Reverse Engineering da Morte que Mata

Jonathan Brossard / CEO @ Moabi.com


ROADSEC
TL;DR
https://github.com/endrazine/wcc/

Who am I ?

- CEO at Moabi (San Francisco, Paris)
- Security Researcher : Bitlocker, MS IE/Edge, most BIOS Firmwares, SAP.
- "Inventor" of Hardware Backdooring (Rakshasa, 2011)
- Firmware Security Pioneer (INTEL-SA-00016)
- Previously Director of Offensive Security at Salesforce
- Speaker at Blackhat (x5), Defcon (x3)
- MS Engineering, MS C.S., PhD candidate
- More at https://endrazine.com

Symbolic execution at scale

- 128b taint analysis and symbolic execution
- Built for IIoT / Industrial Processes
- Scales to 100k+ binaries / day / client
- Covers entire SSDLC
- Finds 0days automatically

IIoT static analysis : a case study

Case Study : GeneIVI

GENIVI

Integrating the central and connected vehicle cockpit.

> Multi-OS Integration in the Centralized Vehicle Cockpit

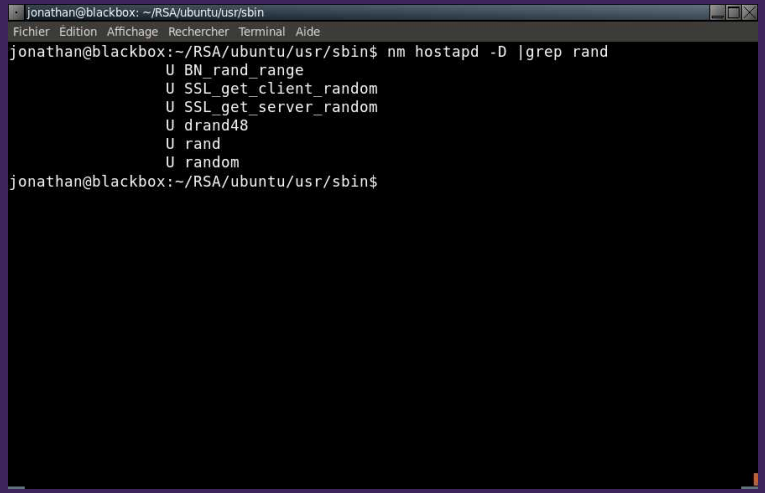
> Future Vehicle Architectures Driving GENIVI Work

Hostapd : Pseudo Random Number Generator

Vulnerability				
	Score: 8.00	Impact: 7	Confidence: 9	Risk: 10
Type	CWE-330: Use of Insufficiently Random Values			
Address	00081cf8			
function	00081cf8			
Description	Vulnerability when calling function rand() : call to rand() without initializing PRNG via srand() first. Cryptographic sequences will be predictable.			
Backtrace	#00 <81cf8> int rand(void) at: ./27a0432f7b54d70611f33f47bd9c0193e181c81b:0x81cf8 #01 <81cf8> function_81cf8() at: ./27a0432f7b54d70611f33f47bd9c0193e181c81b:0x81cf8 #02 <81404> function_81404() at: ./27a0432f7b54d70611f33f47bd9c0193e181c81b:0x81404			

Hostapd : CVE-2016-17043

Hostapd : Pseudo Random Number Generator



Use of rand() and random() without seeding PRNGS

Reference: https://cwe.mitre.org/data/definitions/330.html

Silently fixed in version 2.6 (2016)

Reported to CERT in 03/2019

Advisory: https://moabi.com/advisories/CVE-2019-10064.html

Hostapd : CVE-2016-10743

```c
unsigned long os_random(void)
{
        return random();
}
```

```c
/**
 * wps_generate_pin - Generate a random PIN
 * Returns: Eight digit PIN (i.e., including the checksum digit)
 */
unsigned int wps_generate_pin(void)
{
        unsigned int val;

        /* Generate seven random digits for the PIN */
        if (random_get_bytes((unsigned char *) &val, sizeof(val)) < 0) {
                struct os_time now;
                os_get_time(&now);
                val = os_random() ^ now.sec ^ now.usec;
        }
        val %= 10000000;

        /* Append checksum digit */
        return val * 10 + wps_pin_checksum(val);
}
```

Hostapd : CVE-2016-10743

```
int getRandomNumber()
{
    return 4;  // chosen by fair dice roll.
               // guaranteed to be random.
}
```

Source: https://imgs.xkcd.com/comics/random_number.png

Exploit prototyping

Bridging the gap between static analysis and exploitation


ROADSEC
DEMO TIME :
Analysing Hostapd CVE-2016-10743

BONUS

CVE-2019-1006 : DoS in EAP Server in Hostapd due to lack of entropy in PRNGs

ROADSEC

DEMO TIME :

