CROWDSTRIKE

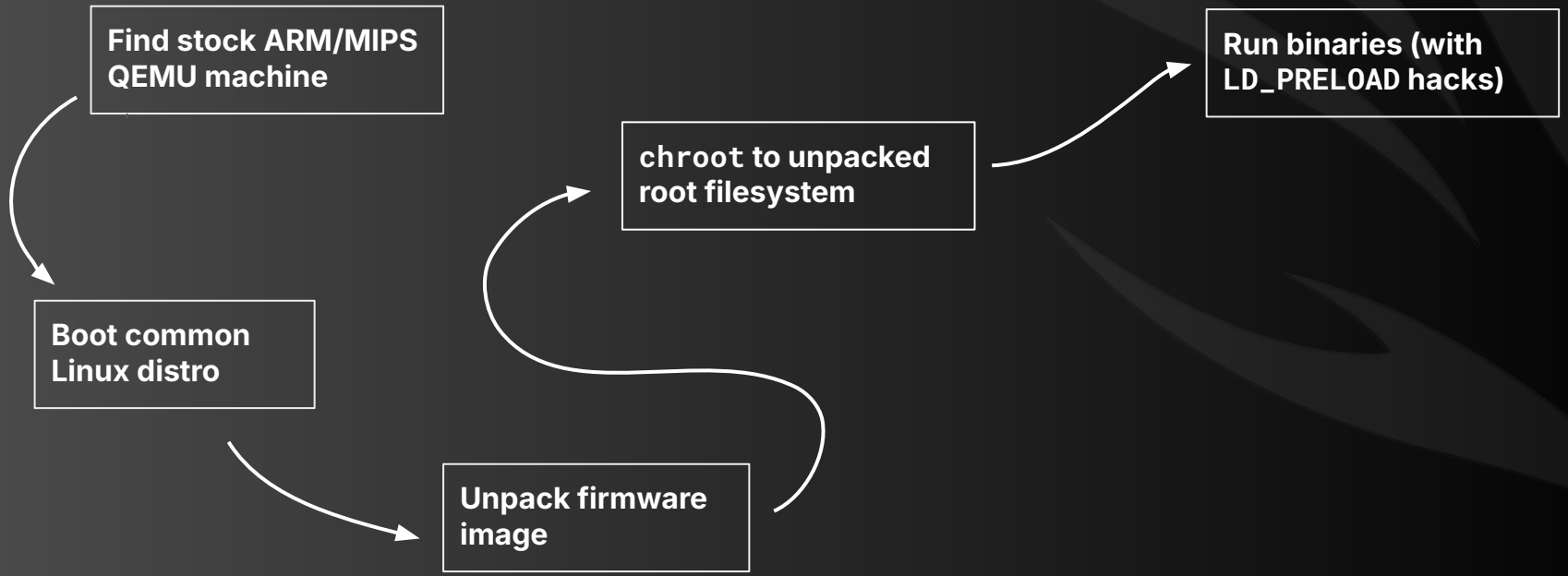


Emulating Embedded Linux Devices at Scale with Light-Touch Firmware Rehosting

® CrowdStrike, Inc. All Rights Reserved

CROWDSTRIKE

Embedded Linux Emulation Today



® CrowdStrike, Inc. All Rights Reserved

CROWDSTRIKE

Emulation Friction

❌ *Missing hardware*

❌ *Non-standard interfaces*

❌ *Holistic system design doesn't "travel"*

® CrowdStrike, Inc. All Rights Reserved

CROWDSTRIKE

What Do We Want?

Full system emulation we can use in:

→ **Vulnerability research**

→ **Honeypots**

® CrowdStrike, Inc. All Rights Reserved

Rehosting Approach

Don't touch userspace!

Looks as close as a "real" device as possible, from userspace.

Rehosting Environment

`Modified QEMU machine`

`Modified kernel`

loads...

& boots into...

`Real firmware user image`

→ Userspace can only use what the kernel exposes.

→ Custom QEMU machine + modified kernel = terrain which looks as close as a "real" device as possible.

® CrowdStrike, Inc. All Rights Reserved

CROWDSTRIKE

Where Do We Start?

What are embedded Linux systems made of?

Built on SoCs (“System-on-Chip”).

Vendors (Netgear, ASUS, TP-Link, etc) are not chip companies!

Vendors don’t build SoCs!

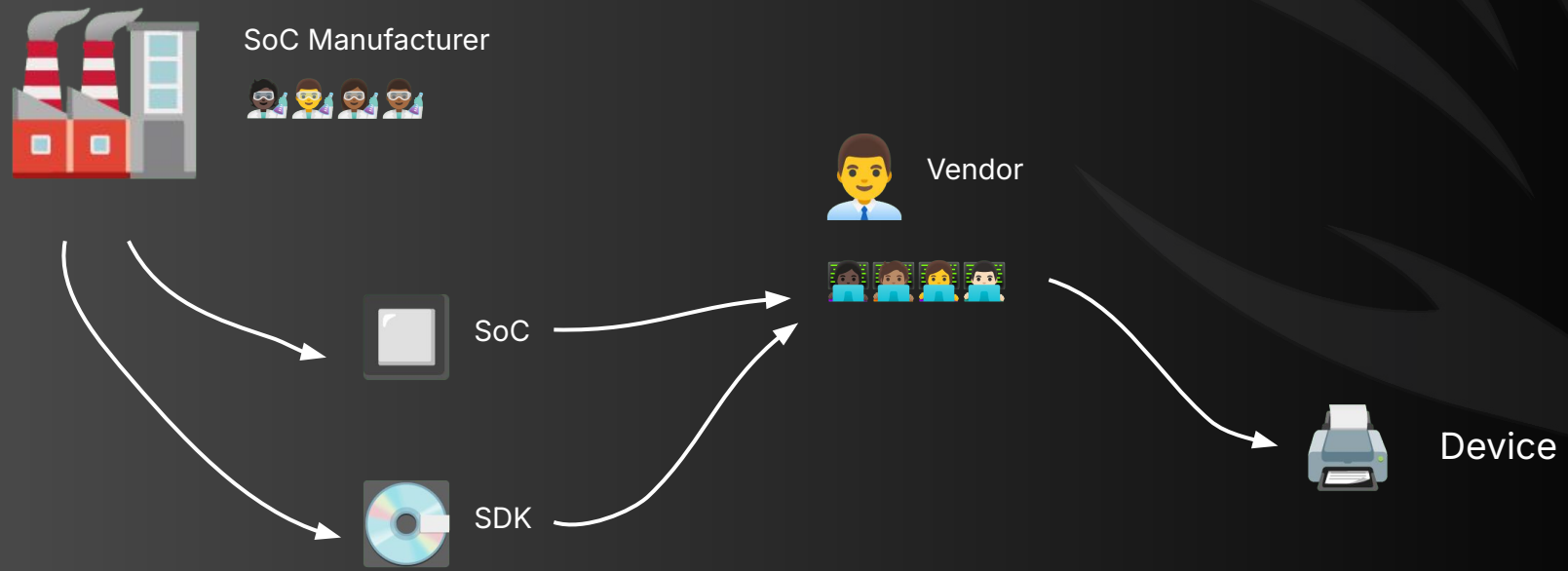
The same SoCs are used across multiple devices by different vendors.


Qualcomm
MDM9607
MDM9206
MDM9640
MDM9650
MDM9655
MDM9207
MDM9628
MDM9230
MDM9250
MDM9255
MDM9628
MDM9230
MDM9250
MDM9255
MDM9628
MDM9230
MDM9250
MDM9255

CROWDSTRIKE

SoC and Load

The supply chain looks something like:



® CrowdStrike, Inc. All Rights Reserved

CROWDSTRIKE

So What?

→ Shift perspective

→ Away from device, towards the SoC/SDK

→ Take firmware from any device based on that SoC family and run it

® CrowdStrike, Inc. All Rights Reserved

CROWDSTRIKE

Info to Lean On

✅ QEMU machines

✅ Linux kernel

✅ Firmware images

✅ RE tools

✅ GPL code

✅ Compiled kernels

✅ OSS distros (OpenWRT, etc)

