

A Comprehensive Review on the Less-Traveled Road

9 Years of Overlooked MikroTik Pre-Auth RCE

NiNi Chen



NiNi Chen

- Security researcher at **DEVCORE**
- Member of Balsn CTF Team
- Member of UNDEFINED

 @terrynini38514

 <https://blog.terrynini.tw>

 nini@undefined.zip

DEV**CORE**

Why?

Target	Vector	Cash Prize	Master of Pwn Points
TP-Link AX1800 WiFi 6 Router (Archer AX21)	WAN Side	\$20,000 (USD)	2
	LAN Side	\$5,000 (USD)	1
NETGEAR Nighthawk WiFi6 Router (RAX30 AX2400)	WAN Side	\$20,000 (USD)	2
	LAN Side	\$5,000 (USD)	1
Synology RT6600ax	WAN Side	\$20,000 (USD)	2
	LAN Side	\$5,000 (USD)	1
Cisco Integrated Service Router C921-4P	WAN Side	\$30,000 (USD)	3
	LAN Side	\$15,000 (USD)	2
Mikrotik RouterBoard RB2011UiAS-IN	WAN Side	\$30,000 (USD)	3
	LAN Side	\$15,000 (USD)	2
Ubiquiti Networks EdgeRouter X SFP	WAN Side	\$30,000 (USD)	3
	LAN Side	\$15,000 (USD)	2

Why?

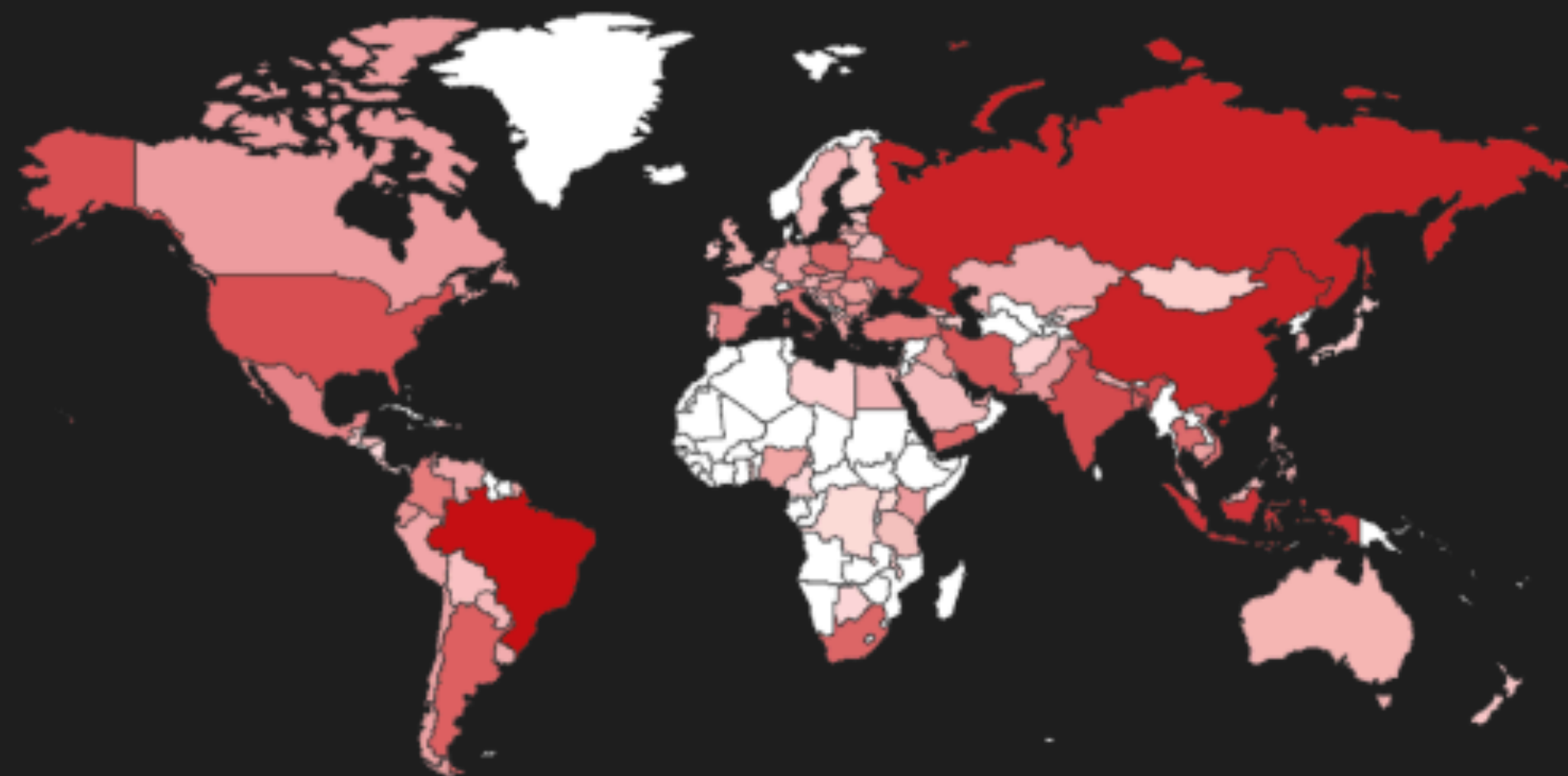
Target		Point
Initial Stage	Final Stage	10
TP-Link AX1800 WiFi 6 Router NETGEAR Nighthawk WiFi6 Router Synology RT6600ax Cisco Integrated Service Router C921-4P Mikrotik RouterBoard RB2011UiAS-IN Ubiquiti Networks EdgeRouter X SFP	Meta Portal Go Amazon Echo Show 15 Google Nest Max Sonos One Speaker Apple HomePod mini Amazon Echo Studio HP Color LaserJet Pro M479fdw Lexmark MC3224i Canon imageCLASS MF743Cdw Synology DiskStation DS920+ My Cloud Pro Series PR4100 from WD	

RouterOS

TOTAL RESULTS

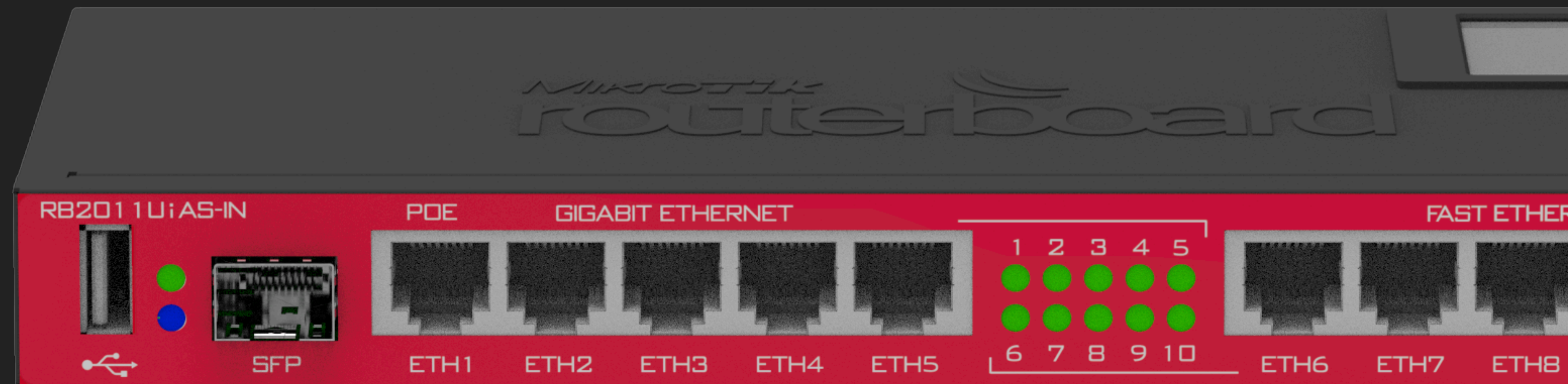
3,032,059

TOP COUNTRIES



RouterOS

- A stand-alone operating system based on the Linux kernel
- Also available for virtual machines to turn a PC into a router



RouterOS

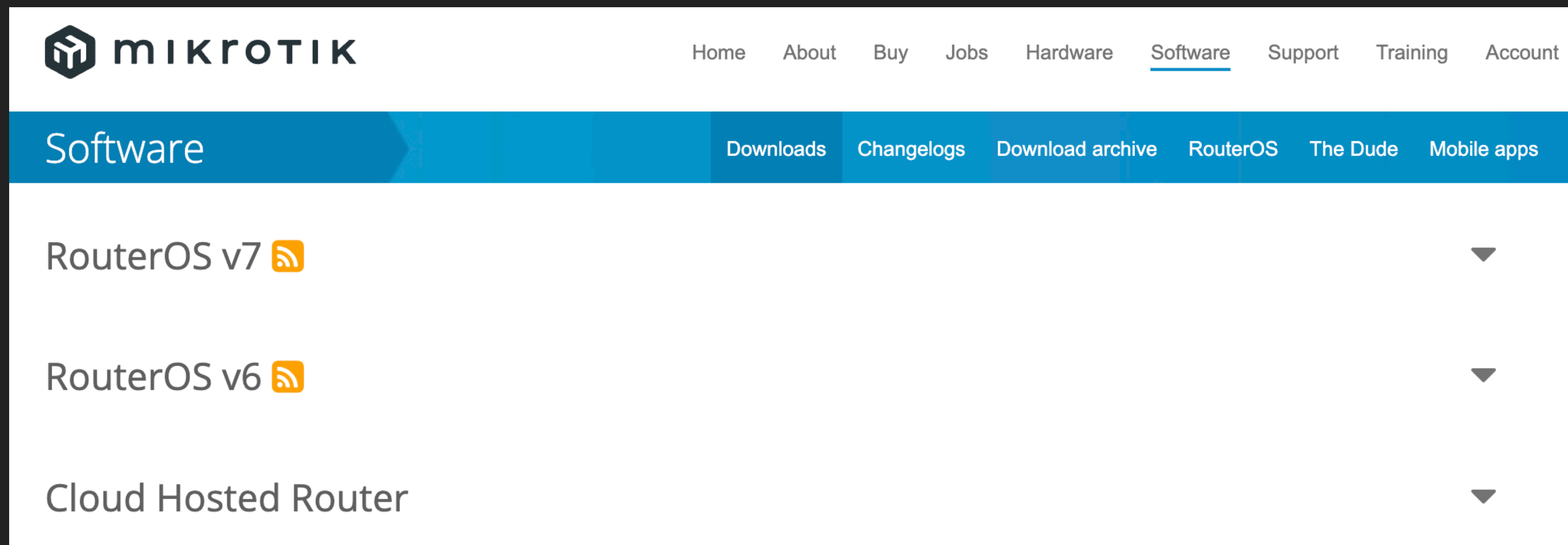
- A stand-alone operating system based on the Linux kernel
- Also available for virtual machines to turn a PC into a router
- Closed source and also a closed ecosystem.
(It is said that you can get GPL sources used in RouterOS if you ask them)

To get a CD with the corresponding source code for the GPL-covered programs in this distribution, wire transfer \$45 to MikroTiks SIA, Ūnijas iela 2, Rīga, LV-1039, Latvia. Please contact MikroTiks SIA for our current account information and wire transfer instructions. Offer valid for three years from the date of distribution of this software. This CD will only include the source code of the following programs and any non-proprietary programs distributed according to license requirements. This CD will not include MikroTiks proprietary SOFTWARE.

<https://mikrotik.com/downloadterms.html>

RouterOS

- RouterOS v6 and RouterOS v7 can be considered two different branches
- CHR is the RouterOS image for VM, x86_64 only



The screenshot shows the Mikrotik website's software page. The top navigation bar includes links for Home, About, Buy, Jobs, Hardware, Software (underlined), Support, Training, and Account. Below this is a blue sub-navigation bar with links for Software, Downloads, Changelogs, Download archive, RouterOS, The Dude, and Mobile apps. The main content area lists three software options: RouterOS v7, RouterOS v6, and Cloud Hosted Router, each with a feed icon and a dropdown arrow.

RouterOS

- Most binaries are “Nova binary”
- No official method to access the linux system.

```
Terminal — 117x26

MMM      MMM      KKK      TTTTTTTTTTT      KKK
MMMM     MMMM     KKK      TTTTTTTTTTT      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 6.49.6 (c) 1999–2022      http://www.mikrotik.com/

[?]          Gives the list of available commands
command [?]  Gives help on the command and list of arguments

[Tab]       Completes the command/word. If the input is ambiguous,
            a second [Tab] gives possible options

/           Move up to base level
..         Move up one level
/command   Use command at the base level
jan/02/1970 00:09:21 system,error,critical login failure for user 1\7Fadmin from 192

[admin@MikroTik] > █
```

Winbox

- Winbox is a native Win32 GUI binary used for managing and configuring MikroTik routers on Windows.

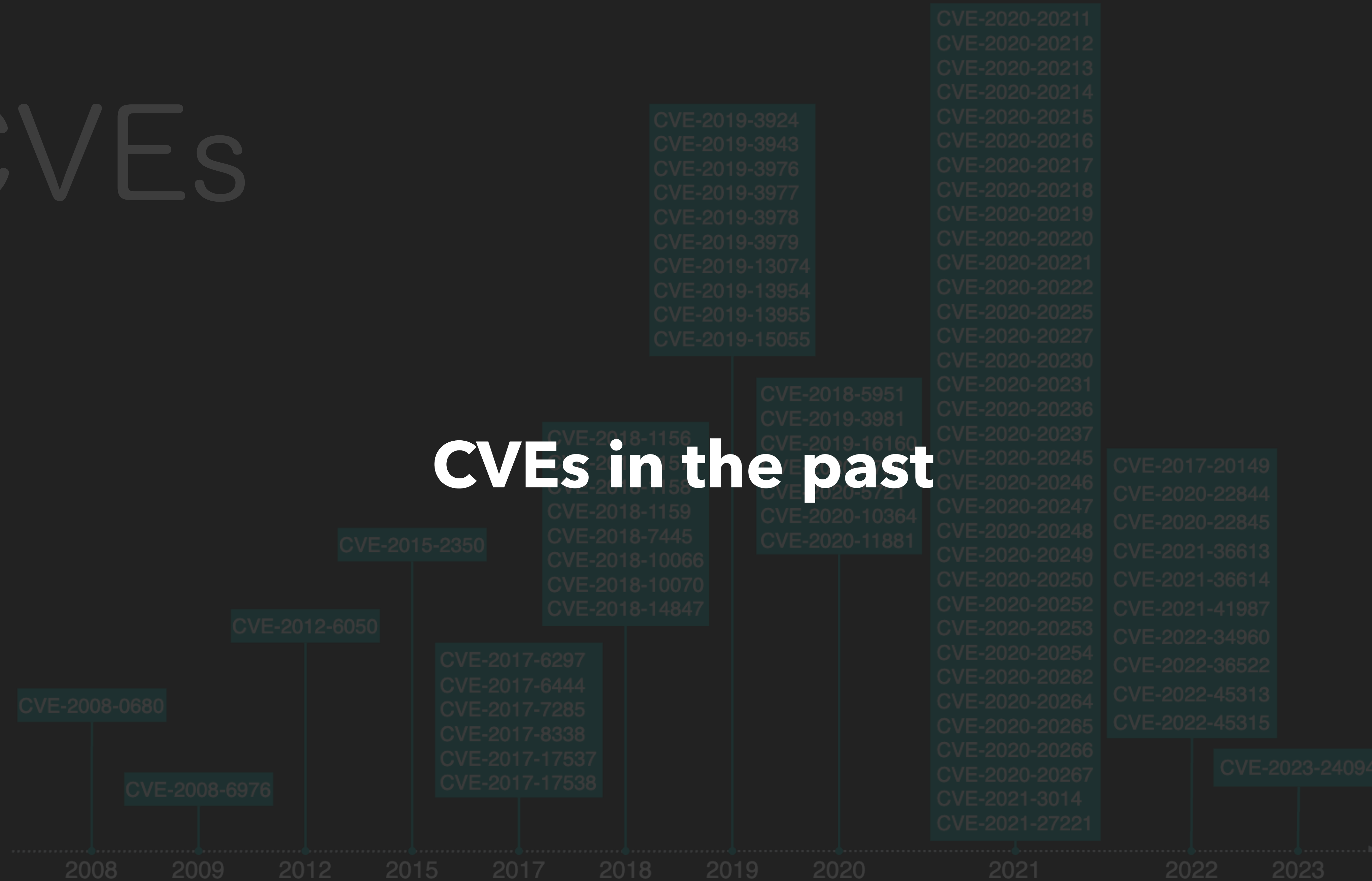
The screenshot displays the WinBox v6.49.6 interface for a MikroTik router. The main window shows the 'Interface List' tab, which contains a table of network interfaces. Below this, the 'Bridge' configuration window is open, showing a table of bridge members.

