TV's and some other Android devices are vulnerable.
  - Certain specific Linux boxes too!
- vold mounts NTFS partitions without "nodev"
- A little known "feature" of NTFS is that it supports Linux block / character devices

http://DC21.GTVHacker.com
Magic USB

• NTFS Drive + Block Device
  o Read / Write on any box, any partition.

• Easy root, on every single box!
  o Dump boot.img
  o Patch init.rc or default.prop to ro.secure=0
  o Write it back (as a user, no root needed)
  o Reboot, you are rooted – win!
  o Sony boxes require an additional step
OOOHHHH YEAH

http://DC21.GTVHacker.com
Hardware Mods / Exploits

Sony NSZ-GS7 with EMMC->SD and SATA Mods

http://DC21.GTVHacker.com
LG 47/55G2

• Dual Core ARM L9
  o aka LG1152

• Signed Everything
  o Even the splash!

• Our “White Whale”
  o Why spend $1K?
  o Next best thing
    ▪ Power supply and Mobo

http://DC21.GTVHacker.com
LG 47/55G2 Root

• Hardware Root!
  o EMMC Flash
    ▪ EMMC
    ▪ MMC
    ▪ SD

• All fall back to SPI mode
LG 47/55G2 Root

http://DC21.GTVHacker.com
LG 47/55G2 Root

partinfo at 0x100000.

Take the filename, count back 6 bytes and byteswap – your location.

/system is at 122,159,104

mount -text4 -o,skip=122159104 /dev/sdXX /mnt/system

http://DC21.GTVHacker.com
LG 47/55G2 Root

- Root FS is a signed squashfs image
- Init script calls: /system/vendor/bin/init_forcestripped.sh
- Mount, edit to spawn telnet, root shell over uart, or over PL2303 USB serial adapter.
- Debug agent (dongle needed) runs over UART

```bash
on boot
    # Fix system date if necessary.
    exec /system/bin/fixdate

    # TODO: remove an unnecessary comment.
    # Init osd0 for fast-boot
    setprop debug.sf.robootanimation 1
    write /proc/lg/fbdev/scrclear "0"
    start init_osd0

    # TODO: remove an unnecessary comment.
    # Init volume
    #exec /system/vendor/bin/init_volume.sh
    start init_volume

    # Sets ro.forcestripped based on build.prop or persist.gtv.forcestripped
    exec /system/vendor/bin/init_forcestripped.sh
```

http://DC21.GTVHacker.com
Sony NSZ-GS7/GS8

• Sony also uses an EMMC Flash making interfacing easier
  o Boot & system are not signed
• To gain root we rewrite /boot or /system
  o However, the RSA signed init scripts check for certain props
    ▪ EX: Check for ro.secure=0, if so, reboot
      o Since we can modify /boot we can remove the check
  o Sony also disabled dd, insmod, and some other bits via kernel calls
  o Being able to write /system and /boot you can change most restrictions at will!

http://DC21.GTVHacker.com
Sony NSZ-GS7/GS8

- SATA HDD
  - Jumpers / caps over front points
  - Add SATA connector on the back
  - Connect a HDD. Ach, it's not being detected!

But no kernel support for SATA
Now What?

We've got root but we want more.

http://DC21.GTVHacker.com
Marvell Armada 1500 Secure Boot Exploit

- Armada 1000 = 88de3010
- Armada 1500 = 88de3100

May also work on the Armada 1000
Marvell Armada 1500 Secure Boot Exploit

- Sony NSZ-GS7
- Netgear NeoTV Prime
- Vizio Co-Star
- Hisense Pulse
- Asus CUBE
- Sony NSZ-GS8
- LG U+ IPTV
- Google “Berlin”
- ZeroDesktop MiiPC
- Hisense XT780 TV
- Lenovo S31/S61 TV
- TCL MoVo
- And Others!