❌ Datasheets

❌ Source code for many drivers, binaries, libraries

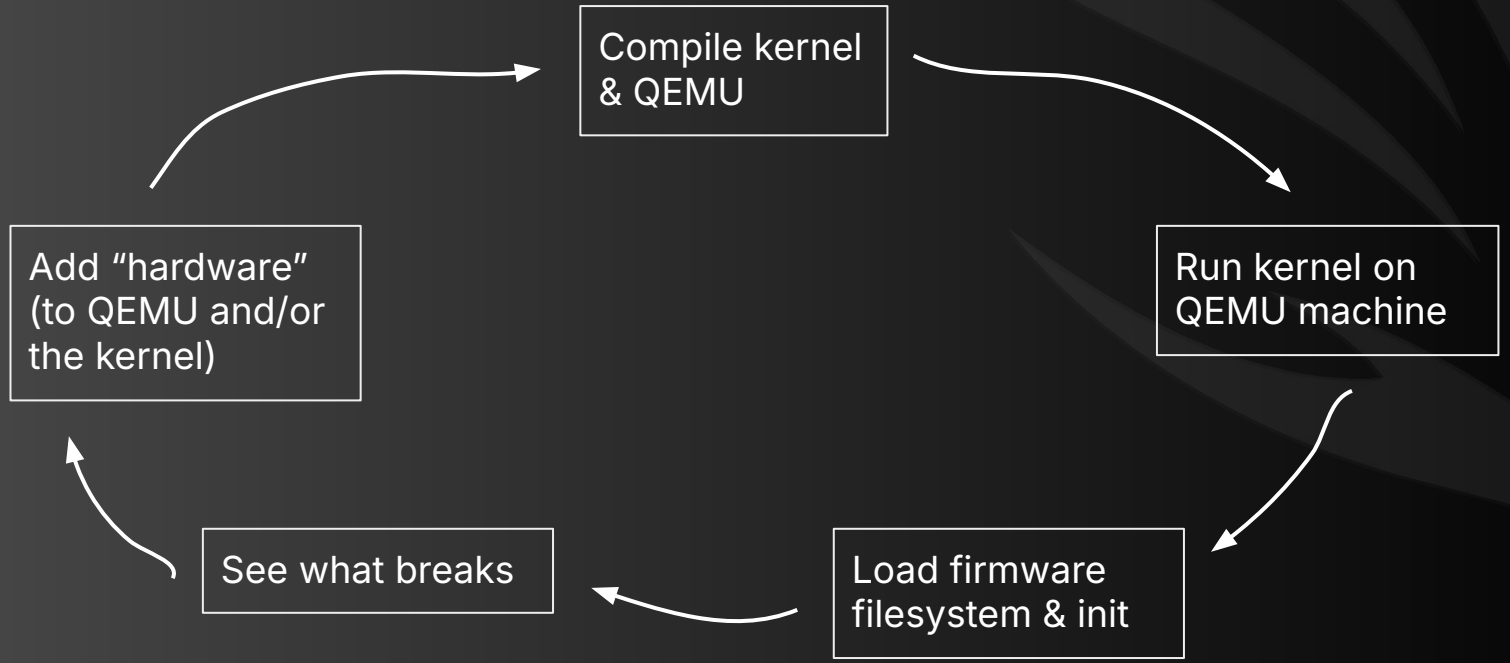
❌ Documentation

❌ **Unlimited time, energy, money**

® CrowdStrike, Inc. All Rights Reserved

The Plan

*This is enough to try **something**.*



® CrowdStrike, Inc. All Rights Reserved

CROWDSTRIKE



ASUS RT-AC87U

Older ASUS router.

Based around the BCM4709.

© CrowdStrike, Inc. All Rights Reserved

CROWDSTRIKE

BCM4709 Data

→ Broadcom

→ Cortex A9

→ BCM4707/4708/4709 family.

→ Five integrated 10/100/1000 PHY ports

→ No actual physical device

ASUS RT-AC56R
ASUS RT-AC56S
ASUS RT-AC56U
ASUS RT-AC66U B1 rev A1
ASUS RT-AC67U
ASUS RT-AC68U V3
ASUS RT-AC68U rev A1
ASUS RT-N18U
ASUS RT-N66U C1
ASUS TM-AC1900
AirTies Air 4920
AirTies Air 4921
AirTies Air 4930
Arcadyan WA9117A-AC23
Belkin F9K1113 v2
Belkin F9K1118 v2
Buffalo WZR-1166DHP
Buffalo WZR-1750DHP
Buffalo WZR-600DHP2
Buffalo WZR-900DHP
D-Link DIR-860L rev A1
D-Link DIR-868L rev A1
D-Link DIR-880L rev A1
Huawei E5186s-22a
Huawei WS880
Kasda KA1200
Kasda KA1750
Linksys EA6200
Linksys EA6300 v0.1
Linksys EA6300 v1
Linksys EA6350
Linksys EA6350 v2
Linksys EA6400
Linksys EA6500 v2
Linksys EA6700
Linksys EA6900 v1.0
Linksys EA6900 v1.1
Motorola MR1900
Netgear AC1450
Netgear EX6200
Netgear EX7000
Netgear Nighthawk RS400
Netgear R6200v2
Netgear R6250
Netgear R6300 v2
Netgear R6400 v1
Netgear R6400 v2 U12H332T20
Netgear R6400 v2 U12H332T30
Netgear R6700 v3
Netgear R6900P
Netgear R7000P
Netgear XR300
SmartRG SR400ac
TP-LINK Archer C5 v2.x
TP-LINK Archer C8 v1.x
TP-LINK Archer C8 v2.x
TP-LINK Archer C8 v4.x
TRENDnet TEW-811DRU
TRENDnet TEW-812DRU v2
TRENDnet TEW-818DRU V1.0R
Tenda AC15
Tenda AC18
Tenda F452
Xunlei rs1309

CROWDSTRIKE

RT-AC87U Firmware

→ 3.0.0.4.382.52545

→ GPL tarball available

→ TRX format

→ Unpack with unblob

→ part0: LZMA-compressed kernel

→ part1: SquashFS filesystem

```
$ file RT-AC87U_3.0.0.4_382_52545-ga0245cc.trx_extract/part0
RT-AC87U_3.0.0.4_382_52545-ga0245cc.trx_extract/part0: LZMA compressed
data, non-streamed, size 4034880


$ file RT-AC87U_3.0.0.4_382_52545-ga0245cc.trx_extract/part1
RT-AC87U_3.0.0.4_382_52545-ga0245cc.trx_extract/part1: Squashfs
filesystem, little endian, version 4.0, xz compressed, 39299338 bytes,
2624 inodes, blocksize: 131072 bytes, created: Fri Apr 30 20:18:52 2021
```

® CrowdStrike, Inc. All Rights Reserved

Interrogate the Firmware

→ `2.6.36.4brcmarm`.

→ Linux 2.6.36.4

→ `brcm` = Broadcom?

→ gcc 4.5.3 (Buildroot 2012.02)

```
$ strings part0_extract/0-1685649.lzma_extract/lzma.uncompressed | grep Linux
Linux version 2.6.36.4brcmarm (root@asus) (gcc version 4.5.3 (Buildroot 2012.02) ) #1 SMP PREEMPT Sat May 1 03:57:40 CST 2021
...


$ strings part0_extract/0-1685649.lzma_extract/lzma.uncompressed | grep root=
root=/dev/mtdblock2 console=ttyS0,115200 init=/sbin/preinit earlyprintk debug
...
```

® CrowdStrike, Inc. All Rights Reserved

CROWDSTRIKE

Interrogate the Firmware

Understand the boot process...

→ `root=/dev/mtdblock2`.

→ MTD = flash.

→ Init process is `/sbin/preinit`.

→ `/sbin/preinit` is a link to `/sbin/rc`.

→ `/sbin/rc` = big binary

```
$ strings part0_extract/0-1685649.lzma_extract/lzma.uncompressed | grep root=
root=/dev/mtdblock2 console=ttyS0,115200 init=/sbin/preinit earlyprintk debug
...


$ ls -al part1_extract/0-39301120.squashfs_v4_le_extract/sbin/preinit
lrwxrwxrwx 1 tests tests 2 Jul  5 20:39 part1_extract/0-39301120.squashfs_v4_le_extract/sbin/preinit -> rc


$ file part1_extract/0-39301120.squashfs_v4_le_extract/sbin/rc
part1_extract/0-39301120.squashfs_v4_le_extract/sbin/rc: ELF 32-bit LSB executable, ARM, EABI5 version 1 (SYSV), dynamically linked, interpreter /lib/ld-uClibc.so.0, stripped
```

® CrowdStrike, Inc. All Rights Reserved

Layout

→ Important system directories missing

→ `/tmp/` must be populated at some point during the boot?

```
$ ls -al part1_extract/0-39301120.squashfs_v4_le_extract/
total 80
drwxrwxr-x 18 tests tests  4096 Jul  5 20:39 .
drwxr-xr-x  3 tests tests  4096 Jul  5 20:39 ..
drwxrwxr-x  2 tests tests  4096 Apr 30  2021 asus_jffs
drwxrwxr-x  2 tests tests  4096 Jul  5 20:39 bin
drwxrwxr-x  2 tests tests  4096 Apr 30  2021 cifs1
drwxrwxr-x  2 tests tests  4096 Apr 30  2021 cifs2
drwxrwxr-x  2 tests tests  4096 Apr 30  2021 dev
lrwxrwxrwx  1 tests tests     7 Jul  5 20:39 etc -> tmp/etc
lrwxrwxrwx  1 tests tests     8 Jul  5 20:39 home -> tmp/home
drwxrwxr-x  2 tests tests  4096 Apr 30  2021 jffs
drwxrwxr-x 11 tests tests  4096 Jul  5 20:39 lib
lrwxrwxrwx  1 tests tests     9 Jul  5 20:39 media -> tmp/media
drwxrwxr-x  2 tests tests  4096 Apr 30  2021 mmc
lrwxrwxrwx  1 tests tests     7 Jul  5 20:39 mnt -> tmp/mnt
lrwxrwxrwx  1 tests tests     7 Jul  5 20:39 opt -> tmp/opt
drwxrwxr-x  2 tests tests  4096 Apr 30  2021 proc
drwxrwxr-x 23 tests tests  4096 Jul  5 20:39 rom
lrwxrwxrwx  1 tests tests    13 Jul  5 20:39 root -> tmp/home/root
drwxrwxr-x  2 tests tests  4096 Jul  5 20:39 sbin
drwxrwxr-x  2 tests tests  4096 Apr 30  2021 sys
drwxrwxr-x  2 tests tests  4096 Apr 30  2021 sysroot
drwxrwxr-x  2 tests tests  4096 Apr 30  2021 tmp
drwxrwxr-x  9 tests tests  4096 Jul  5 20:39 usr
lrwxrwxrwx  1 tests tests     7 Jul  5 20:39 var -> tmp/var
drwxrwxr-x 14 tests tests 12288 Jul  5 20:39 www
```

® CrowdStrike, Inc. All Rights Reserved

Naive Planning

The first cycle.