Interface	Name	Type	Actual MTU	L2 MTU	Tx	Rx	Tx Packet (p/s)	Rx Packet (p/s)
R	bridge	Bridge	1500	1598	120.2 kbps	5.0 kbps	11	0
RS	ether1	Ethernet	1500	1598	0 bps	0 bps	0	0
S	ether2	Ethernet	1500	1598	120.6 kbps	5.2 kbps	11	0
S	ether3	Ethernet	1500	1598	0 bps	0 bps	0	0
S	ether4	Ethernet	1500	1598	0 bps	0 bps	0	0
S	ether5	Ethernet	1500	1598	0 bps	0 bps	0	0
S	ether6	Ethernet	1500	1598	0 bps	0 bps	0	0
S	ether7	Ethernet	1500	1598	0 bps	0 bps	0	0
S	ether8	Ethernet	1500	1598	0 bps	0 bps	0	0
S	ether9	Ethernet	1500	1598	0 bps	0 bps	0	0
S	ether10	Ethernet	1500	1598	0 bps	0 bps	0	0
S	sfp1	Ethernet	1500	1598	0 bps	0 bps	0	0

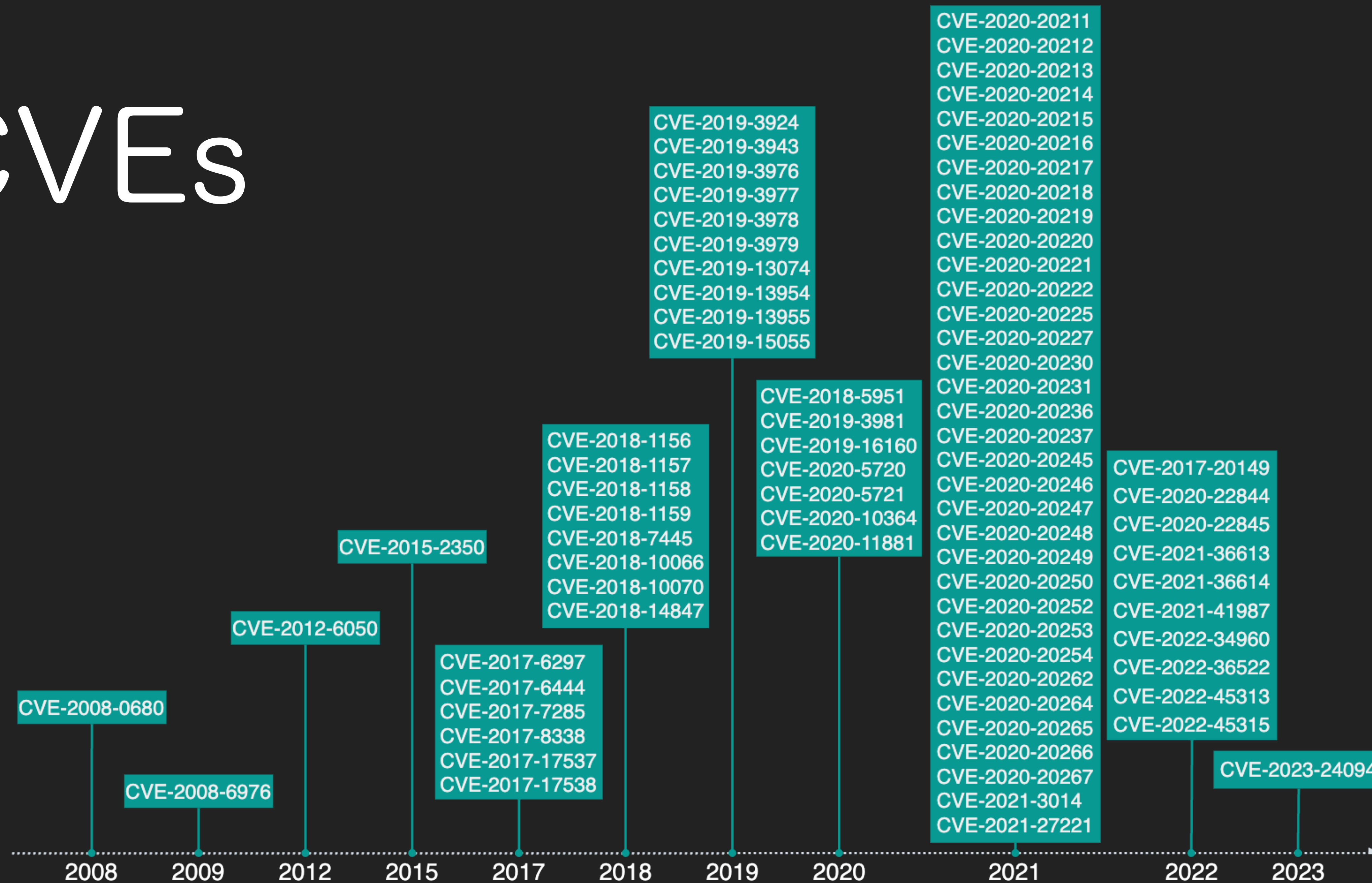
Name	Type	Mirror Source	Mirror Target
switch1	Atheros 8327		
switch2	Atheros 8227		

CVEs

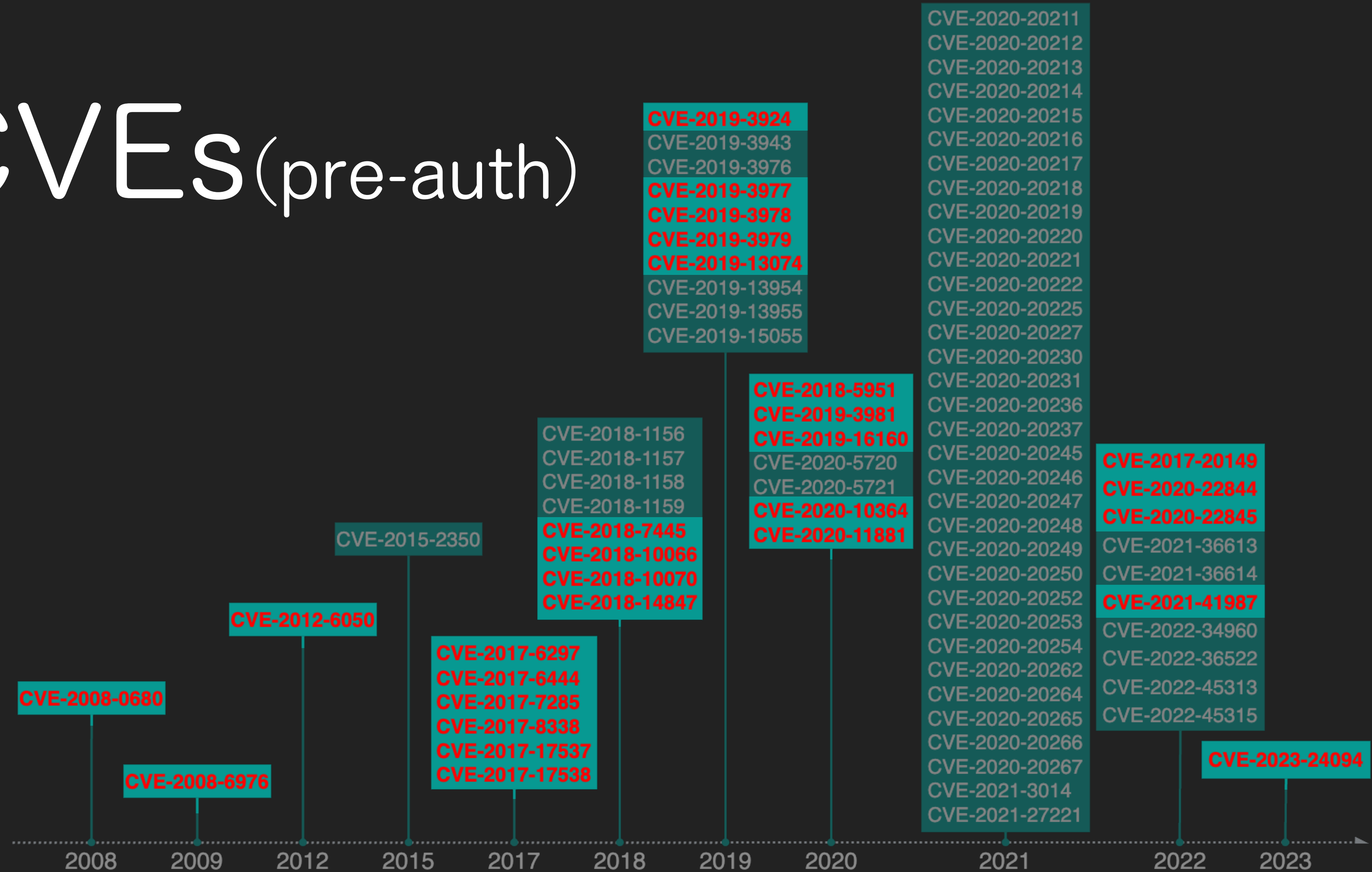
CVEs in the past



CVEs



CVEs (pre-auth)



CVEs (pre-auth)

DoS

FTP

CVE-2019-3924
CVE-2019-3943
CVE-2019-3976
CVE-2019-3977
CVE-2019-3978
CVE-2019-3979

CVE-2019-13074

CVE-2019-13954
CVE-2019-13955
CVE-2019-15055

EoIPv6

CVE-2018-5951

CVE-2019-3981

CVE-2019-16160

SMB

CVE-2018-1157
CVE-2018-1158
CVE-2018-1159

CVE-2020-5720

CVE-2020-5721

SSH

CVE-2020-10364

CVE-2020-11881

CVE-2015-2350

winbox

CVE-2012-6050

FTP

CVE-2018-10070

CVE-2018-14847

SMB

snmp

CVE-2008-0680

CVE-2008-6976

CVE-2017-6297

CVE-2017-6444

CVE-2017-7285

CVE-2017-8338

CVE-2017-17537

CVE-2017-17538

TCP/UDP/DNS/ICMP

CVE-2020-20211
CVE-2020-20212
CVE-2020-20213
CVE-2020-20214
CVE-2020-20215
CVE-2020-20216
CVE-2020-20217
CVE-2020-20218
CVE-2020-20219
CVE-2020-20220
CVE-2020-20221
CVE-2020-20222
CVE-2020-20225
CVE-2020-20227
CVE-2020-20230
CVE-2020-20231
CVE-2020-20236
CVE-2020-20237
CVE-2020-20245
CVE-2020-20246
CVE-2020-20247
CVE-2020-20248
CVE-2020-20249
CVE-2020-20250
CVE-2020-20252
CVE-2020-20253
CVE-2020-20254
CVE-2020-20262
CVE-2020-20264
CVE-2020-20265
CVE-2020-20266
CVE-2020-20267
CVE-2021-3014
CVE-2021-27221

CVE

SMB

CVE-2020-22844

CVE-2020-22845

FTP

CVE-2021-36613

CVE-2021-36614

CVE-2021-41987

CVE-2022-34960

CVE-2022-36522

CVE-2022-45313

CVE-2022-45314

bridge2

CVE-2023-24094

2008

2009

2012

2015

2017

2018

2019

2020

2021

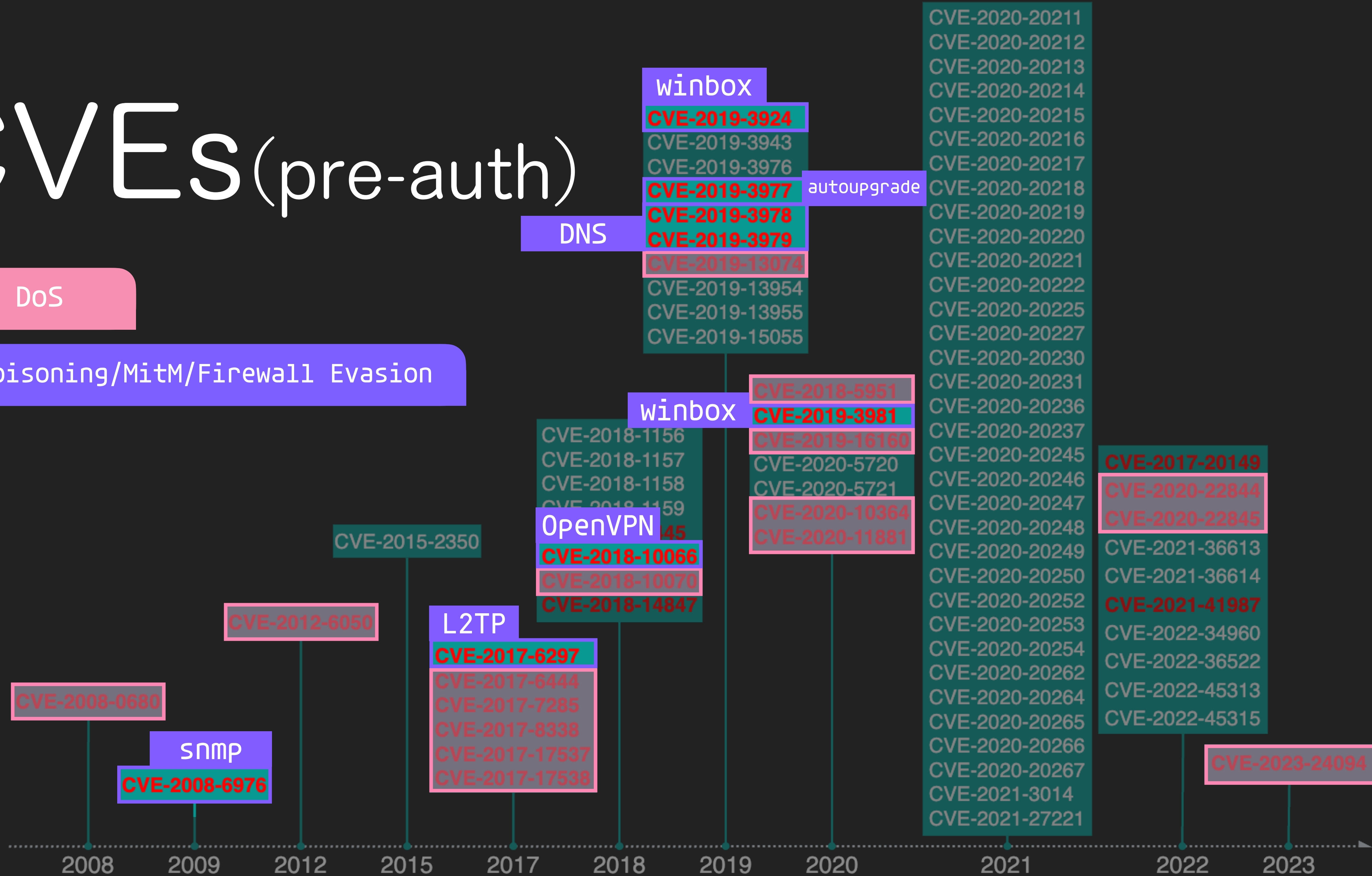
2022

2023

CVEs (pre-auth)