http://DC21.GTVHacker.com
Detailed Security Overview

1. Power On
   - Execute ROM Code
   - Load Stage 2

2. AES Decrypt and RSA verify.
   - Return decrypted Stage 2 on success.

3. Execute Stage 2.
   - Initialize EMMC.
   - Load Stage 3 to 0x0680000

4. AES Decrypt and RSA verify.
   - Return decrypted payload on success.

5. Execute Bootloader (Stage 3)
   - Load Kernel to 0x01000800

6. Load Ramdisk to 0x20000000

7. Execute Kernel

8. SHA1 Hashcheck /init

9. RSA Verify init*.rc

http://DC21.GTVHacker.com
Bootloader Messages

Start kernel at 0x01008000

Uncompressing Linux... done, booting the kernel.

http://DC21.GTVHacker.com
Android Kernel + Marvell Secure Image

00000000 414E 4452 4F49 4421 20BF 2800 0080 0001
00000010 78E0 0100 0080 0001 0000 0000 0000 0000
00000020 0001 0001 0008 0001 0000 0000 0000 0000
00000030 0000 0000 0000 0000 0000 0000 0000 0000
00000040 2063 6F6E 736F 6C65 3D20 696E 6974 3D2F
00000050 696E 6974 2072 6F6E 743D 2F64 6576 2F72
00000060 616D 3020 726F 6765 743D 7449 7465 3D65
00000070 7874 3420 7261 6D64 6973 6B5F 7369 7A65
00000080 3D31 3034 3835 3736 3030 0000 0000 0000
00000090 0000 0000 0000 0000 0000 0000 0000 0000
000000A0 0000 0000 0000 0000 0000 0000 0000 0000

******************************************************************************

00000220 0000 0000 0000 0000 0000 0000 0000 0000
00000230 0000 0000 0000 0000 0000 0000 0000 0000
00000240 0CBE 754A C641 116E 183E 3F95 AFDC 8C62
00000250 04CA 734E 0000 0000 0000 0000 0000 0000
00000260 0000 0000 0000 0000 0000 0000 0000 0000
00000270 0000 0000 0000 0000 0000 0000 0000 0000

******************************************************************************

000007E0 0000 0000 0000 0000 0000 0000 0000 0000
000007F0 0000 0000 0000 0000 0000 0000 0000 0000
00000800 8000 0000 0000 0000 0000 0000 0000 0000
00000810 BC18 7E62 2DE0 F662 15D4 D23C 9D48 1618
00000820 0000 0000 0000 0000 0000 0000 0000 0000
00000830 0000 0000 0000 0000 0000 0000 0000 0000
00000840 0200 0000 0280 0000 B670 BEF7 7233 1F87
00000850 03B0 E49C 9E22 6D1D 1127 4684 0000 0000
00000860 0000 0000 0000 0000 0000 0000 0000 0000
00000870 0000 0000 0000 0000 0000 0000 0000 0000
00000880 7A06 3B30 E9F1 064F D7F6 46FA BB40 1E5E
00000890 827A 49C5 30BA B16C DE03 6FB5 4462 CED5
000008A0 AC78 1064 E25F 6165 E7F5 1000 0000 0000
000008B0 6327 0207 9019 B472 06F2 56C9 B5C3 FFAB
000008C0 61B4 A6FD EEA4 28E1 EAA7 9364 C012 B1ED
000008D0 0F06 6937 BE81 5BDA 6442 29D0 CCE0 C01D
000008E0 E232 8070 2706 3866 8ADA 57D5 44D2 E76C
000008F0 0D9E B0DB 81BD 21A3 D630 E607 E088 0DF0
00000900 6787 8489 E056 25A7 CC52 174E 890A FA08
00000910 6999 F461 E719 1DE9 C41D DD9E F263 2025