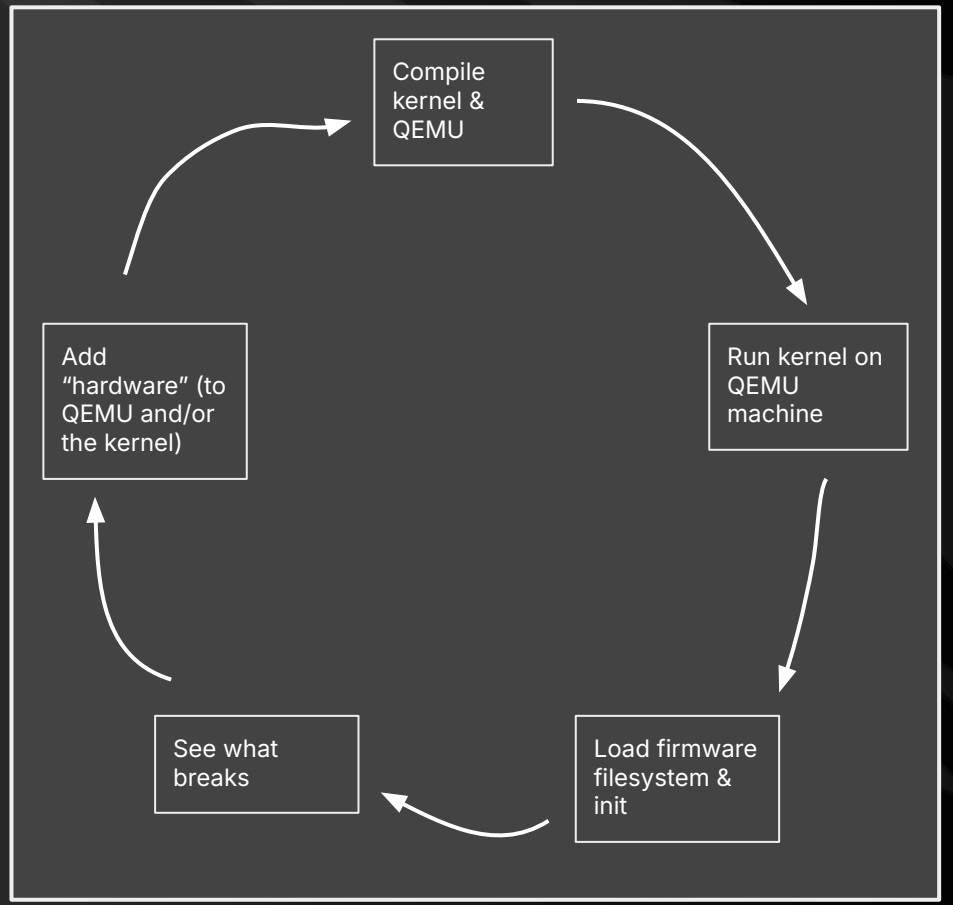
→ Find/build Cortex A9 QEMU machine with flash memory.

→ Build a 2.6.36.4 kernel using gcc 4.5.3

→ Use a defconfig written for the QEMU machine

→ Pass raw SquashFS filesystem binary (`part1`) as flash

→ ???



® CrowdStrike, Inc. All Rights Reserved

Initial Setup

→ `vexpress-a9` board

→ Cortex A9 core

→ Flash memory (`pflash`).

→ `vexpress_defconfig` for the 2.6.36.4 Linux kernel

→ Build GCC 4.5.3 from Buildroot 2012.2 first...

```
static void vexpress_a9_class_init(ObjectClass *oc, void *data)
{
    MachineClass *mc = MACHINE_CLASS(oc);
    VexpressMachineClass *vmc = VEXPRESS_MACHINE_CLASS(oc);

    mc->desc = "ARM Versatile Express for Cortex-A9";
    mc->default_cpu_type = ARM_CPU_TYPE_NAME("cortex-a9");

    vmc->daughterboard = &a9_daughterboard;
}
```

```
sysbus_create_simple("pl111", map[VE_CLCD], pic[14]);

dinfo = drive_get(IF_PFLASH, 0, 0);
pflash0 = ve_pflash_cfi01_register(map[VE_NORFLASH0], "vexpress.flash0",
                                   dinfo);

if (map[VE_NORFLASHALIAS] != -1) {
    /* Map flash 0 as an alias into low memory */
    MemoryRegion *flash0mem;
    flash0mem = sysbus_mmio_get_region(SYS_BUS_DEVICE(pflash0), 0);
    memory_region_init_alias(&vms->flashalias, NULL, "vexpress.flashalias",
                             flash0mem, 0, VEXPRESS_FLASH_SIZE);
    memory_region_add_subregion(sysmem, map[VE_NORFLASHALIAS], &vms->flashalias);
}

dinfo = drive_get(IF_PFLASH, 0, 1);
ve_pflash_cfi01_register(map[VE_NORFLASH1], "vexpress.flash1", dinfo);

sram_size = 0x2000000;
memory_region_init_ram(&vms->sram, NULL, "vexpress.sram", sram_size,
                       &error_fatal);
```

® CrowdStrike, Inc. All Rights Reserved

CROWDSTRIKE

First Test

→ Get the kernel running

→ `CONFIG_*` audit/checks/changes

→ How to pass a “flash” chip?



→ Lay out file as “flash”

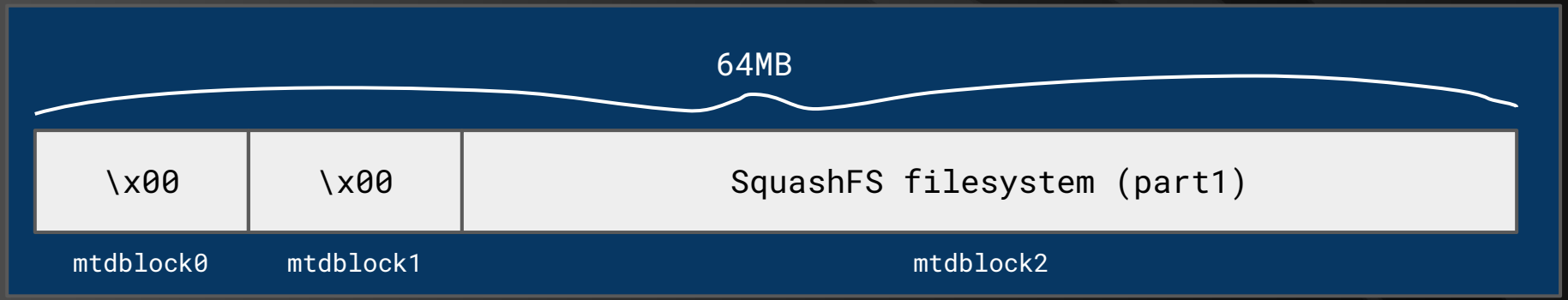
→ `mtdparts` boot arg tells the kernel the layout

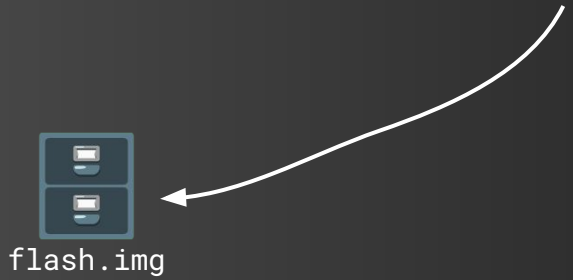
→ Pass flash as `pflash` to QEMU

® CrowdStrike, Inc. All Rights Reserved

CROWDSTRIKE

Flash Layout





QEMU

```
-drive if=pflash,format=raw,file=flash0.img
```

Kernel boot args

```
mtdparts=armflash:512k(something1),512k(something2),63m(rootfs)
```

® CrowdStrike, Inc. All Rights Reserved

CROWDSTRIKE

Expecting Failure

But what kind?

→ Filesystem doesn't unpack!