DoS

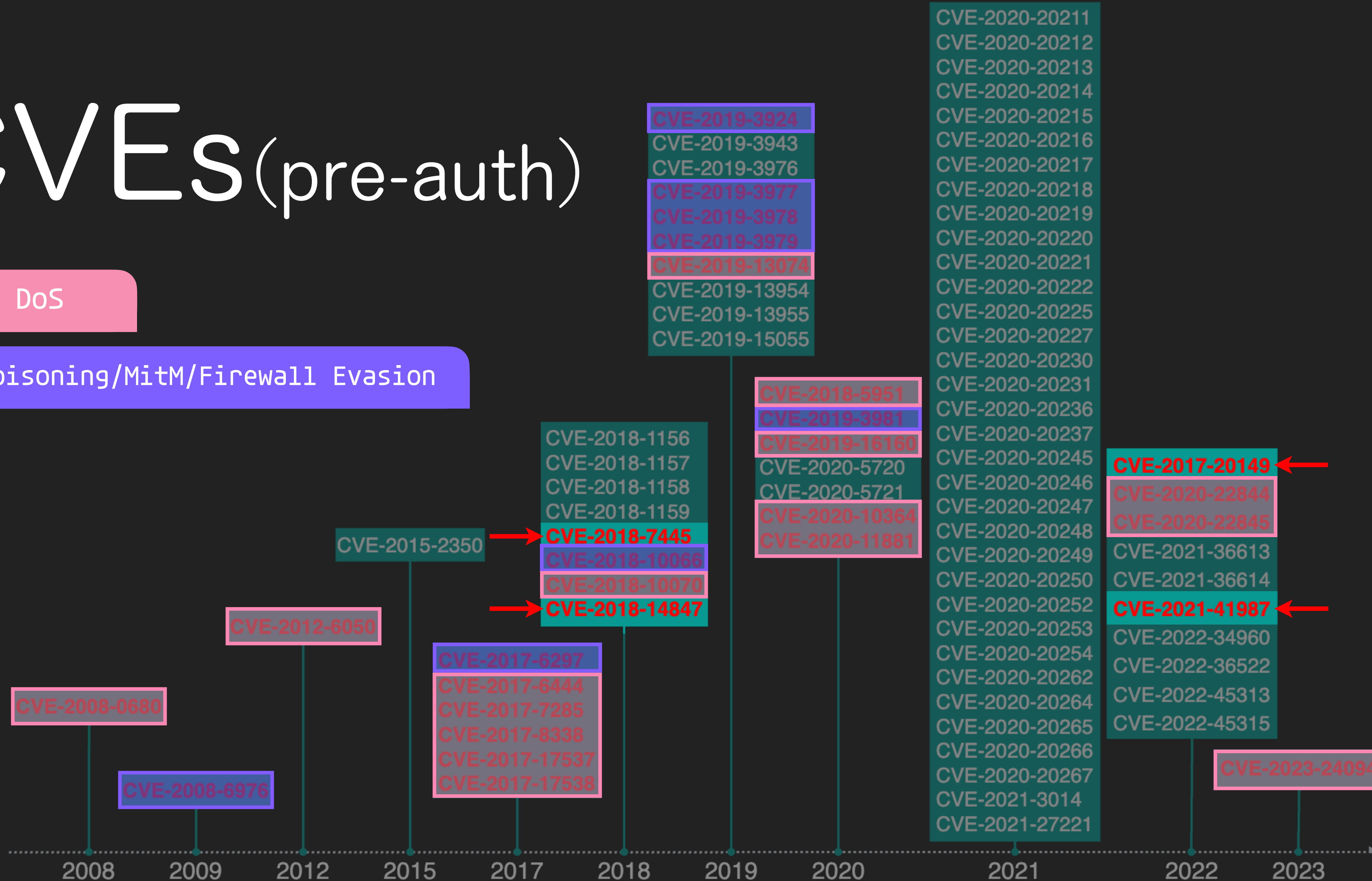
Poisoning/MitM/Firewall Evasion



CVEs (pre-auth)

DoS

Poisoning/MitM/Firewall Evasion



CVEs (pre-auth)

CVE-2017-20149

CVE-2020-22844

CVE-2020-22845

CVE-2021-36613



- aka Chimay-Red
- One of the exploits leaked from the CIA
- Lacking input validation on Content-Length in HTTP request cause the stack clash attack.

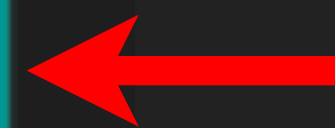
CVEs (pre-auth)

CVE-2018-7445

CVE-2018-10066

CVE-2018-10070

CVE-2018-14847



- Buffer overflow in SMB
- It was found by blackbox fuzzing

CVEs (pre-auth)

CVE-2018-7445

CVE-2018-10066

CVE-2018-10070

CVE-2018-14847

- One of the exploits leaked from the CIA
- A directory traversal vulnerability in the WinBox interface
- Allows unauthenticated attackers to read arbitrary files on RouterOS

CVEs (pre-auth)

CVE-2018-7445

```
password_db = password_raw xor md5(username + "283i4jfkai3389")
```

CVE-2018-14847

- Allows unauthenticated attackers to read arbitrary files on Router0S

CVEs (pre-auth)

CVE-2021-36614

CVE-2021-41987

CVE-2022-34960

- 00B in the base64decode of the SCEP service
- The attacker must know the scep_server_name
- It was discovered on an APT's C2 server

2017

Kirils

Rooting the MikroTik routers

focus on jailbreak

Kirils

A deeper journey into MikroTik routers

focus on jailbreak

Kirils

Tools for effortless reverse engineering of MikroTik router

focus on jailbreak

2018

Jacob Baines

Bug Hunting in RouterOS

focus on nova message in IPC

2019

Jacob Baines

Make It Rain with MikroTik

focus on nova message in IPC

Maximiliano Vidal, Juan Caillava

Finding and exploiting CVE-2018-7445

find pre-auth RCE in SMB
by blackbox fuzzing

Jacob Baines

Help Me Vulnerabilities
You're My Only Hope

focus on jailbreak

Jacob Baines

MikroTik Firewall & NAT Bypass

Firewall bypass (winbox)

Tomas Kirnak

Deep-dive:
MikroTik exploits - a security analysis

analyze the CIA exploits

Jacob Baines

RouterOS: Chain to Root

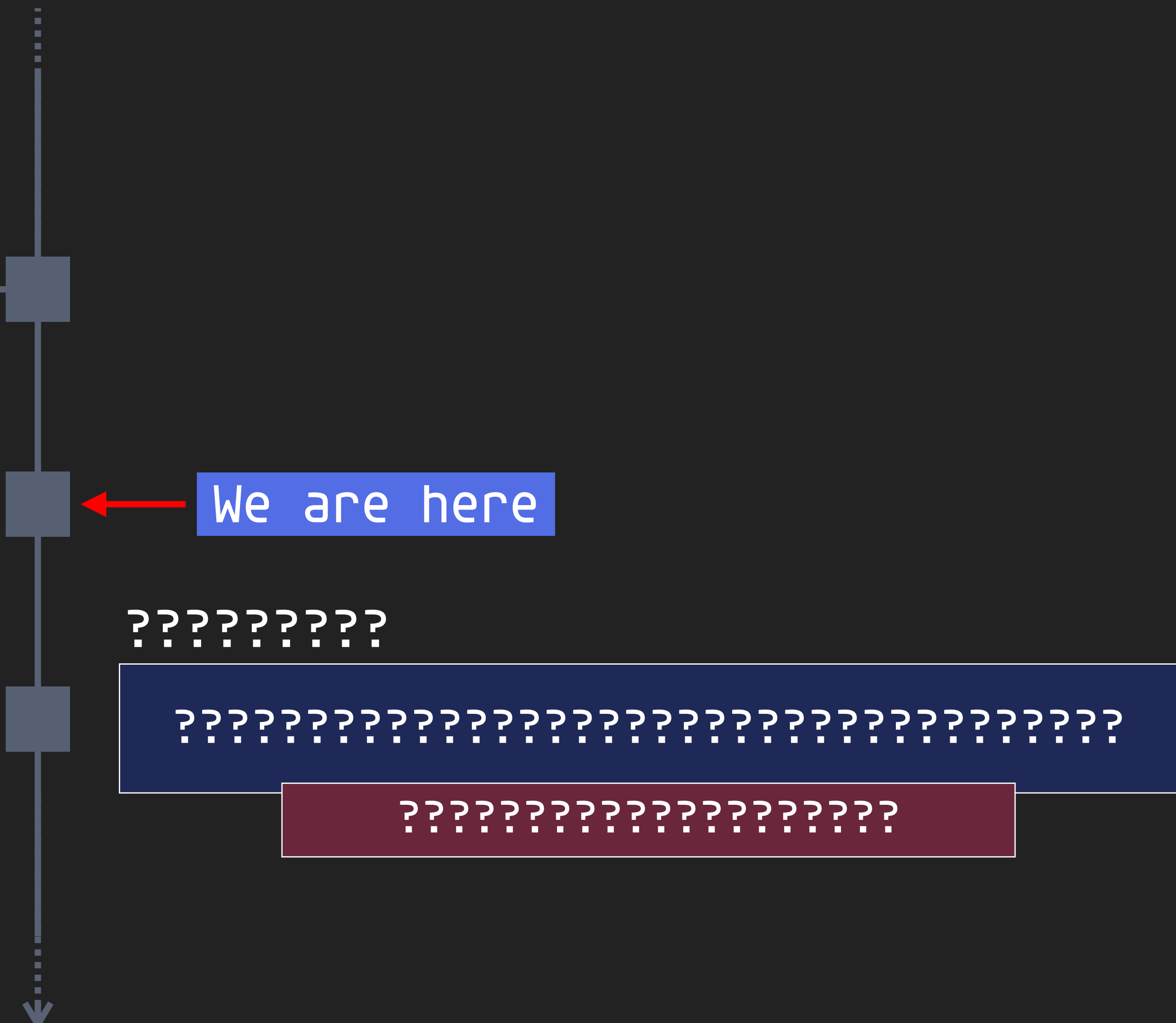
DNS poisoning (winbox)