******************************************************************************

uJ An ==> b
SN

******************************************************************************

http://DC21.GTVHacker.com
#define BOOT_MAGIC "ANDROID!"
#define BOOT_MAGIC_SIZE 8
#define BOOT_NAME_SIZE 16
#define BOOT_ARGS_SIZE 512

struct boot_img_hdr {
    unsigned char magic[BOOT_MAGIC_SIZE];
    unsigned kernel_size; /* size in bytes */
    unsigned kernel_addr; /* physical load addr */
    unsigned ramdisk_size; /* size in bytes */
    unsigned ramdisk_addr; /* physical load addr */
    unsigned second_size; /* size in bytes */
    unsigned second_addr; /* physical load addr */
    unsigned tags_addr; /* physical addr for kernel tags */
    unsigned page_size; /* flash page size we assume */
    unsigned unused[2]; /* future expansion: should be 0 */
    unsigned char name[BOOT_NAME_SIZE]; /* ascii product name */
    unsigned char cmdline[BOOT_ARGS_SIZE];
    unsigned id[8]; /* timestamp / checksum / sha1 / etc */
};

/*
 ** | boot header | 1 page |
 ** +------------------------+
 ** | kernel | n pages
 ** +------------------------+
 ** | ramdisk | m pages
 ** +------------------------+
 ** | second stage | o pages
 ** +------------------------+

http://DC21.GTVHacker.com
A Second Look

You got your Ramdisk in my Kernel!

Red = Ramdisk Size
Black = Ramdisk Load Address

http://DC21.GTVHacker.com
Secure Boot Exploit

• Note the ramdisk load address
  o Can be modified without breaking kernel signature

• Allows us to load a "ramdisk" anywhere in memory
  o Ramdisk in this case is a chunk of our own unsigned code

• Copies in our "ramdisk" at the address specified, and without any additional checks, we can run our own unsigned code
Area Of Attack (Pseudocode)

```c
// hard coded load address
kernel_load_addr = 0x01000000;

// read kernel from emmc / nand flash to memory
do_emmcread(kernel_buf, kernel_load_addr);

// some stuff to parse the header into nice names
printf("Kernel image decrypt start now");

// start to decrypt and verify, send the image to the security processor
res = LoadImage(kernel_buf, header_kernel_size);
printf("Kernel image decrypt finished");

if(res){
    printf("Verify Kernel image failed!");
    return 1;
} else{
    // copy kernelbuf
    memcpy(kernel_buf, kernel_load_addr, header_kernel_size);
}

if(ramdisk_header_size){
    do_emmcread(ramdisk_buf, headerramdisk_size);
    memcpy(ramdisk_buf, header_ramdisk_load_addr, header_ramdisk_size);
}

printf("verify Kernel image passed.");
```

http://DC21.GTVHacker.com
New Boot Flow / Memory Map

http://DC21.GTVHacker.com
Exploit Process

• GTVHacker Custom Recovery on Sony NSZ-GS7
  o Sony box has additional security
  o Append a tiny secure image that will validate
  o Normal signed kernel will do

• Add on our custom Recovery + Kernel (w/ ramdisk)
  o Change Ramdisk size to match our new "ramdisk"

• Set Ramdisk Load Address:
  ▪ 0x1008000 – Size of Signed Kernel

• Our custom Recovery ends up at 0x1008000, and boots!
Exploit Process

Exploit Image Placeholder

http://DC21.GTVHacker.com
U-Boot

- We can also trigger the exploit and run uboot
- ASUS was kind enough to GPL parts, and with some patches, it runs
- Load a kernel via TFTP, Flash, or USB for development
Future Research

Areas that need more work:

• Unsigned kernels on Gen 1 (Revue) w/ NTFS exploit

http://DC21.GTVHacker.com
Demo
Thank You!

Slide Resources can be found at:

http://DC21.GTVHacker.com

WIKI: http://www.GTVHacker.com

FORUM: http://forum.GTVHacker.com

BLOG: http://blog.GTVHacker.com

IRC: irc.freenode.net #GTVHacker

Follow us on Twitter:
@GTVHacker