→ **Why?**

```
qemu-system-arm -M vexpress-a9 -m 1024 \
    -kernel linux-2.6.36.4/arch/arm/boot/zImage \
    -drive if=pflash,format=raw,file=flash0.img \
    -append "root=/dev/mtdblock2 console=ttyS0
mtdparts=armflash:512k(smth1),512k(smth1),63m(rootfs)
mtd_probe=cmdline init=/sbin/preinit"
```

```
...
input: AT Raw Set 2 keyboard as /devices/mb:kmi0/serio0/input/input0
input: ImExPS/2 Generic Explorer Mouse as /devices/mb:kmi1/serio1/input/input1
SQUASHFS error: Filesystem uses "unknown" compression. This is not supported
List of all partitions:
1f00             512 mtdblock0 (driver?)
1f01             512 mtdblock1 (driver?)
1f02           63488 mtdblock2 (driver?)
No filesystem could mount root, tried:  ext3 ext2 cramfs squashfs vfat
Kernel panic - not syncing: VFS: Unable to mount root fs on unknown-block(31,2)
[<8003c594>] (unwind_backtrace+0x0/0x17c) from [<80402e78>] (panic+0x64/0x190)
[<80402e78>] (panic+0x64/0x190) from [<80009194>] (mount_block_root+0x1d0/0x210)
[<80009194>] (mount_block_root+0x1d0/0x210) from [<800093f4>]
(prepare_namespace+0x12c/0x190)
[<800093f4>] (prepare_namespace+0x12c/0x190) from [<80008dbc>]
(kernel_init+0x1f4/0x248)
[<80008dbc>] (kernel_init+0x1f4/0x248) from [<8003724c>] (kernel_thread_exit+0x0/0x8)
```

® CrowdStrike, Inc. All Rights Reserved

uashfs-LZMA.org

quashfs filesystems smaller
ompression algorithm

News

SquashFS version 4 is included in Linux Kernel since 2.6.29 and it supports XZ (LZMA2) compression since 2.6.38. What does it mean for this project (squashfs-lzma.org)? There is no longer any need to use external patches, so this project is discontinued. You can happily mount your newly created squashfs images directly with recent kernel! :) This website will remain with slight updates for reference.

CROWDSTRIKE

Broadcom Kernel Changes

> asus-ac5300 > gpl > asuswrt > release > src-rt-7.14.114.x > src > linux > linux-2.6 > fs > squashfs > C xz_wrapper.c

```
/* Modified by Broadcom Corp. Portions Copyright (c) Broadcom Corp, 2012. */
/*
 * Squashfs - a compressed read only filesystem for Linux
 *
 * Copyright (c) 2002, 2003, 2004, 2005, 2006, 2007, 2008, 2009, 2010
 * Phillip Lougher <phillip@lougher.demon.co.uk>
 *
```

® CrowdStrike, Inc. All Rights Reserved

CROWDSTRIKE

Firmware Unpacks!

```
...
IPv6 over IPv4 tunneling driver
sit0: Disabled Privacy Extensions
ip6tnl0: Disabled Privacy Extensions
NET: Registered protocol family 17
Bridge firewalling registered
Ebtables v2.0 registered
L2TP core driver, V2.0
802.1Q VLAN Support v1.8 Ben Greear <greearb@candelatech.com>
All bugs added by David S. Miller <davem@redhat.com>
lib80211: common routines for IEEE802.11 drivers
VFP support v0.3: implementor 41 architecture 3 part 30 variant 9 rev 0
rtc-pl031 mb:rtc: setting system clock to 2025-07-10 10:18:07 UTC (1752142687)
input: AT Raw Set 2 keyboard as /devices/mb:kmi0/serio0/input/input0
input: ImExPS/2 Generic Explorer Mouse as /devices/mb:kmi1/serio1/input/input1
VFS: Mounted root (squashfs filesystem) readonly on device 31:2.
devtmpfs: mounted
Freeing init memory: 180K
...
```

® CrowdStrike, Inc. All Rights Reserved

CROWDSTRIKE

init=/bin/sh

```
...
VFP support v0.3: implementor 41 architecture 3 part 30 variant 9 rev 0
rtc-pl031 mb:rtc: setting system clock to 2025-07-10 14:27:55 UTC (1752157675)
input: AT Raw Set 2 keyboard as /devices/mb:kmi0/serio0/input/input0
input: ImExPS/2 Generic Explorer Mouse as /devices/mb:kmi1/serio1/input/input1
VFS: Mounted root (squashfs filesystem) readonly on device 31:2.
devtmpfs: mounted
Freeing init memory: 180K
/ # ls -al
drwxr-xr-x   18 0        0             325 Apr 30  2021 .
drwxr-xr-x   18 0        0             325 Apr 30  2021 ..
drwxr-xr-x    2 0        0               3 Apr 30  2021 asus_jffs
drwxr-xr-x    2 0        0             667 Apr 30  2021 bin
drwxr-xr-x    2 0        0               3 Apr 30  2021 cifs1
drwxr-xr-x    2 0        0               3 Apr 30  2021 cifs2
drwxr-xr-x    4 0        0            2760 Jan  1  1970 dev
lrwxrwxrwx    1 0        0               7 Apr 30  2021 etc -> tmp/etc
lrwxrwxrwx    1 0        0               8 Apr 30  2021 home -> tmp/home
drwxr-xr-x    2 0        0               3 Apr 30  2021 jffs
drwxr-xr-x    3 0        0             260 Apr 30  2021 lib
lrwxrwxrwx    1 0        0               9 Apr 30  2021 media -> tmp/media
drwxr-xr-x    2 0        0               3 Apr 30  2021 mmc
lrwxrwxrwx    1 0        0               7 Apr 30  2021 mnt -> tmp/mnt
...
```

© CrowdStrike, Inc. All rights reserved

CROWDSTRIKE

/sbin/preinit

→ /sbin/rc in preinit mode

→ Mounts sys, dev, tmp, proc, etc.

→ Copies /etc/ from /rom/etc/*

→ Reads NVRAM a lot

→ Brings up network interfaces

→ & more!

```
if ( a1 > 1 && !strcmp(*(const char **)(a2 + 4), "reboot") )
{
  if ( mount("proc", "/proc", "proc", 0, 0) )
    perror("## mount proc");
  if ( mount("tmpfs", "/tmp", "tmpfs", 0, 0) )
    perror("## mount tmpfs");
  if ( mount("sysfs", "/sys", "sysfs", 0xC0ED0000, 0) )
    perror("## mount sysfs");
  if ( mount("devpts", "/dev/pts", "devpts", 0xC0ED0000, 0) )
    perror("## mount devpts");
  if ( sub_12B30() )
    dword_EE230 = 1;
  stime(&when);
  f_write("/tmp/settings", 0, 0, 0, 420);
  symlink("/proc/mounts", "/etc/mtab");
  set_action(0);
  if ( !dword_EE230 )
  {
    puts("\n\nHit ENTER for new console...\n");
    waitfor_run_bin_login(1);
  }
  for ( i = &unk_B70DC; i != (_DWORD *)&unk_B70FC; ++i )
  {
    v3 = i[1];
    signal(v3, handler);
  }
```

```
mkdir("/var/state", 0x1FFu);
mkdir("/var/state/parport", 0x1FFu);
mkdir("/var/state/parport/svr_statue", 0x1FFu);
mkdir("/var/lock", 0x1FFu);
mkdir("/var/tmp/dhcp", 0x1FFu);
mkdir("/home/root", 0x1C0u);
chmod("/tmp", 0x1FFu);
chmod("/tmp/mnt", 0x1FFu);
f_write("/etc/hosts", 0, 0, 0, 420);
f_write("/etc/fstab", 0, 0, 0, 420);
f_write("/tmp/settings", 0, 0, 0, 420);
v11 = opendir("/rom/etc");
if ( v11 )
{
  while ( 1 )
  {
    v13 = readdir(v11);
    if ( !v13 )
      break;
    if ( v13->d_name[0] != 46 )
    {
      d_name = v13->d_name;
      snprintf(s, 0x100u, "%s/%s", "/rom/etc", v13->d_name);
      snprintf(v233, 0x100u, "%s/%s", "/etc", d_name);
      symlink(s, v233);
    }
  }
```

® CrowdStrike, Inc. All Rights Reserved

CROWDSTRIKE

init=/sbin/preinit

```
...
input: ImExPS/2 Generic Explorer Mouse as /devices/mb:kmi1/serio1/input/input0
input: AT Raw Set 2 keyboard as /devices/mb:kmi0/serio0/input/input1
VFS: Mounted root (squashfs filesystem) readonly on device 31:2.
devtmpfs: mounted
Freeing init memory: 180K
/dev/nvram: No such file or directory
/dev/nvram: No such file or directory
## mknod /dev/null: File exists
## mknod /dev/console: File exists
/dev/nvram: No such file or directory
/dev/nvram: No such file or directory
/dev/nvram: No such file or directory
1: set_action 0
/dev/nvram: No such file or directory


Hit ENTER for console...

firmware version: 3.0.0.4.382_52545-ga0245cc
/dev/nvram: No such file or directory
/dev/nvram: No such file or directory
/dev/nvram: No such file or directory
...
```