2022

Ian Dupont, Harrison Green

Pulling MikroTik into the Limelight

focus on nova message in IPC



IPC

Nova binary

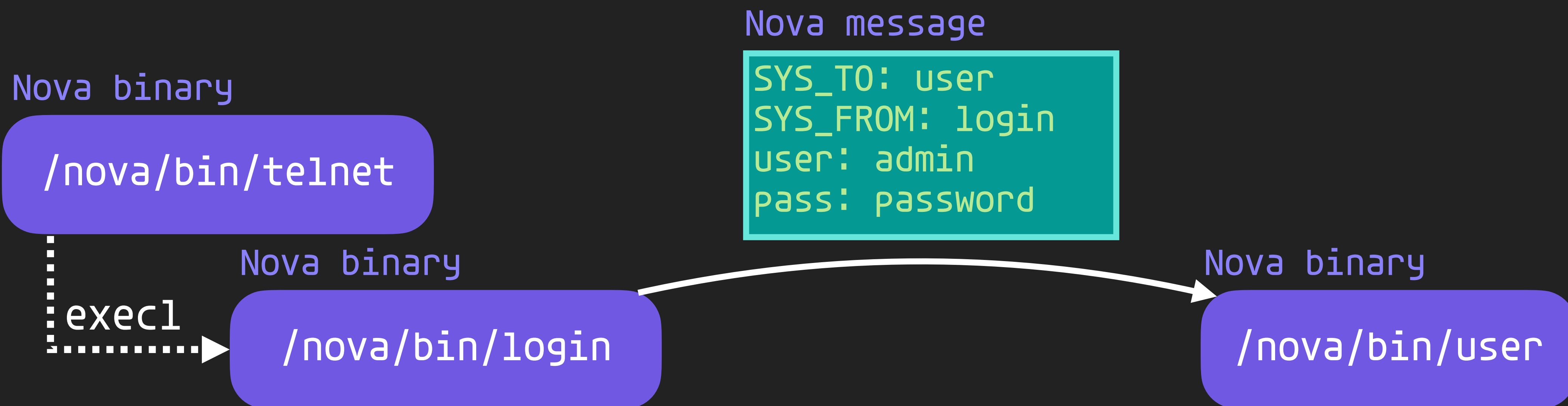
`/nova/bin/telnet`

`exec1`

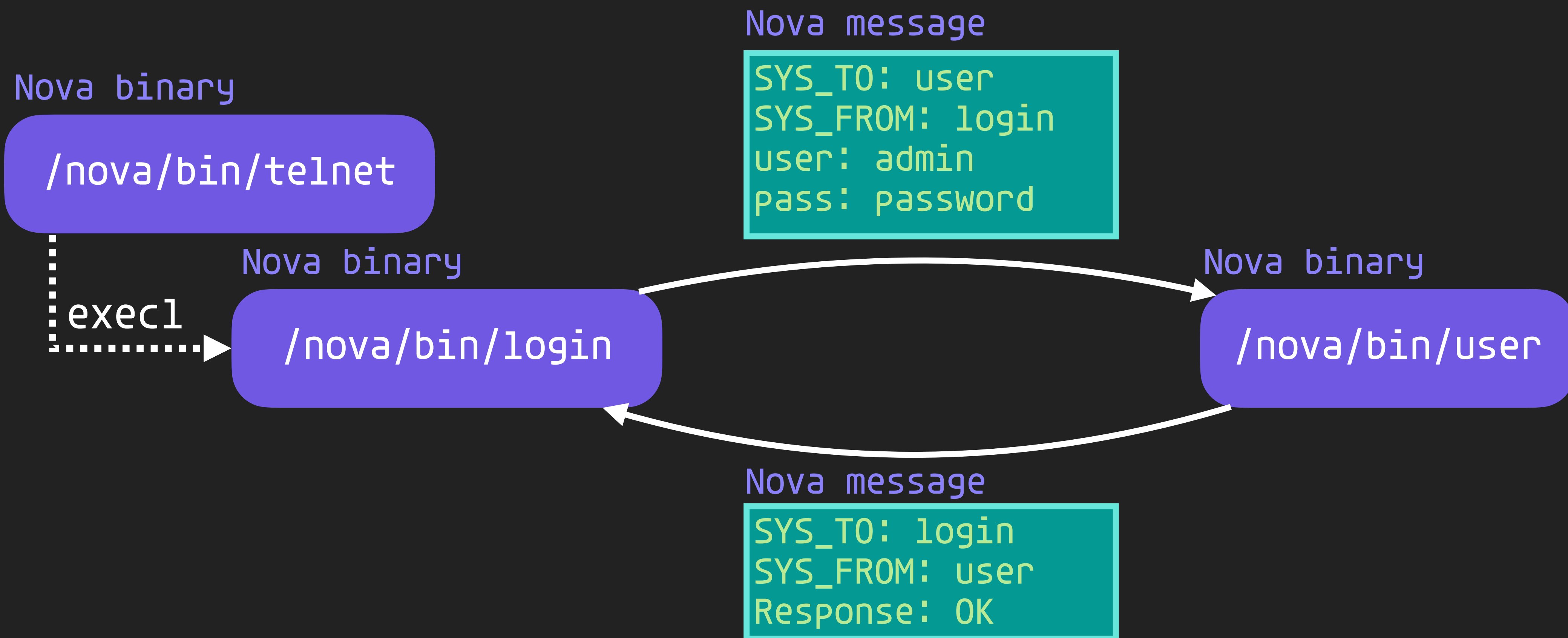
Nova binary

`/nova/bin/login`

IPC



IPC



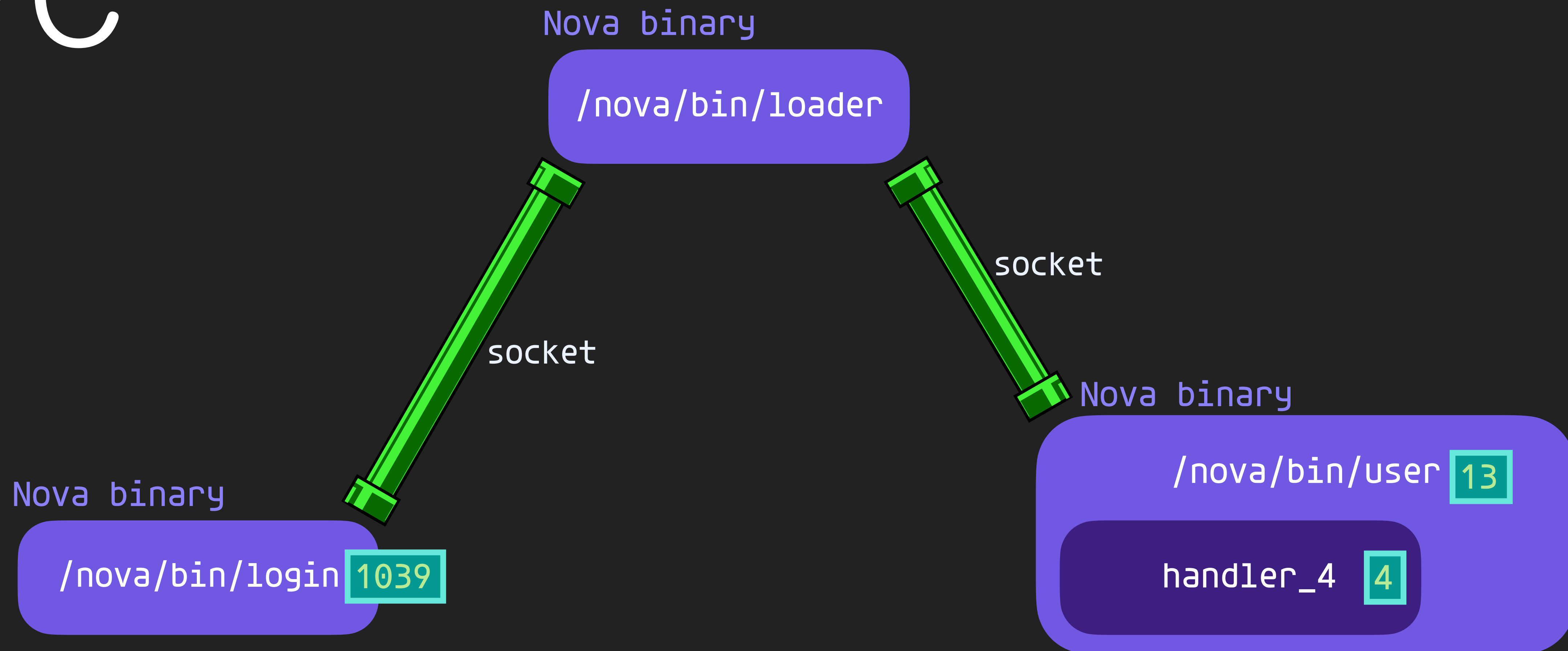
Nova Message

```
SYS_TO: user  
SYS_FROM: login  
user: admin  
pass: password
```

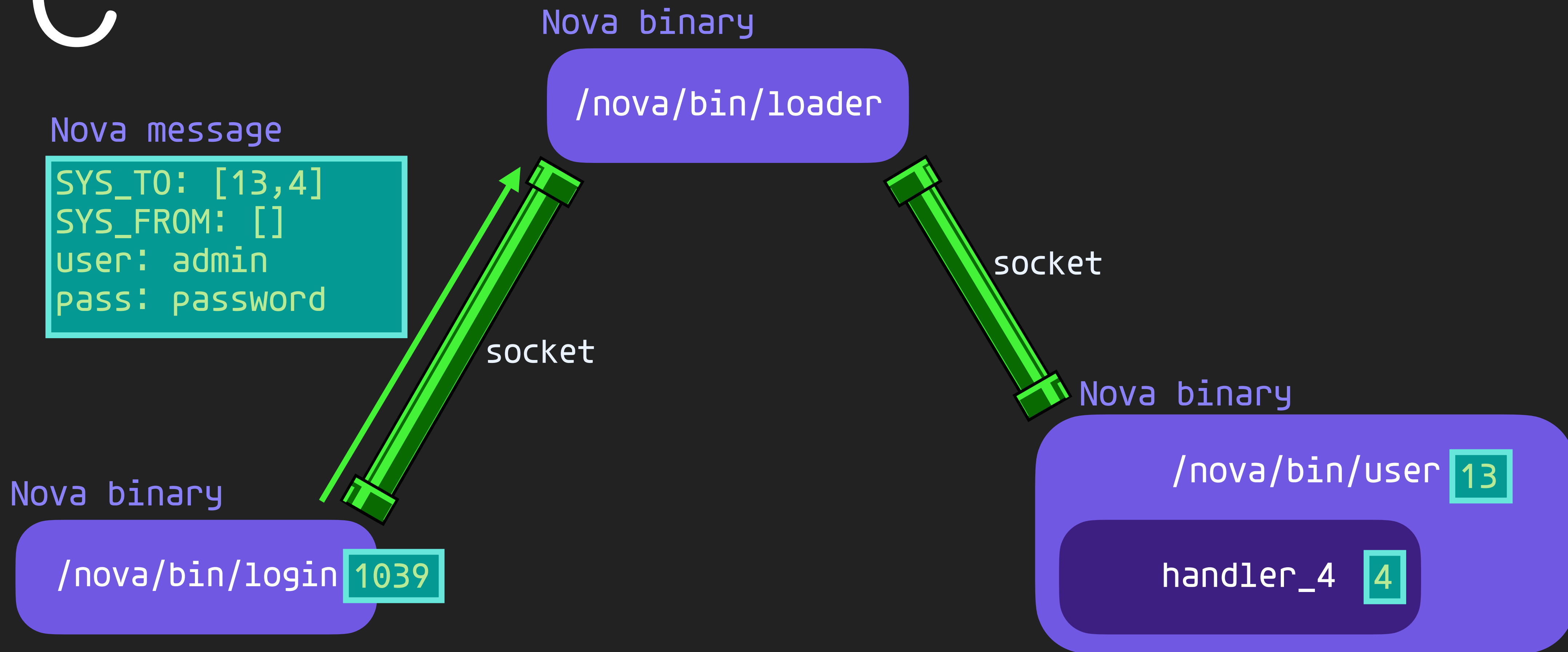


```
0xFF0001: [13,4]  
0xFF0002: [1039]  
1: admin  
3: password
```

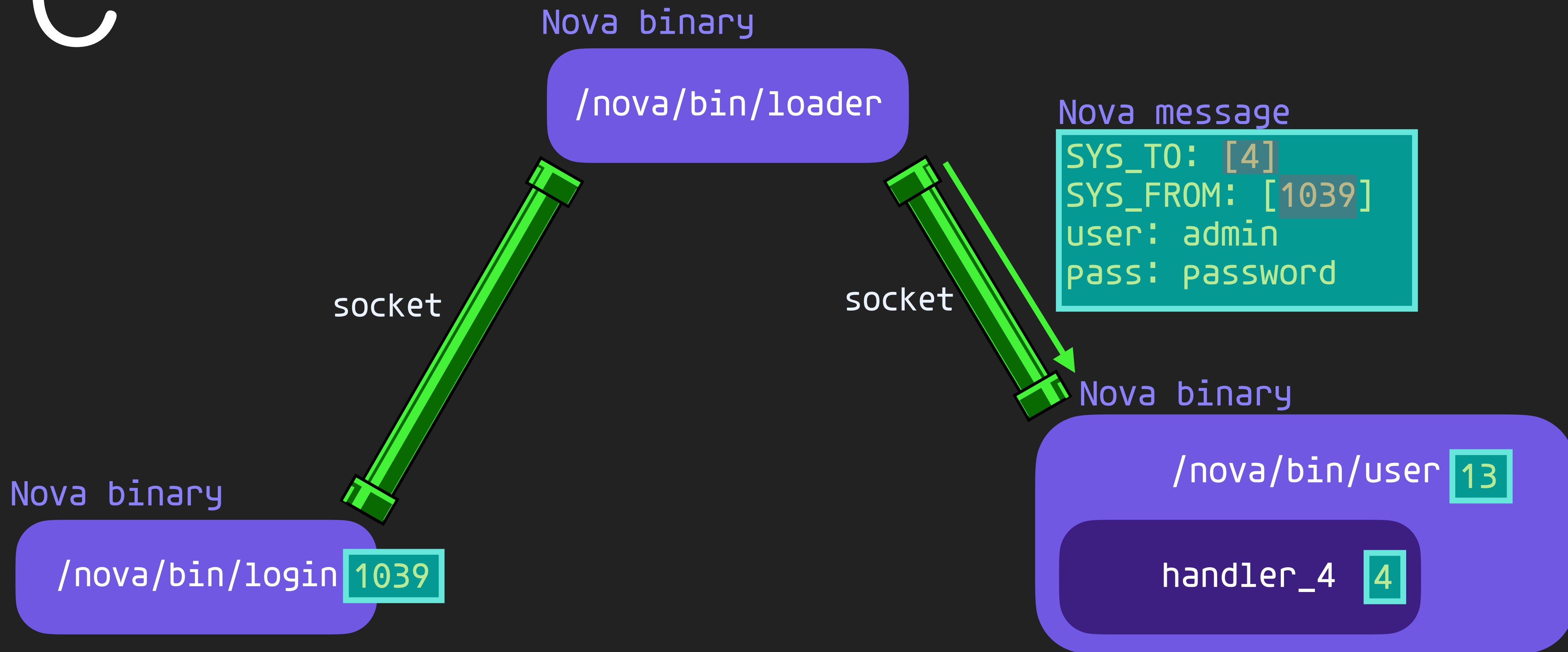
IPC



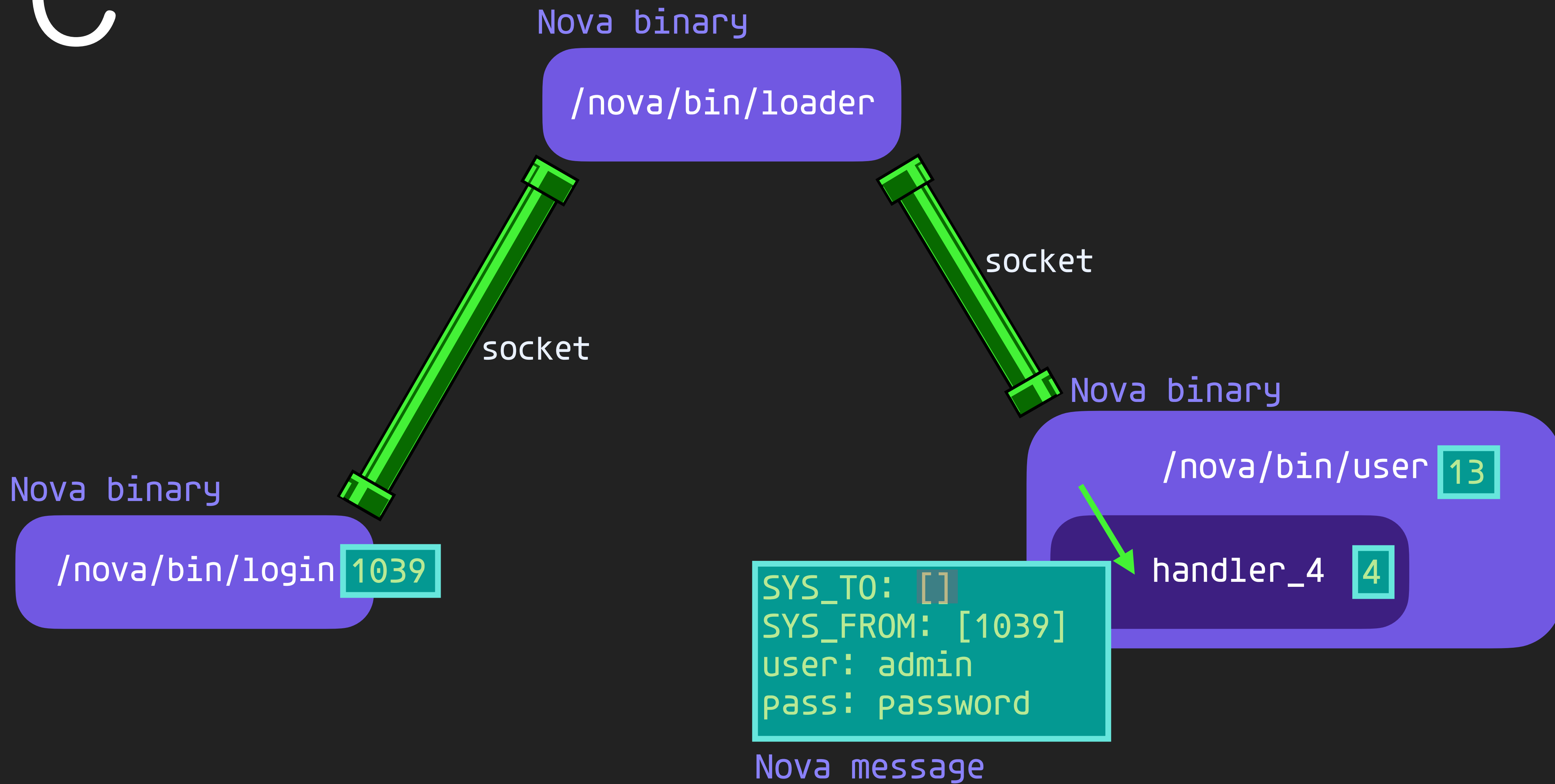
IPC



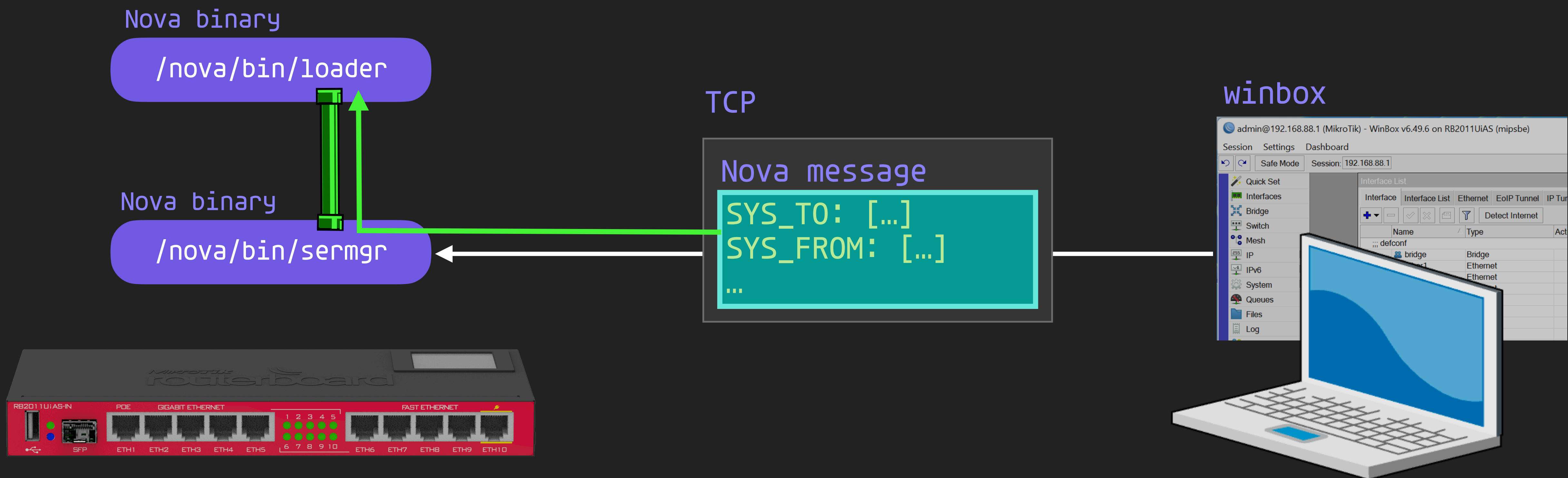
IPC



IPC



IPC



Nova Message

- The message used in IPC is constructed by `nv::message` and relative functions
- Nova message is typed key-value mapping and can contain u32, u64, bool, string, bytes, IP, and nova message.