Debugging ARM from x86_64 using WCC (wsh) and quemu JIT #NoVM

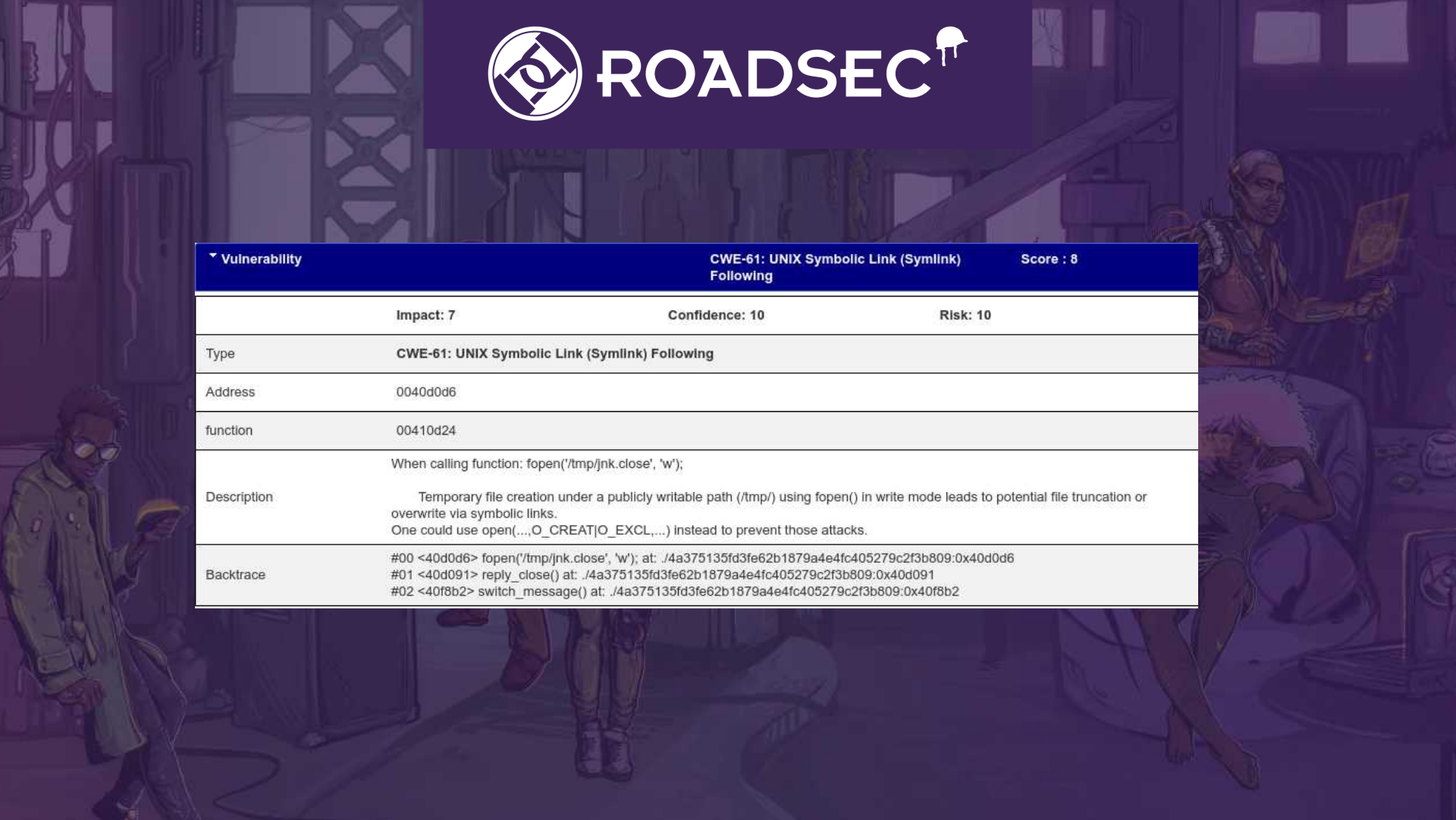
From static analysis to reliable exploit : Witchcraft (WCC)

- The binary is ARM
- How do we verify if the vulnerability exists ?
- Let's rely on the Witchcraft Compiler Collection (WCC)

The Witchcraft Compiler Collection (WCC)

- Open Source Software
- Introduced at DEFCON and BLACKHAT 2016
https://github.com/endrazine/wcc
- Rapid Cross platform analysis
- JIT binary translation from ARM to x86_64 thanks to qemu


ROADSEC
DEMO TIME :
Symlink attack in Samba
CVE-2015-5252



Vulnerability	CWE-61: UNIX Symbolic Link (Symlink) Following	Score : 8
	Impact: 7	**Confidence: 10**	**Risk: 10**
Type	**CWE-61: UNIX Symbolic Link (Symlink) Following**
Address	0040d0d6
function	00410d24
Description	When calling function: fopen('/tmp/jnk.close', 'w'); Temporary file creation under a publicly writable path (/tmp/) using fopen() in write mode leads to potential file truncation or overwrite via symbolic links. One could use open(...,O_CREAT|O_EXCL,...) instead to prevent those attacks.
Backtrace	#00 <40d0d6> fopen('/tmp/jnk.close', 'w'); at: ./4a375135fd3fe62b1879a4e4fc405279c2f3b809:0x40d0d6 #01 <40d091> reply_close() at: ./4a375135fd3fe62b1879a4e4fc405279c2f3b809:0x40d091 #02 <40f8b2> switch_message() at: ./4a375135fd3fe62b1879a4e4fc405279c2f3b809:0x40f8b2


ROADSEC
Obrigado!
endrazine@gmail.com
@endrazine
https://endrazine.com
MAIS UM EVENTO:
Flipside
SECURITY BEYOND TECHNOLOGY
REALIZAÇÃO:
Green Helmet