® CrowdStrike, Inc. All Rights Reserved

CROWDSTRIKE

NVRAM 101

→ Filesystem is r/o, NVRAM is r/w

→ key=value store

→ `nvram_get()`, `nvram_set()`, `nvram_commit()`, etc in `libnvram.so`

```
Ate_temp_5G_max=0
Ate_temp_5G_over_sec=
Ate_temp_cpu_limit=115
Ate_temp_cpu_max=0
Ate_temp_cpu_over_sec=
Ate_temp_phy_limit=115
Ate_temp_phy_max=0
Ate_temp_phy_over_sec=
CoBrand=4
DCode=20230328
HwBom=60IG0820-MB0B01
HwId=A
HwVer=1.0
ICFILTER_MAC=
ICFILTER_MACFILTER_DAYTIME=
MULTIFILTER_BLOCK_ALL=0
MULTIFILTER_MACFILTER_DAYTIME_V2_CONVERTED=1
NVRAMRev=$Rev: 791930 $
OPTUS_MULTIFILTER_ALL=0
OPTUS_MULTIFILTER_ENABLE=
TM_EULA_time=Sat, 05 May 2018 08:33:45 +0100
aae_area=
aae_awsiot_desc_updated=0
aae_portal=
abl_eid=0
acs_unii4=0
acs_version=2
acscli2_acs_restart=1
ahs_bhc_log=/jffs/bhc_chg.log
ahs_bhc_log_1=/jffs/bhc_chg.log.1
```

libnvram.so

→ Contains `nvram_init()`, `nvram_get()`, `nvram_set()`, etc.

`nvram_init()`
Opens `/dev/nvram` and stores the `fd` at a global memory location.

`nvram_get()`
`read(fd, key, outbuf).`

`nvram_set()`
`write(fd, keyvalpair, ret).`

`nvram_commit()`
`ioctl(fd, 0x48534C46u, 0).`

```
int nvram_init()
{
 int fd; // r0

 if ( nvramfd >= 0 )
   return 0;
 fd = open("/dev/nvram", 2);
 nvramfd = fd;
 if ( fd >= 0 )
 {
   dword_8C90 = (int)mmap(0, 0x10000u, 1, 1, fd, 0);
   if ( dword_8C90 != -1 )
   {
     fcntl(nvramfd, 2, 1);
     return 0;
   }
   close(nvramfd);
   nvramfd = -1;
 }
 perror("/dev/nvram");
 return *_errno_location();
}
```

® CrowdStrike, Inc. All Rights Reserved

Fake NVRAM in the Kernel

```
...
static struct miscdevice nvram_miscdev = {
   .minor = MISC_DYNAMIC_MINOR,
   .name = "nvram",
   .fops = &nvram_fops,
};
...
```

Device at `/dev/nvram`

```
...
#define MAX_MSG_LEN 512
#define MAX_FILTER_LIST_ENTRIES 16
#define MAX_FILTER_ENTRY_LEN    32

#define NVRAM_MAX_DEVICE_SIZE (512 * 1024)
#define NVRAM_BUFSIZE 0x10000

#define NVRAM_DEVICE "/dev/mtdblock1"
...
```

Backed by a part of the “flash”

```
...
static struct file_operations nvram_fops = {
   .owner = THIS_MODULE,
   .read = nvram_read,
   .write = nvram_write,
   .mmap = nvram_mmap,
   .unlocked_ioctl = nvram_ioctl,
};
...
```

Handles all `read()`, `write()`, `mmap()`, `ioctl()`.

® CrowdStrike, Inc. All Rights Reserved

CROWDSTRIKE

Fake NVRAM in the Kernel

```
static ssize_t nvram_write(struct file *file, const char __user
*buf, size_t count)
{
   char *input = NULL;
   char *key = NULL, *value = NULL;
   char *eq = NULL;
   pid_t pid = current->pid;
   const char *proc_name = current->comm;

   if (count == 0 || count >= PAGE_SIZE)
       return -EINVAL;

   input = kmalloc(count + 1, GFP_KERNEL);
   if (!input)
       return -ENOMEM;

   if (copy_from_user(input, buf, count)) {
       kfree(input);
       return -EFAULT;
   }

   input[count] = '\0';
   pr_warn(proc_name, "[nvram] %s (%d) trying to write key
%s\n", proc_name, pid, input);

   nvram_load_from_device();

   eq = strchr(input, '=');
   if (eq) {
       *eq = '\0';
       key = input;
       value = eq + 1;
   } else {
```

```
static long nvram_ioctl(struct file *file, unsigned int cmd,
unsigned long arg)
{
   if (cmd == 0x48534C46)
       pr_warn("NVRAM commit called with 0x%x\n", cmd);
       return nvram_commit_to_device();
   pr_err("NVRAM ioctl called with unknown argument 0x%x\n",
cmd);
   return -EINVAL;
}
```

```
static void nvram_load_from_device(void)
{
   struct file *filp;
   mm_segment_t oldfs;
   loff_t pos;
   ssize_t bytes;
   char *buf;

   if (g_nvram_loaded)
       return;

   buf = kmalloc(NVRAM_BUFSIZE, GFP_KERNEL);
   if (!buf)
       return;
```

```
static ssize_t nvram_read(struct file *file, char __user *buf,
size_t count)
{
   char key[128];
   char *ptr, *eq;

   nvram_load_from_device();

   pid_t pid = current->pid;
   const char *proc_name = current->comm;

   key[count] = '\0';
   ptr = g_nvram_buf;
   while (*ptr) {
       eq = strchr(ptr, '=');
       if (!eq)
           break;
       if (strncmp(ptr, key, eq - ptr) == 0 && strlen(key) ==
(size_t)(eq - ptr)) {
           char *value = eq + 1;
           unsigned long offset = value - g_nvram_buf;
           pr_warn(proc_name, "[nvram] %s (%d) read key %s,
value: %s\n", proc_name, pid, buf, value);
           return copy_to_user(buf, &offset, sizeof(offset)) ?
-EFAULT : sizeof(offset);
       }
       ptr += strlen(ptr) + 1;
   }

   pr_warn(proc_name, "[nvram] %s (%d) tried to read key %s,
but couldn't find it\n", proc_name, pid, buf);
   return 0;
}
```

® CrowdStrike, Inc. All Rights Reserved

CROWDSTRIKE

Working Fake System-Wide NVRAM 🙂

- ✅ `/dev/nvram` device
- ✅ Persistent
- ✅ Transferrable
- ✅ Hackable

```
admin@RT-AX87U:/tmp/home/root# nvram show
...
68feaaa3=1
6e3f7d37=1
767cd197=Sat, 05 May 2018 10:39:08 +0100
ASUS_EULA_time=Sat, 05 May 2018 08:16:43 +0100
AllLED_brightness=255
Ate_runin_skip_stress=0
Ate_temp_2G_limit=115
Ate_temp_2G_max=0
Ate_temp_2G_over_sec=
Ate_temp_5G_limit=115
Ate_temp_5G_max=0
Ate_temp_5G_over_sec=
Ate_temp_cpu_limit=115
Ate_temp_cpu_max=0
Ate_temp_cpu_over_sec=
Ate_temp_phy_limit=115
Ate_temp_phy_max=0
Ate_temp_phy_over_sec=
CoBrand=4
...
```

® CrowdStrike, Inc. All Rights Reserved

CROWDSTRIKE

What Goes in NVRAM?

```
{
  restore_defaults = 1;
  v50 = fputs("\n## Restoring defaults... ##\n", (FILE *)stderr);
}
restore_defaults_g = restore_defaults;
if ( restore_defaults )
{
  nvram_unset("wps_config_state");
  nvram_unset("wps_proc_status");
  nvram_unset("wps_proc_status_x");
  nvram_unset("wps_config_method");
  nvram_unset("wps_band");
  nvram_unset("wps_restart");
  nvram_unset("wps_proc_mac");
  nvram_unset("wps_sta_devname");
  nvram_unset("wps_sta_mac");
  nvram_unset("wps_pinfail");
  nvram_unset("wps_pinfail_mac");
  nvram_unset("wps_pinfail_name");
  nvram_unset("wps_pinfail_state");
  nvram_unset("wps_enr_ssid");
  nvram_unset("wps_enr_bssid");
  v51 = nvram_unset("wps_enr_wsec");
  productid = get_productid(v51);
  v53 = nvram_set("wps_device_name", productid);
  v54 = get_productid(v53);
  v55 = nvram_set("wps_modelnum", v54);
  lan_hwaddr = (const char *)get_lan_hwaddr(v55);
```

```
nvram_set("ntp_ready", "0");
nvram_set("svc_ready", "0");
nvram_unset((int)"qtn_ready");
if ( a1 )
{
  nvram_set("jffs2_clean_fs", "1");
  nvram_set("qtn_restore_defaults", "1");
}
else
{
  nvram_unset((int)"qtn_restore_defaults");
}
nvram_set("freeze_duck", "0");
nvram_unset((int)"ateCommand_flag");
nvram_unset((int)"ateUpgarde_flag");
nvram_set_int("vpnc_state_t", 0);
nvram_set_int("vpnc_sbstate_t", 0);
nvram_set("apps_ipkg_old", "0");
nvram_set("apps_new_arm", "0");
nvram_set("apps_install_folder", "asusware.arm");
nvram_set("apps_ipkg_server", "http://nw-dlcdnet.asus.com/asusware/arm/stable");
nvram_set("aae_support", "1");
nvram_set("aae_enable", "0");
v27 = nvram_get_int("aae_enable");
nvram_set_int("aae_enable", v27 | 2);
v28 = nvram_get_int("aae_enable");
nvram_set_int("aae_enable", v28 | 4);
nvram_unset((int)"wps_reset");
nvram_set_int("AllLED", 1);
return nvram_unset((int)"reload_svc_radio");
```