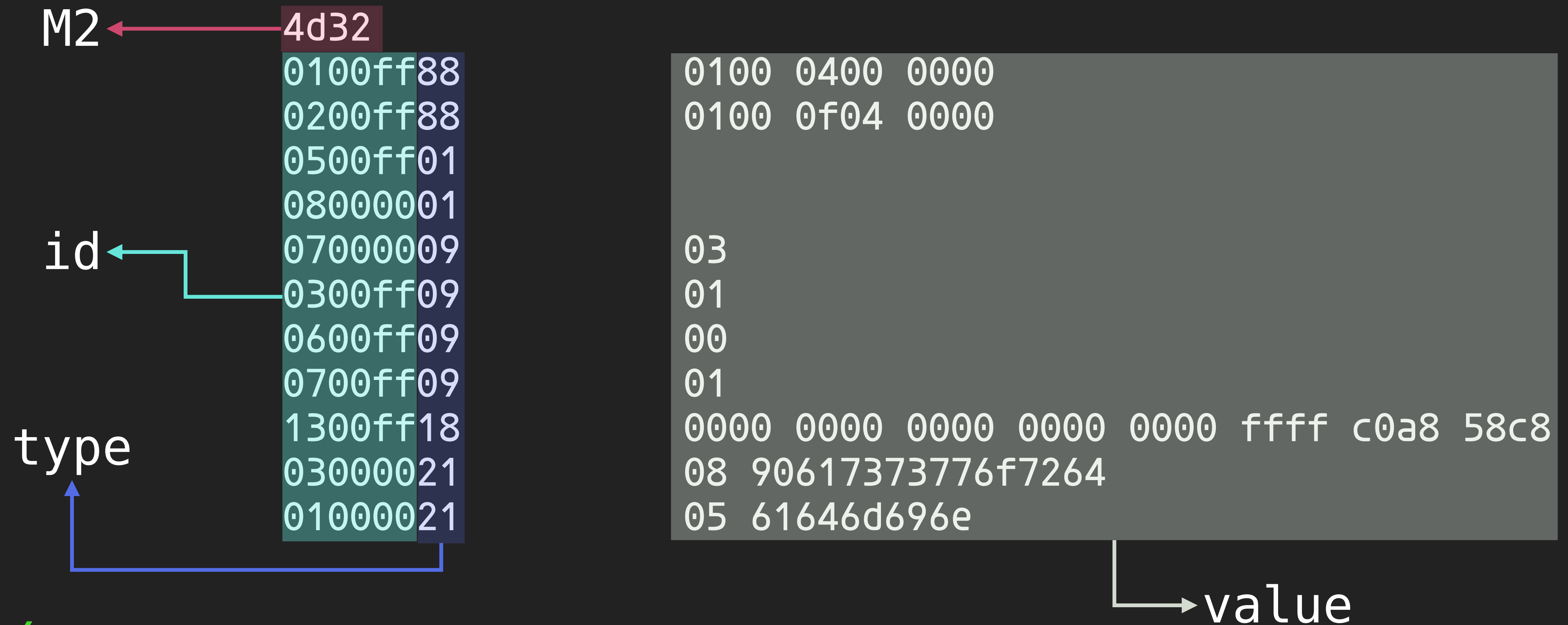
```
nv::message::message((nv::message *)&v121);
v87 = v82 + 4;
nv::message::insert_vector((int)&v121, 0xFF0001, 13, 4);
nv::message::insert<nv::string_id>(&v121, 1, v87); // account
nv::message::insert<nv::string_id>(&v121, 3, v75 + 4); // password
nv::message::insert<nv::u32_id>((int)&v121, 7, 9);
nv::message::insert<nv::addr6_id>(&v121, 23, *(_DWORD *) (a2 + 32) + 152);
nv::message::insert<nv::bool_id>(&v121, 8, 1);
```

Nova Message

```
0xFF0001: [13,4]
0xFF0002: [1039]
1: admin
3: password
```

```
00000000: 5d00 0000 4d32 0100 ff88 0100 0400 0000 ]...M2.....
00000010: 0200 ff88 0100 0f04 0000 0500 ff01 0800 .....
00000020: 0001 0700 0009 0303 00ff 0901 0600 ff09 .....
00000030: 0007 00ff 0901 1300 ff18 0000 0000 0000 .....
00000040: 0000 0000 ffff c0a8 58c8 0300 0021 0870 .....X...!.p
00000050: 6173 7377 6f72 6401 0000 2105 6164 6d69 assword...!.admi
00000060: 6e n
```

Nova Message



Nova Message

M2 ← 4d32

0xff0001	0100ff88	u32	array	0100	0400	0000	['4(0x4)']
0xff0002	0200ff88	u32	array	0100	0f04	0000	['1039(0x40f)']
0xff0005	0500ff01	bool					True
0x8	08000001	bool					True
id ← 0x7	07000009	u32		03			3(0x3)
0xff0003	0300ff09	u32		01			1(0x1)
0xff0006	0600ff09	u32		00			0(0x0)
0xff0007	0700ff09	u32		01			1(0x1)
0xff0013	1300ff18	IPv6		0000	0000	0000	<too long, skip>
type ↑ 0x3	03000021	string		08	90617373776f7264		b'password'
0x1	01000021	string		05	61646d696e		b'admin'

value →

Nova Message

- To understand which binary corresponds to the id in the SYS_FROM or SYS_TO of the nova message, we need to parse the *.x3 file under the /nova/etc/loader/system.x3

```
<33>
<30 (7)=b'/nova/bin/log' (4)=i32 3 (153)=b'\x01' (173)=b'\x01'
<30 (7)=b'/nova/bin/radius' (4)=i32 5/>
<30 (7)=b'/nova/bin/moduler' (4)=i32 6 (153)=b'\x01' (173)=b'\
<30 (7)=b'/nova/bin/user' (4)=i32 13 (204)=b'\x01' />
<30 (7)=b'/nova/bin/resolver' (4)=i32 14 (173)=b'\x01' />
<30 (7)=b'/nova/bin/mactel' (4)=i32 15 (173)=b'\x01' />
<30 (7)=b'/nova/bin/undo' (4)=i32 17/>
```

Nova Message

- If the binary was introduced by installing a package, its id is in the `/ram/pckg/<package_name>/nova/etc/loader/<package_name>.x3`

```
<33>  
  <30 (7)=b'/nova/bin/ippool6' (4)=i32 30 (153)=b'\x01' />  
  <30 (7)=b'/nova/bin/radvd' (4)=i32 31 (153)=b'\x01' />  
</33>%
```

Nova Message

- If the binary was introduced by install a packet, it's id is in the /ram/pckg/<package_name>/nova/etc/loader/<package_name>.x

```
<33>  
  <30 (7)=b'/no  
  <30 (7)=b'/no  
</33>%  
  0 (153)=b'\x01' />  
(153)=b'\x01' />
```

```
SYS_TO: [13, 4]
```

```
SYS_FROM: [1039]
```

```
user: admin
```

```
pass: password
```

Nova Message

- Other binaries also have their .x3 files for different purposes.

```
<169 (2)=b'www'>
  <154 (38)=b'index' (7)=b'/' (40)=b'\x01' />
  <154 (38)=b'jsproxy' (7)=b'/jsproxy' />
  <154 (38)=b'dir' (7)=b'/img/' (28)=b'/home/web/img' />
  <154 (38)=b'dir' (7)=b'/doc/' (28)=b'/home/web/doc' />
  <154 (38)=b'dir' (7)=b'/help/' (28)=b'/home/web/help' />
  <154 (38)=b'dir' (7)=b'/webfig/list' (28)=b'/home/web/webfig/list' />
  <154 (38)=b'dir' (7)=b'/webfig/' (28)=b'/home/web/webfig' (283)=b'\x01' />
  <154 (38)=b'winbox' (7)=b'/winbox' (40)=b'\x01' />
  <154 (38)=b'webgraph' (7)=b'/graphs' />
  <154 (38)=b'kidcontrol' (7)=b'/kid-control' (40)=b'\x01' />
  <154 (38)=b'dir' (7)=b'/winbox/' (28)=b'/home/web/winbox' />
  <154 (38)=b'traflog' (7)=b'/accounting/ip.cgi' (40)=b'\x01' />
  <154 (38)=b'dir' (7)=b'/' (28)=b'/home/web' />
  <154 (38)=b'dir' (7)=b'/crl' (28)=b'/var/cm/ca_crl' />
  <154 (38)=b'scep' (7)=b'/scep' />
</169>
```

Nova Message

- It seems like it's a good target for fuzzing.
- But we can't just fuzz it and expect we can get a pre-auth RCE after two months.

```
00000000: 5d00 0000 4d32 0100 ff88 0100 0400 0000 ]...M2.....
00000010: 0200 ff88 0100 0f04 0000 0500 ff01 0800 .....
00000020: 0001 0700 0009 0303 00ff 0901 0600 ff09 .....
00000030: 0007 00ff 0901 1300 ff18 0000 0000 0000 .....
00000040: 0000 0000 ffff c0a8 58c8 0300 0021 0870 .....X...!.p
00000050: 6173 7377 6f72 6401 0000 2105 6164 6d69 assword...!.admi
00000060: 6e n
```

2022

Spoiler! Spoiler! Spoiler! Spoiler!

Spoiler! Spoiler! Spoiler! Spoiler!

Spoiler! Spoiler! Spoiler! Spoiler!