→ No device = no NVRAM dump

→ `/sbin/preinit` populates “defaults”

® CrowdStrike, Inc. All Rights Reserved

Uncharted Waters

→ `/sbin/preinit` get further

→ Needs more network interfaces!

```
...
ifconfig: SIOCSIFADDR: No such device
generate_wl_para(0x0b6c): unit 0 subunit -1
eth1: WLC_GET_VAR(cap): No such device
eth1: cmd=39 (errno 19): No such device
generate_wl_para(0x0f0a): bw: 0
...
```

® CrowdStrike, Inc. All Rights Reserved

Patching QEMU

Modify vexpress-a9 to add another interface

Modify kernel machine definition to support it

```
static void vexpress_common_init(MachineState *machine)
    if (nd_table[0].used) {
    /* 0x4e100000 Second LAN9118 Ethernet (custom addition) */
    // pic[32] corresponds to the offset value of
    // IRQ_V2M_LAN9118_2 in motherboard.h in the mach-vexpress board.
    if (nd_table[1].used) {
        lan9118_init(&nd_table[1], map[VE_ETHERNET2], pic[32]);
    }

    /* VE_USB: not modelled */
```

```
static hwaddr motherboard_legacy_map[] = {
    /* CS1: 0x44000000 .. 0x48000000 */
    [VE_NORFLASH1] = 0x44000000,
    /* CS2: 0x48000000 .. 0x4a000000 */
    [VE_SRAM] = 0x48000000,
    /* CS3: 0x4c000000 .. 0x50000000 */
    [VE_VIDEORAM] = 0x4c000000,
    [VE_ETHERNET1] = 0x4e000000,
    [VE_ETHERNET2] = 0x4e800000,
    [VE_USB] = 0x4f000000,
};
```

```
#ifndef __MACH_MOTHERBOARD_H
#define IRQ_V2M_KMI0            { (32 + 12) }
#define IRQ_V2M_KMI1            { (32 + 13) }
#define IRQ_V2M_CLCD            { (32 + 14) }
#define IRQ_V2M_LAN9118         (32 + 15)
#define IRQ_V2M_LAN9118_2           (32 + 32)   // ADDED
#define IRQ_V2M_ISP1761         (32 + 16)
#define IRQ_V2M_PCIE            (32 + 17)
```

® CrowdStrike, Inc. All Rights Reserved

Patching QEMU

→ Compile QEMU & kernel

→ Run with new machine

→ Run with more NICs

```
$ qemu-system-arm -M bcm4709 -m 1024 \
    -kernel zImage \
    -drive if=pflash,format=raw,file=flash0.img \
    -append "root=/dev/mtdblock2 console=ttyAMA0 mtdparts=armflash:512k(bootloader),512k(nvram),62m(rootfs)
mtd_probe=cmdline init=/sbin/preinit" \
    -nographic \
    -netdev user,id=net0 -net nic,netdev=net0 \
    -netdev user,id=net1 -net nic,netdev=net1

...
```

® CrowdStrike, Inc. All Rights Reserved

CROWDSTRIKE

New Errors!

```
...
cp: can't stat '/jffs/syslog.log': No such file or directory
cp: can't stat '/jffs/syslog.log-1': No such file or directory
...
wanduck(0)(fo change): state 0, state_old 0, changed 0, wan_state 0.
wanduck(0)(all   end): state 0, state_old 0, changed 0, wan_state 0.
[register_feature] blockfile registered
mv: can't rename '/jffs/asd.log': No such file or directory
[register_feature] chknvram registered
mv: can't rename '/jffs/asd.log': No such file or directory
[register_feature] misc registered
...
Registering HINFO record with values 'ARMV7L'/'LINUX'.
json_object_from_file: error opening file /jffs/nmp_cl_json.js: No such file or directory
/jffs/cert.tgz -C / etc/cert.pem etc/key.pem
decomp: gzip -dc /var/lib/misc/rstats-history.gz > /var/tmp/rstats-uncomp != 0
load_history: load failed
ls: /jffs/usericon/*.log: No such file or directory
[rc 710] hour_monitor
tar: can't open '/jffs/cert.tgz': No such file or directory
...
```

® CrowdStrike, Inc. All Rights Reserved

CROWDSTRIKE

`preinit` Again

→ `preinit` creates JFFS2 partitions

→ Based on `/proc/mtd` map file

```
if ( !nvram_get_compare_1("jffs2_on") )
  return notice_set();
result = wait_action_idle(10);
if ( result )
{
  result = mtd_getinfo("brcmnand", &blockNum, &partSize);
  if ( result )
  {
    model = get_model(result);
    dword_FF5E8 = 0;
    cprintf("start jffs2: %d, %d\n", blockNum, partSize);
    compare_1 = nvram_get_compare_1("jffs2_format");
    if ( compare_1 )
    {
      nvram_set("jffs2_format", "0");
      v3 = model > 0x38;
      if ( model != 56 )
        v3 = model - 53 > 1;
```

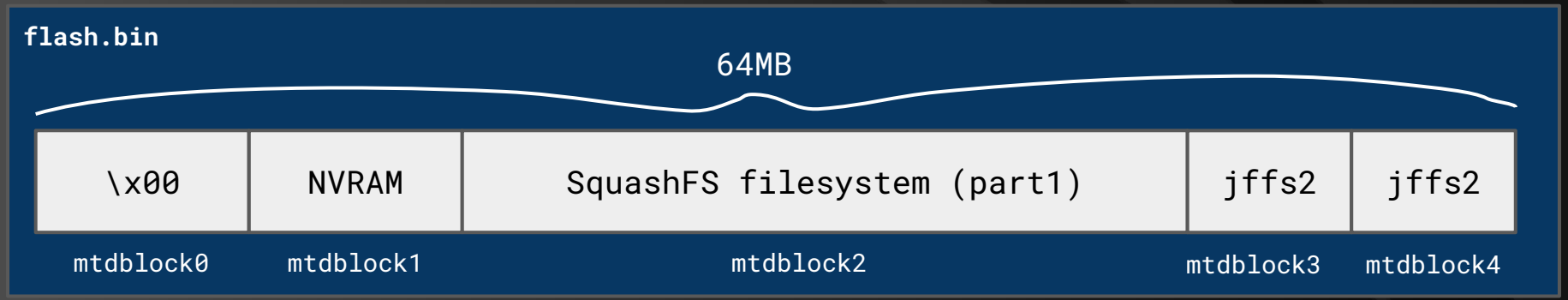
```
int __fastcall mtd_getinfo(const char *a1, _DWORD *a2, _DWORD *a3)
{
  FILE *v6; // r7
  int v7; // r4
  int result; // r0
  char v9[256]; // [sp+0h] [bp-220h] BYREF
  char v10[288]; // [sp+100h] [bp-120h] BYREF

  if ( strlen(a1) <= 0x7F )
  {
    sprintf(v9, "\"%s\"", a1);
    v6 = fopen("/proc/mtd", "r");
    if ( v6 )
    {
      while ( fgets(v10, 256, v6) )
      {
        if ( sscanf(v10, "mtd%d: %x", a2, a3) == 2 && strstr(v10, v9) )
        {
          v7 = 1;
          goto LABEL_9;
        }
      }
      v7 = 0;
```

® CrowdStrike, Inc. All Rights Reserved

CROWDSTRIKE

What Can We Do?




```
mtdparts=armflash:512k(bootloader),512k(nvram),58m(rootfs),2m(brcmnand),2m(asus)
```

→ CONFIG_JFFS*

→ Add more parts to flash

```
admin@RT-AX87U:/tmp/home/root# cat /proc/mtd
dev:    size   erasesize  name
mtd0: 00080000 00040000 "bootloader"
mtd1: 00080000 00040000 "nvram"
mtd2: 03a00000 00040000 "rootfs"
mtd3: 00200000 00040000 "brcmnand"
mtd4: 00200000 00040000 "asus"
```

® CrowdStrike, Inc. All Rights Reserved

CROWDSTRIKE

JFFS Works!