2022

Ian Dupont, Harrison Green

Pulling MikroTik into the Limelight

focus on nova message in IPC

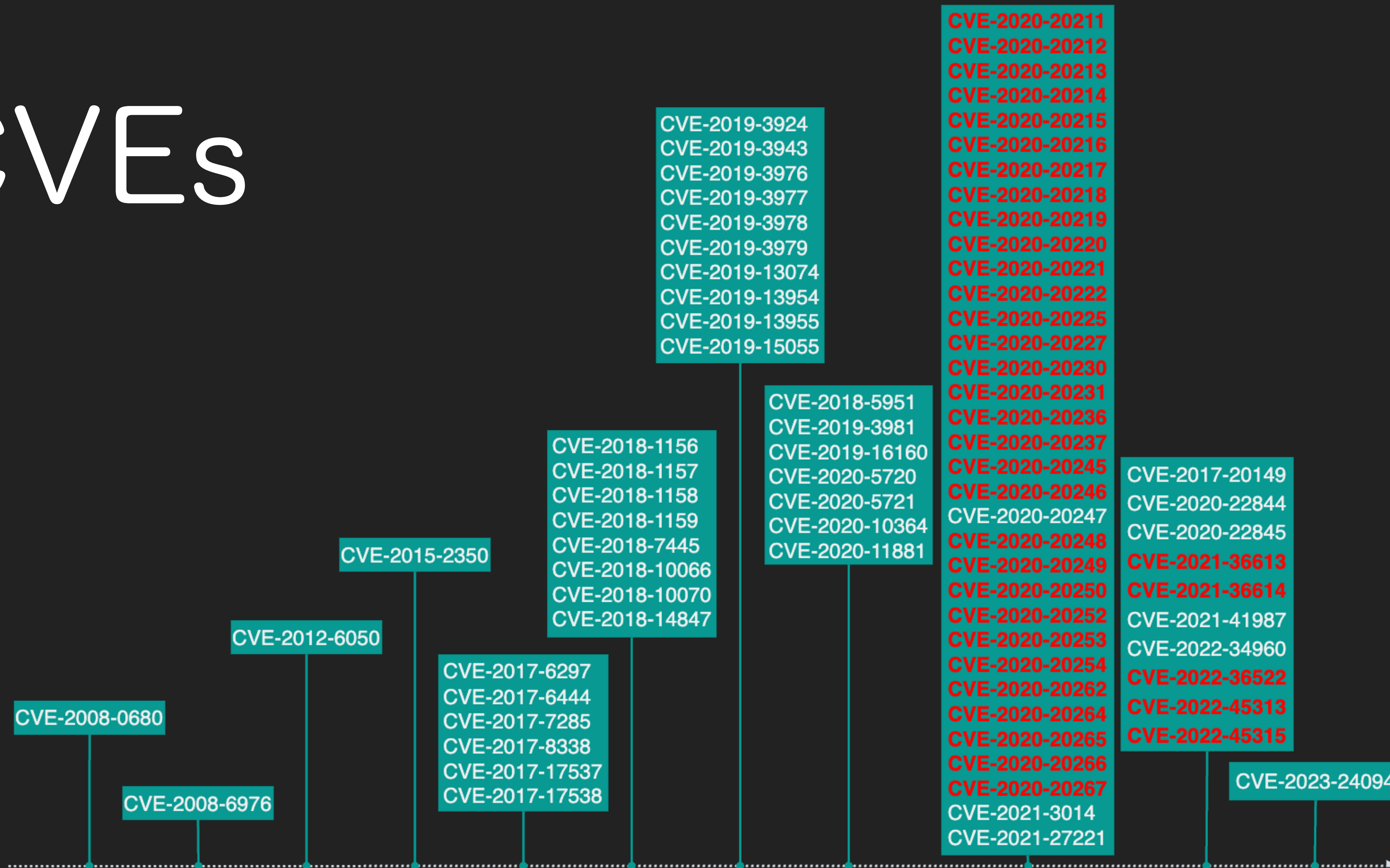
We are here

Qian Chen

MikroTik RouterOS Security:
The Forgotten IPC Message

fuzzing nova message in IPC

CVEs

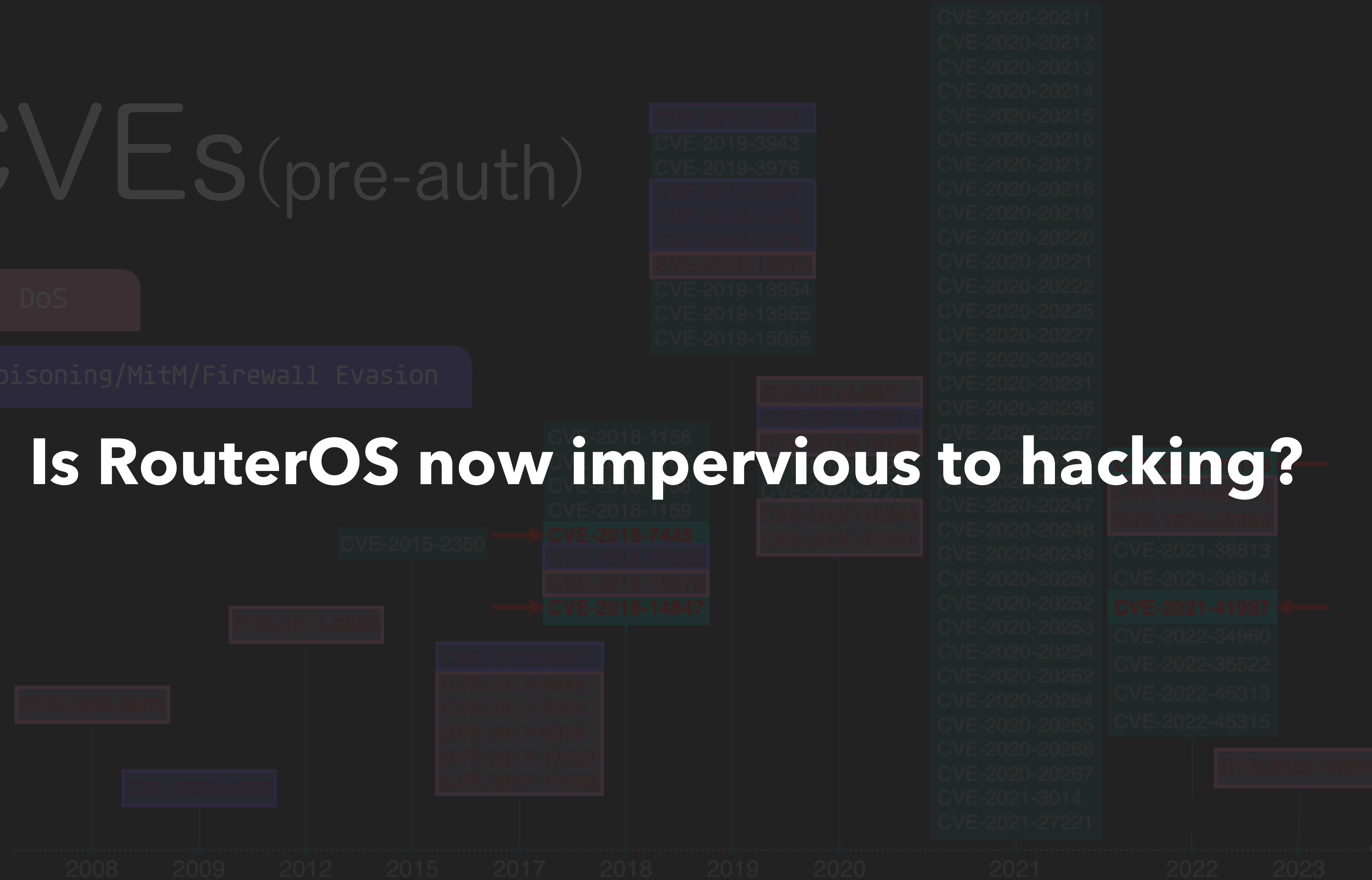


CVEs (pre-auth)

DoS

Poisoning/MitM/Firewall Evasion

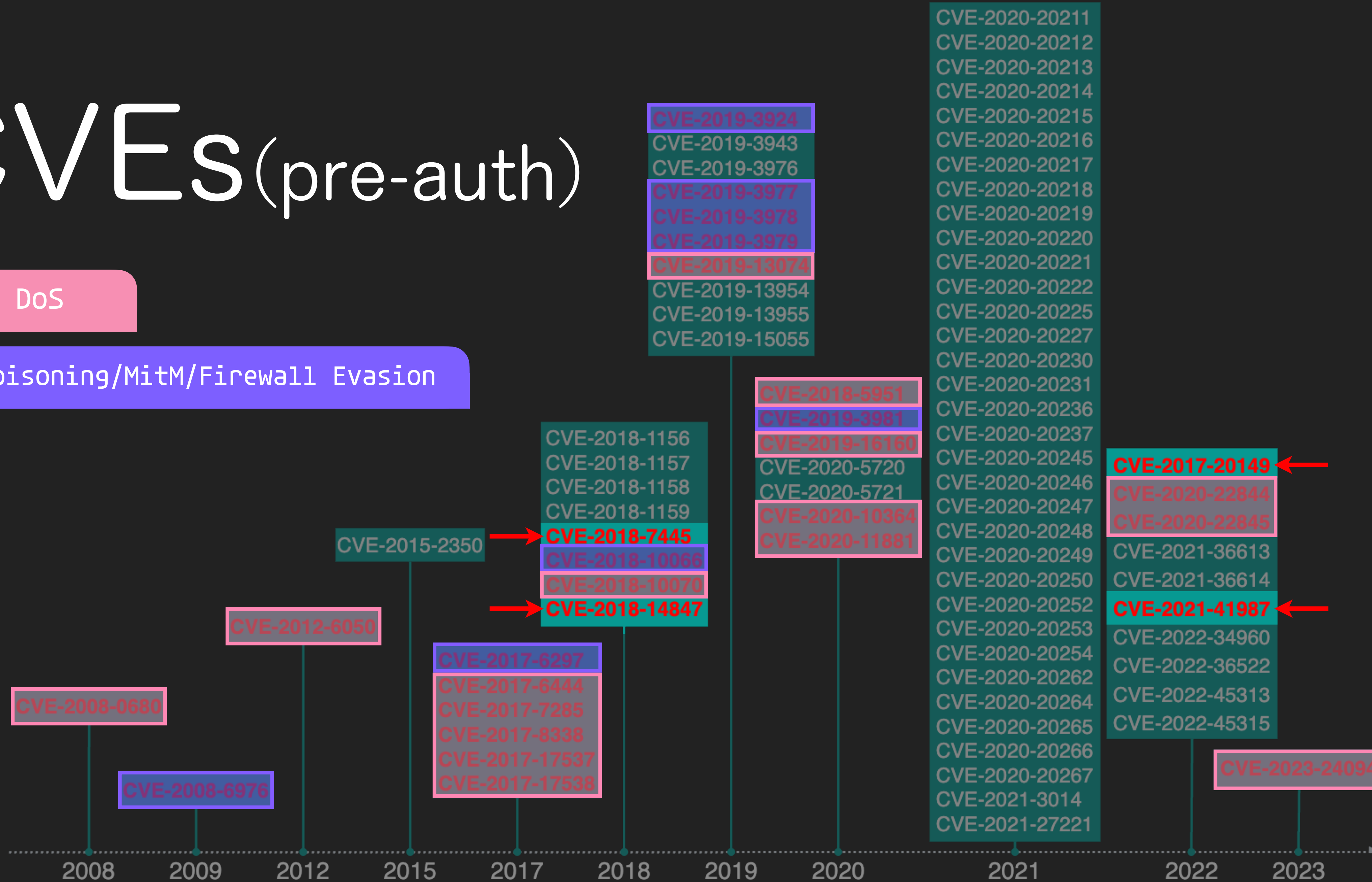
Is RouterOS now impervious to hacking?



CVEs (pre-auth)

DoS

Poisoning/MitM/Firewall Evasion



Observation

- Most researches are about
 - Jailbreaking
 - Analyzing the ITW exploits
 - Nova message in IPC

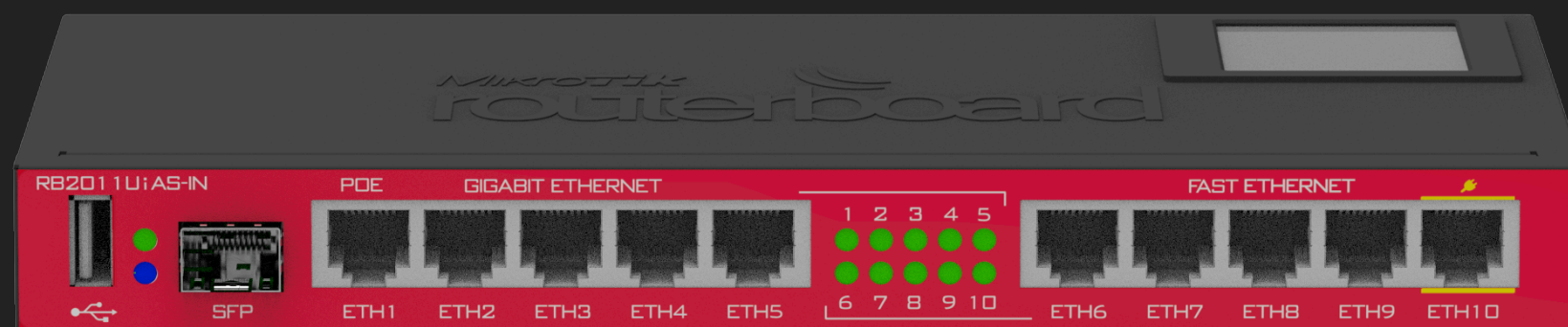
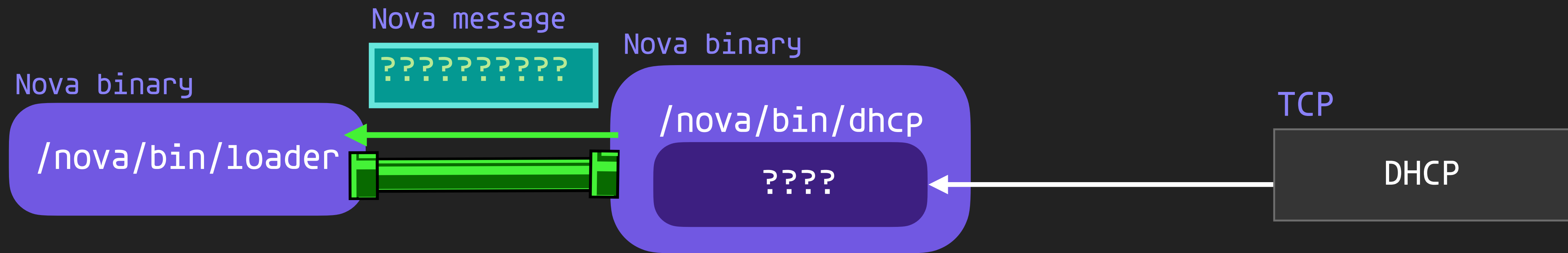
Brief Summary

No one with sanity
would like to dive into the details of Nova Binary

Where to start ?

Where to start ?

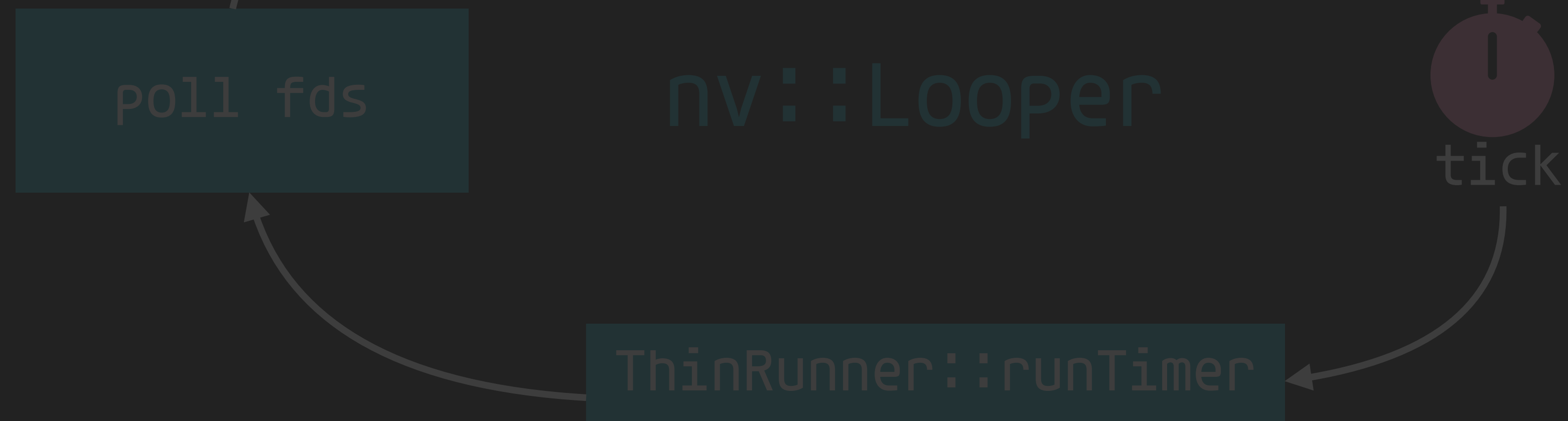
- Where are the entry points to the customize IPC ?



Nova Binary

- Every Nova binary has a Looper or MultifiberLooper.

Architecture of the Nova Binary

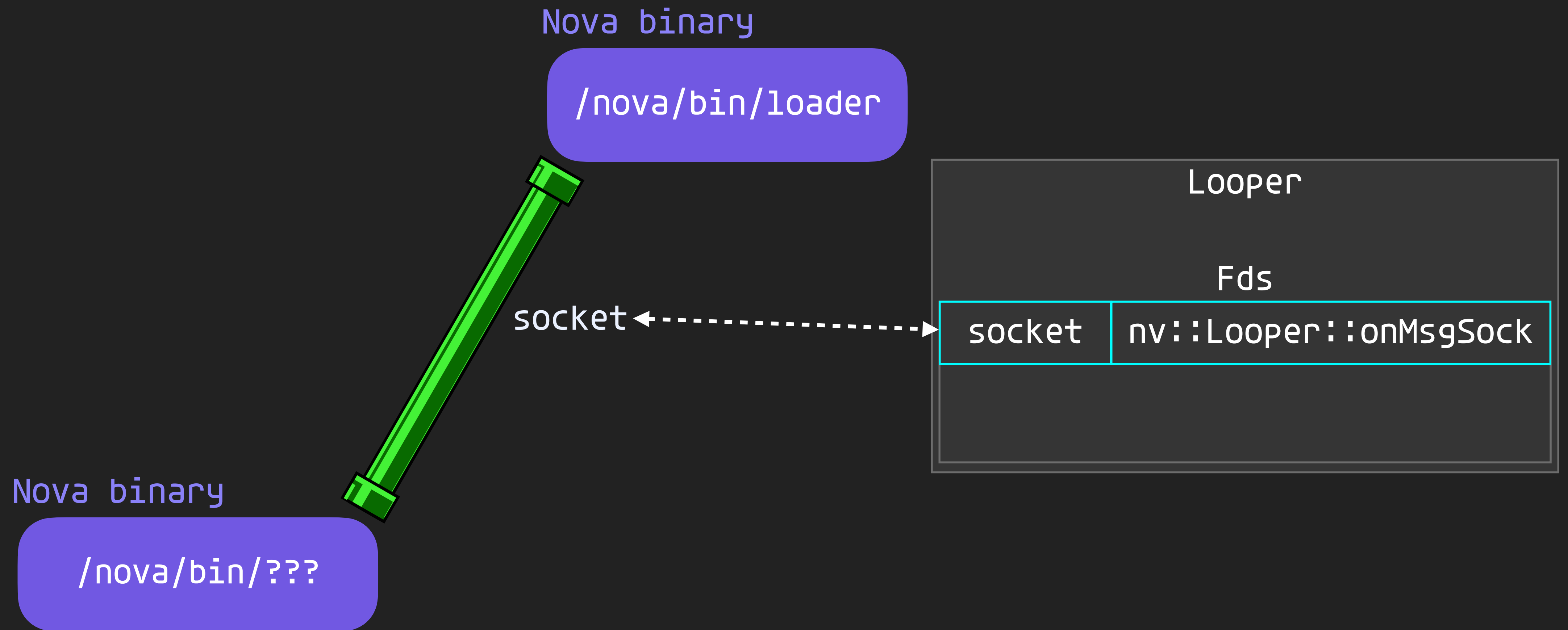


Nova Binary

- Every Nova binary has a Looper or a MultifiberLooper.