→ Log files

→ Certificates

```
admin@RT-AX87U:/tmp/home/root# ls -al /jffs/
drwxr-xr-x    4 admin    root             0 May  5 06:05 .
drwxr-xr-x   18 admin    root           325 Apr 30  2021 ..
-rw-rw-rw-    1 admin    root         48554 May  5 06:07 HTTPD_DEBUG.log
drwxr--r--    2 admin    root             0 May  5 06:05 asd
-rw-rw-rw-    1 admin    root          1821 May  5 06:05 asd.log
-rw-rw-rw-    1 admin    root          2078 May  5 06:05 cert.tgz
-rw-r--r--    1 admin    root            80 May  5 06:06 nmp_cl_json.js
-rw-rw-rw-    1 admin    root         19296 May  5 06:04 syslog.log
```

® CrowdStrike, Inc. All Rights Reserved

CROWDSTRIKE

More Errors

→ `/dev/gpio` doesn't exist

→ Doesn't completely block, but nice-to-have

→ Create quick dummy GPIO

```
...
Failed to open /dev/gpio
Failed to open /dev/gpio
wanduck: delay 3 seconds before the first detect...
wanduck: delay 4 seconds before the first detect...
wanduck: delay 5 seconds before the first detect...
wanduck(0)(first detect start): state 1, state_old 0, changed 0,
wan_state 0.
...
```

® CrowdStrike, Inc. All Rights Reserved

CROWDSTRIKE

GPIO Hell

→ Button state is polled

→ Read/write state in [libshared.so]

→ `0xC0084701` - `0xC0084705` ops

```
int __fastcall gpio_write(int pin, int a2)
{
  int gpiofd; // r0
  int gpiofd_1; // r5
  int pinbitmap; // r4
  int value; // r3

  gpiofd = j_gpio_open(0);
  gpiofd_1 = gpiofd;
  if ( gpiofd >= 0 )
  {
    pinbitmap = 1 << pin;
    gpio_ioctl(gpiofd, 0xC0084701, pinbitmap, pinbitmap);
    gpio_ioctl(gpiofd_1, 0xC0084705, pinbitmap, pinbitmap);
    if ( a2 )
      value = pinbitmap;
    else
      value = 0;
    gpio_ioctl(gpiofd_1, 0xC0084703, pinbitmap, value);
    return close(gpiofd_1);
  }
  return gpiofd;
}
```

```
gpiofd = open_dev_gpio();
v3 = gpiofd >= 0;
if ( !a1 )
  v3 = 0;
gpiofd_1 = gpiofd;
if ( v3 )
{
  for ( i = 0; i != 16; ++i )
  {
    if ( ((1 << i) & a1) == 1 << i )
    {
      gpio_ioctl(gpiofd_1, 0xC0084701, (unsigned int *)((1 << i) & a1), (1 << i) & a1);
      gpio_ioctl(gpiofd_1, 0xC0084705, (unsigned int *)((1 << i) & a1), 0);
    }
  }
  close(gpiofd_1);
  return open_dev_gpio();
```

® CrowdStrike, Inc. All Rights Reserved

CROWDSTRIKE

GPIO Hell

→ Figure out “active” state

→ Reference DTS files

→ Return “high” if ACTIVE_LOW on e.g. reset button. Otherwise: boot loop!

```
        2ghz {
                label = "bcm53xx:white:2ghz";
                gpios = <&chipcommon 13 GPIO_ACTIVE_LOW>;
                linux,default-trigger = "default-off";
        };
};

gpio-keys {
        compatible = "gpio-keys";
        #address-cells = <1>;
        #size-cells = <0>;

        rfkill {
                label = "WiFi";
                linux,code = <KEY_RFKILL>;
                gpios = <&chipcommon 4 GPIO_ACTIVE_LOW>;
        };

        wps {
                label = "WPS";
                linux,code = <KEY_WPS_BUTTON>;
                gpios = <&chipcommon 5 GPIO_ACTIVE_LOW>;
        };

        restart {
                label = "Reset";
                linux,code = <KEY_RESTART>;
                gpios = <&chipcommon 6 GPIO_ACTIVE_LOW>;
        };
};
```

® CrowdStrike, Inc. All Rights Reserved

NIC Hell

```
int __fastcall get_phy_status(unsigned __int8 a1)
{
  int v2; // r7
  int fd; // r0
  int v4; // r5
  int v6; // [sp+Ch] [bp-1Ch] BYREF

  v2 = j_get_switch();
  if ( v2 )
  {
    fd = socket(2, 2, 0);
    v4 = fd;
    if ( fd >= 0 )
    {
      if ( v2 == 4 )
      {
        if ( j_robo_ioctl(fd, 0, 1, 0, &v6) < 0 )
          j_cprintf("et ioctl SIOCGETCROBORD failed!\n");
        v2 = v6 & 0x1F & a1;
      }
      else
      {
        v2 = 0;
      }
      close(v4);
```

```
int __fastcall robo_ioctl(int sockfd, int a2, int a3, int a4, int *a5)
{
  int v7; // r6
  int result; // r0
  unsigned int *v9[8]; // [sp+0h] [bp-50h] BYREF
  int v10[12]; // [sp+20h] [bp-30h] BYREF

  v7 = a4 | (a3 << 16);
  memset(v9, 0, sizeof(v9));
  strcpy((char *)v9, "eth0");
  v10[0] = v7;
  v9[4] = (unsigned int *)v10;
  v10[1] = 0;
  v10[2] = *a5;
  result = ioctl(sockfd, dword_3EB58[a2], v9);
  *a5 = v10[2];
  return result;
```

→ QEMU uses `smsc911x`

→ Firmware expects BCM57XXX

→ `get_phy_status()` in `libshared.so`

® CrowdStrike, Inc. All Rights Reserved

CROWDSTRIKE

Commit Fraud

→ Handle these operations in the `smsc911x` driver.

→ Tell userspace that the interface is “up”

```
static int smsc911x_do_ioctl(struct net_device *dev, struct ifreq *ifr, int cmd)
{
...
   switch (cmd) {
...
       case SIOCGETCROBORD:
           if (copy_from_user(&args, ifr->ifr_data, sizeof(args)))
               return -EFAULT;

           // Correct structure for BCM5301X?
           page_reg = args[0];
           page = (args[0] >> 16) & 0xffff;
           reg = args[0] & 0xffff;

           if ((page == 0x1) && (reg == 0x0)) {
               int override = 0xffffffff;
               args[2] = override;

           } else {
               ...
           }

           if (copy_to_user(ifr->ifr_data, &args, sizeof(args)))
               return -EFAULT;

           return ret;

...
       default:
           // This is the original handler
...
}
```

© CrowdStrike, Inc. All Rights Reserved

CROWDSTRIKE

LAN Plan

```
admin@RT-AX87U:/tmp/home/root# brctl show
bridge name     bridge id               STP enabled     interfaces
br0             8000.525400123457       yes             eth1
```

→ The AC87U creates a bridge, adds eth1 to that

→ Add a TAP device on the host

```
$ sudo ip tuntap add dev tap0 mode tap
$ sudo ip addr add 192.168.1.22/24 dev tap0
$ sudo ip link set tap0 up
```

→ Pass a tap network device to QEMU as the eth1 interface.

```
../../qemu/build/qemu-system-arm -M bcm4709 -m 1024 \
   -kernel ../../linux-2.6.36.4/arch/arm/boot/zImage \
   -drive if=pflash,format=raw,file=flash0.img \
   -append "root=/dev/mtdblock2 console=ttyS0
mtdparts=armflash:512k(bootloader),512k(nvram),58m(rootfs),2m(brcmnand),2m(asus) mtd_probe=cmdline
init=/sbin/preinit" \
   -nographic \
   -netdev user,id=net0 -net nic,netdev=net0 \
   -netdev tap,id=lan0,ifname=tap0,script=no,downscript=no -net nic,netdev=lan0
```

® CrowdStrike, Inc. All Rights Reserved

CROWDSTRIKE

It (mostly) works!

DEMO

® CrowdStrike, Inc. All Rights Reserved

CROWDSTRIKE

It (mostly) works!

As much as we need it to, for now.

✅ **httpd**

✅ SSH/telnetd

✅ General configuration

✅ Most other services

❌ WLAN

🤔 Flash-heavy mechanisms

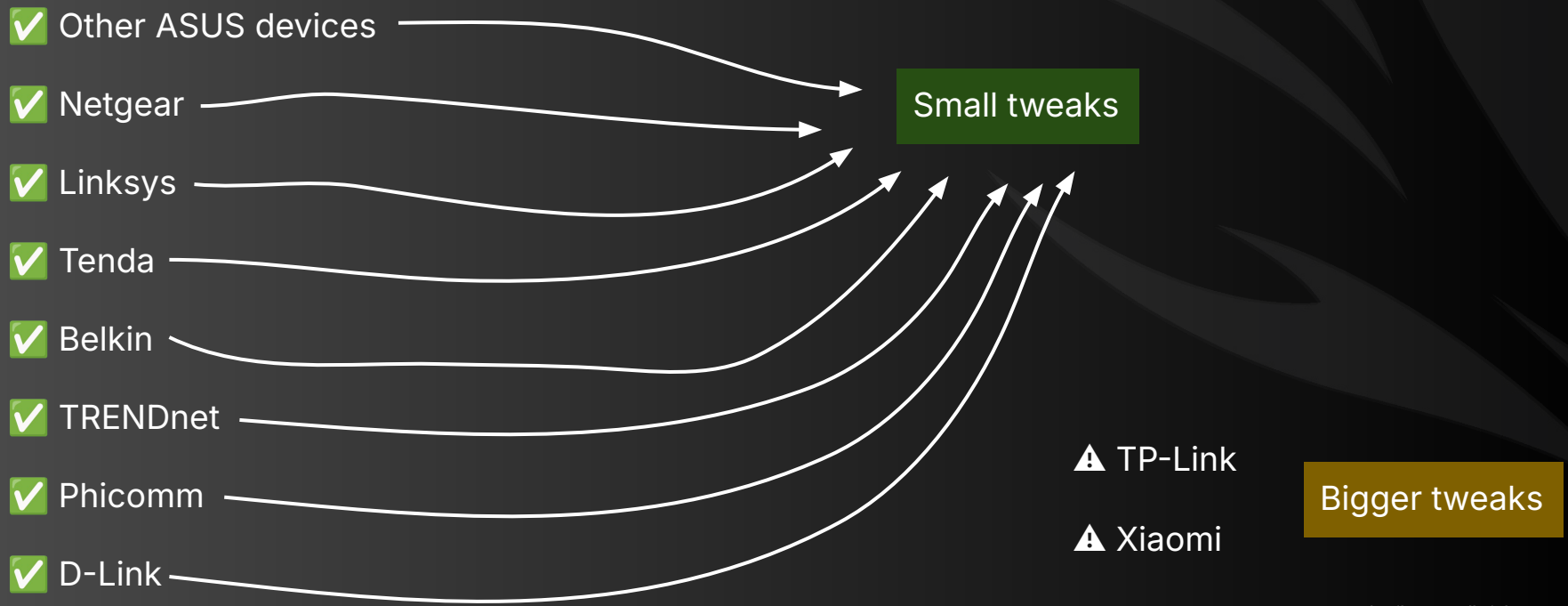
ℹ️ Use kernel-level logging to see which processes are trying to do what.

® CrowdStrike, Inc. All Rights Reserved

CROWDSTRIKE

More Devices!

More devices based on BCM4709 work, some with more tweaks than others...



® CrowdStrike, Inc. All Rights Reserved

CROWDSTRIKE

Conclusions

→ Still early days/PoC, but it works

→ Relatively light-touch

→ Hybrid QEMU/kernel approach works!

→ Concepts are transportable

→ Porting to other SoCs looking promising

Questions? Abuse?
@n-o.bsky.social

® CrowdStrike, Inc. All Rights Reserved

CROWDSTRIKE

Appendix: Other Tweaks

→ Change the `pl011` default name from `ttyAMA*` to `ttyS*`.

→ Flash layouts are more important in some than others

→ Proprietary mechanisms exist (e.g. Tenda CFM, TP-Link bcm_flash_chrdev)

® CrowdStrike, Inc. All Rights Reserved

CROWDSTRIKE

Appendix: Tenda CFM

→ First 8 bytes of the partition are a header

→ First 4 bytes of the header is the CRC32 of the non-header section

→ That's it!

```
...
# Tenda libCfm.so -> get_cfm_new_flag_from_mtd()
crc = zlib.crc32(b"\x00"*(part.size-8))
outfile.write(crc.to_bytes(4, byteorder="little"))
...
```

® CrowdStrike, Inc. All Rights Reserved

CROWDSTRIKE

Appendix: Dynamic RE From Kernel

→ Want to know where ioctls are being issued!

→ In dummy drivers, use `current` to get registers

→ ARM32 link register has return address

→ Combination `mm` & `task_pt_regs()` to get the `libname.so+0x<offset>`

```
...
GetUnitFromIfname(1844) ifname=eth0
wan_up network.c 2919 wan_ifname=eth0 index=0
wan_up 2939 wan_proto
[acos_nat_cli hack] udhcpc (10939, libnat.so+0xc474) called ioctl. cmd: 0x800464b2, arg: 0x2.
[acos_nat_cli hack] udhcpc (10939, libnat.so+0xc474) called unknown ioctl. cmd: 0x800464b2, arg: 0x2.
[Bonjour] Can not open "/proc/printer_status"
[acos_nat_cli hack] httpd (10955, libnat.so+0x9fe4) called ioctl. cmd: 0x8004643d, arg: 0x7e842d14.
[acos_nat_cli hack] httpd (10955, libnat.so+0x9fe4) called ioctl. cmd: 0x8004643d, arg: 0x7e842d14.
[Bonjour] Can not open "/proc/printer_status"
call add_ns wan_ifname=eth0
route: ioctl 0x890c failed: No such process
GetUnitFromIfname(1844) ifname=eth0
...
```

```c
static int get_caller_info(struct caller_info *info)
{
    struct pt_regs *regs;
    struct mm_struct *mm;
    struct vm_area_struct *vma;
    unsigned long caller_addr;
    int ret = -1;

    if (!info || !current || !(mm = current->mm)) {
        return -EINVAL;
    }

    regs = task_pt_regs(current);
    // Use link register (LR) to get the return address
    caller_addr = regs->ARM_lr;

    down_read(&mm->mmap_sem);

    // Find VMA containing the return address
    vma = find_vma(mm, caller_addr);
    if (vma && vma->vm_start <= caller_addr) {
        if (vma->vm_file) {
            char buf[256];
            char *path = d_path(&vma->vm_file->f_path, buf, sizeof(buf));
            if (!IS_ERR(path)) {
                char *filename = strrchr(path, '/');
                if (filename) {
                    filename++; // Skip the '/'
                }else {
                    filename = path;
                }

                strncpy(info->lib_name, filename, sizeof(info->lib_name) - 1);
                info->lib_name[sizeof(info->lib_name) - 1] = '\0';
                info->offset = caller_addr - vma->vm_start;
                ret = 0;
            }
        }
    }

    up_read(&mm->mmap_sem);
    return ret;
}
```

® CrowdStrike, Inc. All Rights Reserved

CROWDSTRIKE

Appendix: Netgear `acos_nat_cli`

→ Char device

→ Lots of `ioctls` to it

→ Dummy driver which catches known `ioctls` and returns 1

→ Can “do something” if needed

```
/bin/mknod -m 755 /dev/acos_nat_cli c 100 0
                                      ^  ^  ^
                                      |  |  |
                                char dev | minor number
                                         |
                                    major number
```

```
static int device_ioctl(struct file *file, unsigned int cmd, unsigned long arg)
{
   int ret;
   const pid_t pid = current->pid;
   const char *proc_name = current->comm;

   struct caller_info info;

   switch (cmd & 0xFFFFFFFF) {
       case 0x400464aa:
           // httpd calls IOCTL_AG_REGION_SET in agApi_natSetReadyshareName (libnat.so)
           ret = 1;
           break;
       [...]
       default:
           if (get_caller_info(&info) != 0) {
               pr_warn("Couldn't get caller info!");
           };
           pr_warn("[acos_nat_cli hack] %s (%d, %s+0x%lx) called unknown ioctl. cmd: 0x%x, arg: 0x%lx.\n",
proc_name, pid, info.lib_name, info.offset, cmd, arg);
           ret = 0;
           break;
   }
   return ret;
}
```

® CrowdStrike, Inc. All Rights Reserved

CROWDSTRIKE

Appendix: Useful Additions

→ Global `do_execve()` logging

→ `loglevel=8` to boot args

→ Helps to unpick errors

```
/ # /etc/system/wait[do_execve] pid=885 comm=sh
cmdline="/etc/system/wait"
[do_execve] pid=886 comm=wait cmdline="mkdir -p /tmp/etc/.root"
[do_execve] pid=887 comm=wait cmdline="chmod 711 /tmp/etc/.root"
[do_execve] pid=888 comm=wait cmdline="mkdir -p /tmp/udev/rules.d"
[do_execve] pid=889 comm=wait cmdline="mount -t cgroup -o memory
memory /cgroup"
[do_execve] pid=890 comm=wait cmdline="awk -F: /brcmnand/ { print
$1 } /proc/mtd"
[do_execve] pid=893 comm=wait cmdline="sed s/mtd/mtdblock/"
[do_execve] pid=894 comm=wait cmdline="mkdir -p /var/config"
[do_execve] pid=896 comm=wait cmdline="mtd_debug info /dev/mtd3"
[do_execve] pid=897 comm=wait cmdline="grep mtd.type"
[do_execve] pid=899 comm=wait cmdline="cut -d  -f2"
[do_execve] pid=898 comm=wait cmdline="cut -d= -f2"
...
```

```
int do_execve(const char * filename,
    const char __user *const __user *argv,
    const char __user *const __user *envp,
    struct pt_regs * regs)
{
    struct linux_binprm *bprm;
    struct file *file;
    struct files_struct *displaced;
    bool clear_in_exec;
    int retval;

    // ADDED - this is (somewhat) insane logging for all the execve calls the system makes!
    // But it seems to work without too much (if any?) overhead.
    char cmdline[MAX_CMDLINE_LEN];

    build_exec_cmdline(argv, cmdline, sizeof(cmdline));

    pr_debug("[do_execve] pid=%d comm=%s cmdline=\"%s\"\n",
        current->pid, current->comm, cmdline);
    // END ADDED
```

® CrowdStrike, Inc. All Rights Reserved