Nova Binary



Nova Binary

Nova binary

/nova/bin/loader

socket

Handler and its derived classes

/nova/bin/???

nv::Looper::onMsgSock

Nova binary

SYS_T0: [14]

Looper

SYS_T0: [14,0]

Handler 0

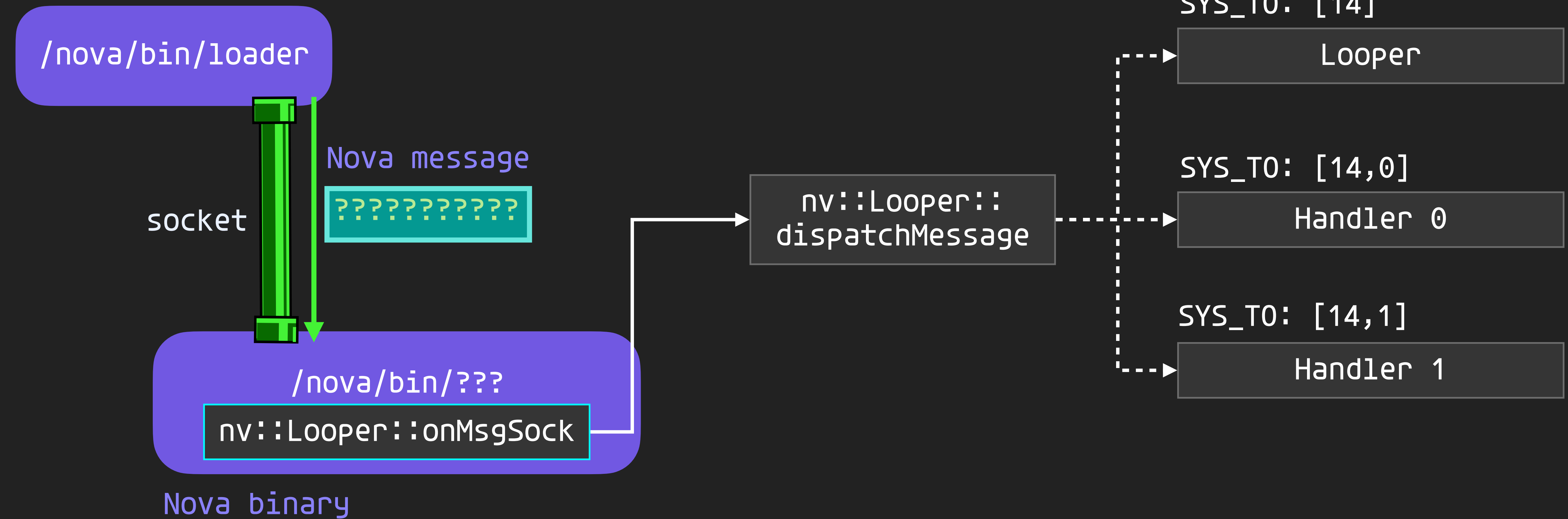
SYS_T0: [14,1]

Handler 1

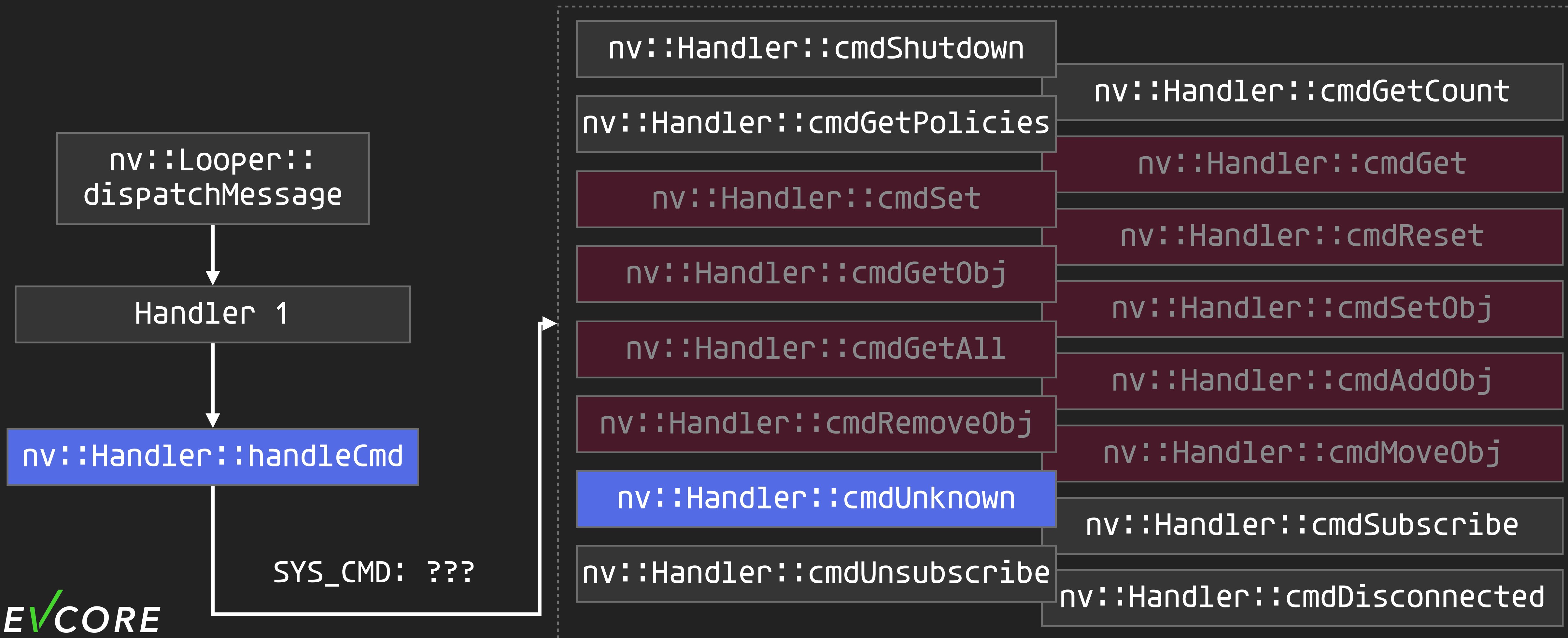
nv::Looper::dispatchMessage

Nova Binary

Nova binary

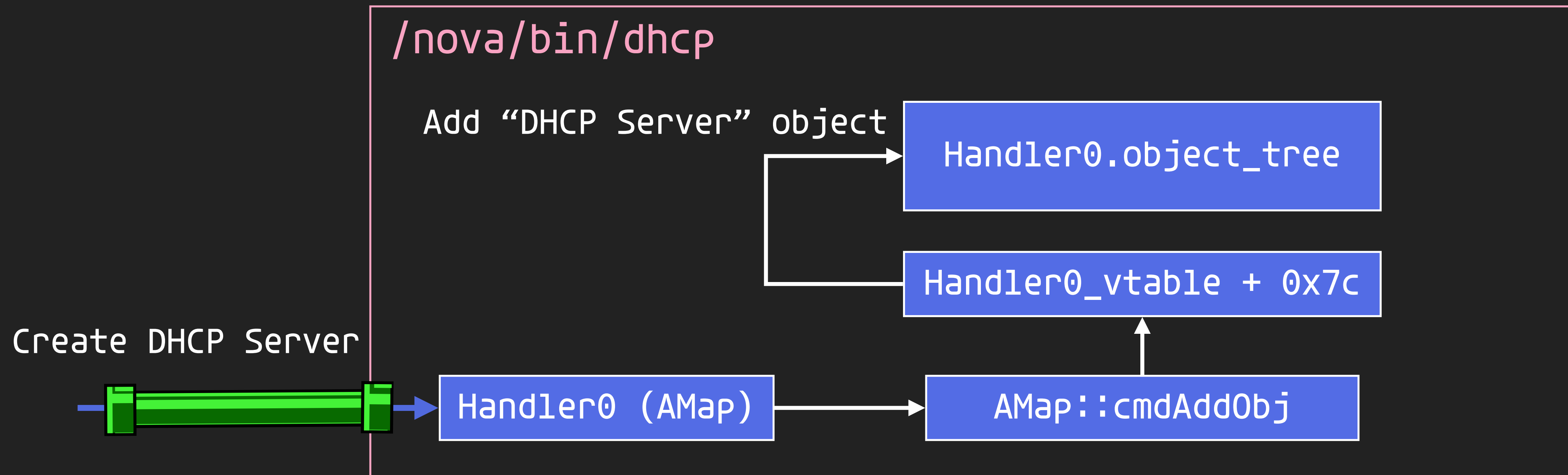


Nova Binary



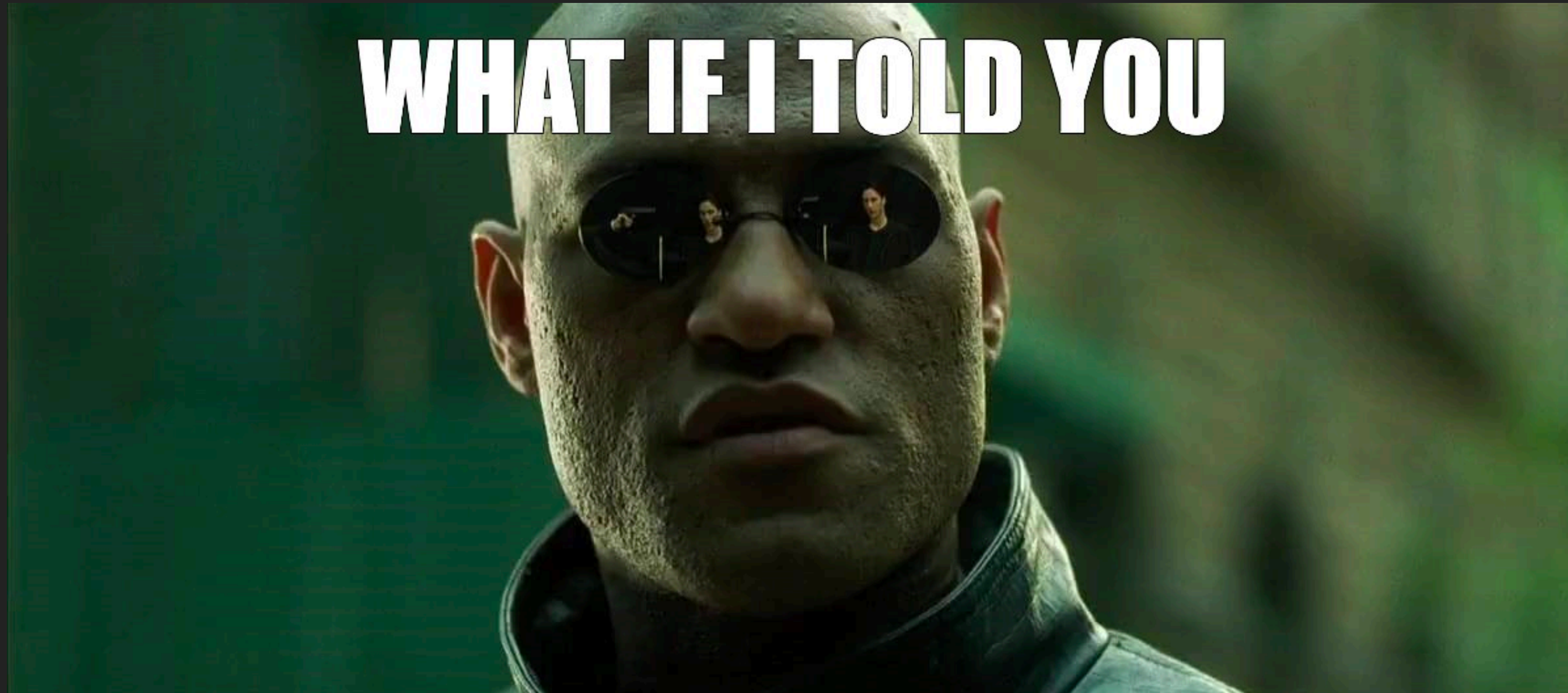
Nova Binary

- Base class: Handler
- Derived classes: AMap, AHolder, ASecMap, A0map, etc



Nova Message

- Some functionalities don't even use the Nova message.



/nova/bin/discover

Handler0::cmdUnknown

nv::createPacketReceiver

Example: CDP, LLDP

Register pairs of
socket and callback

Looper

Fds

CDP socket	callback
LLDP socket	callback

/nova/bin/discover

Handler0::cmdUnknown



nv::createPacketReceiver

Register pairs of
socket and callback



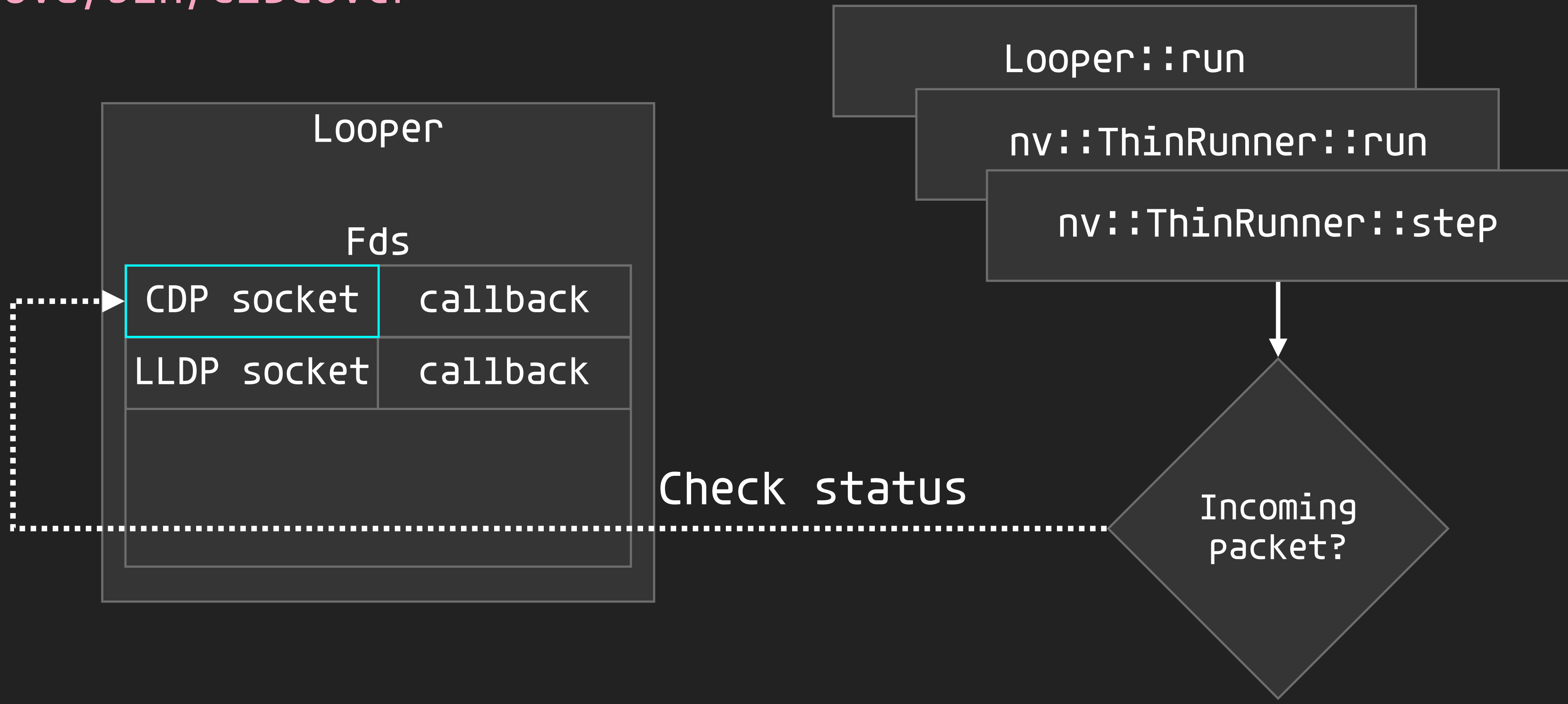
Looper

Fds

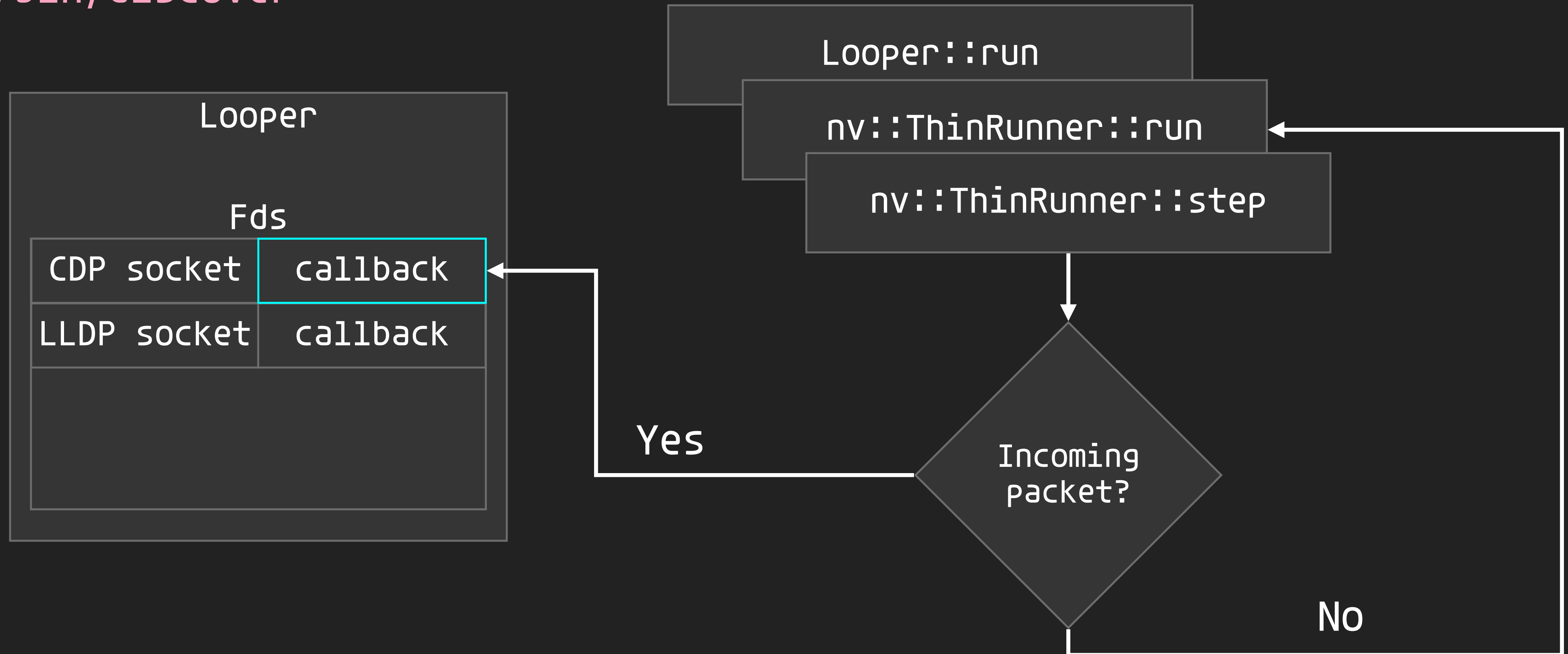
CDP socket	callback
LLDP socket	callback



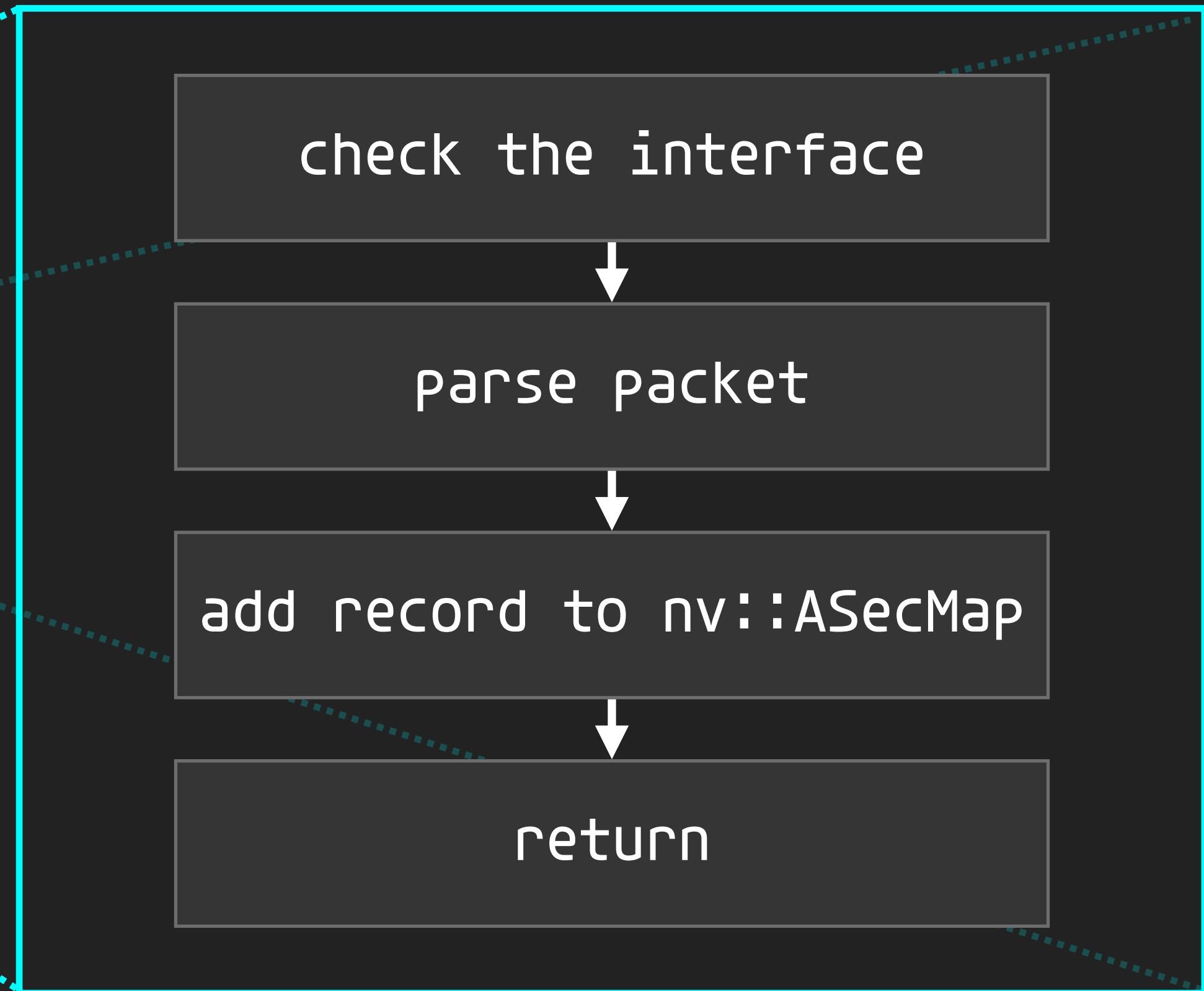
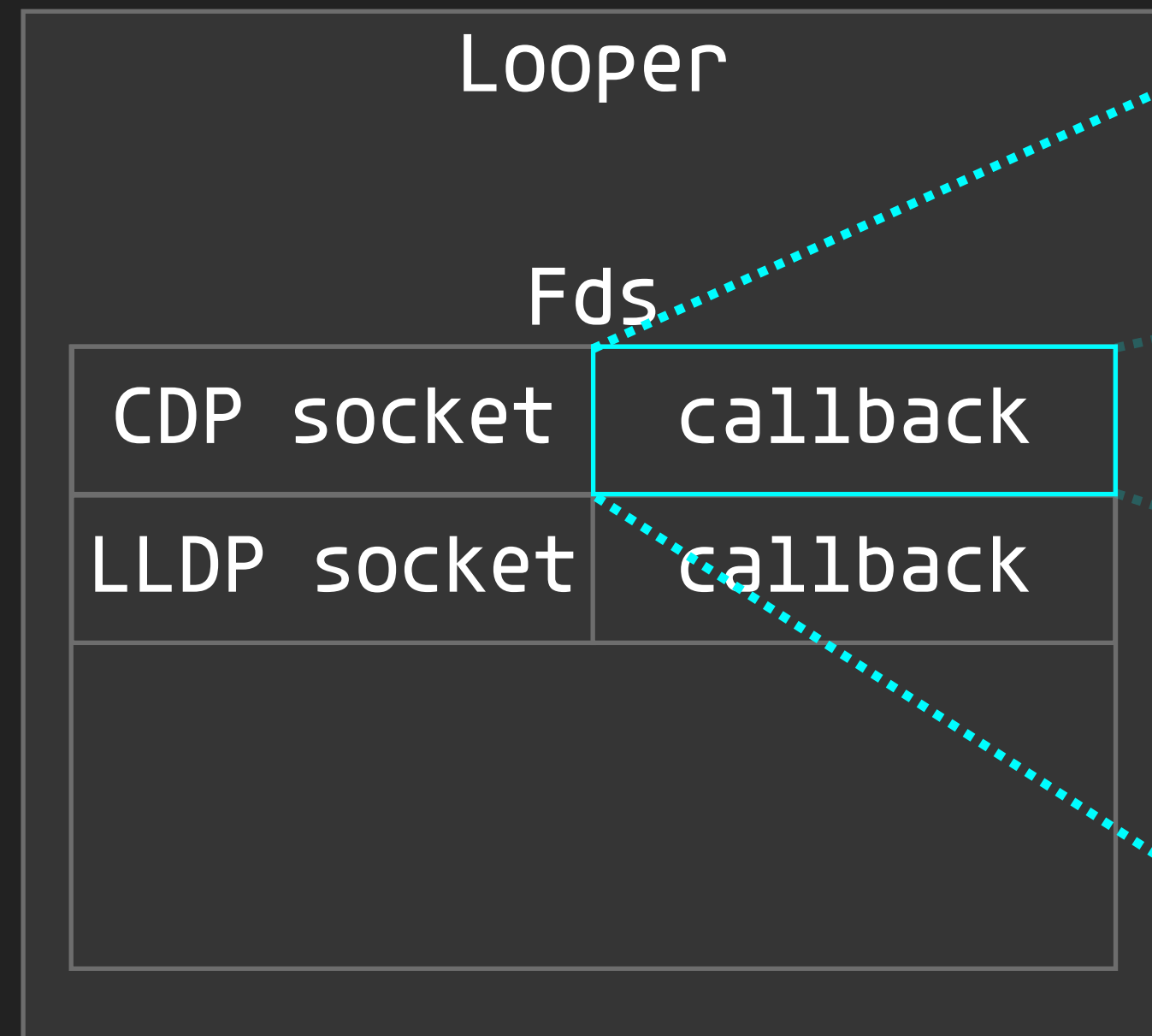
/nova/bin/discover



/nova/bin/discover



/nova/bin/discover



The pre-auth RCE

The pre-auth RCE

- Some random crashes of radvd occur while we plugging and unplugging cables on RouterBoard

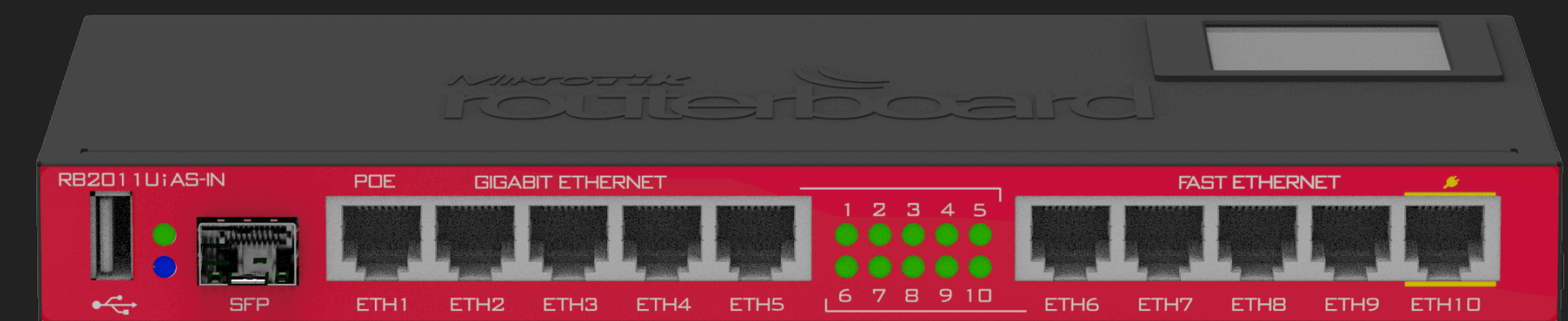


IPv6 SLAAC



ICMPv6

Router Solicitation (RS)

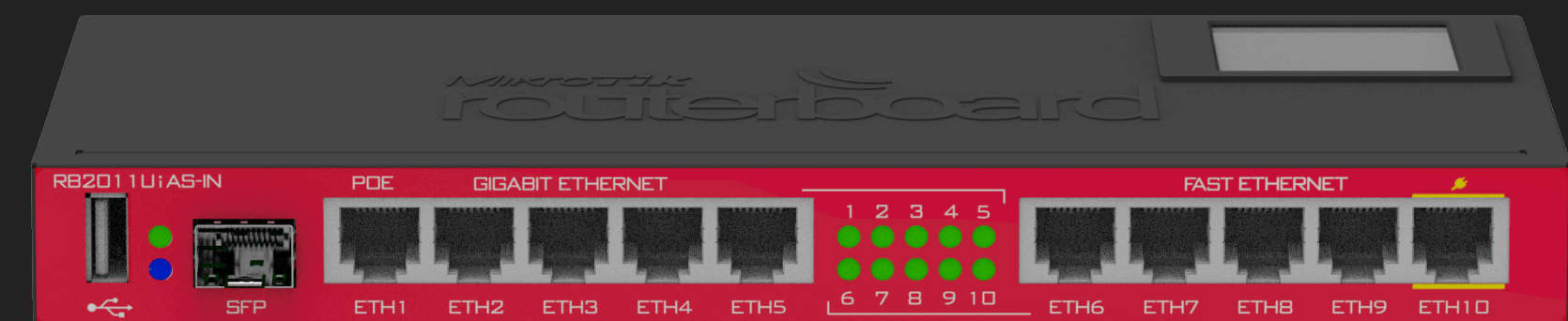


IPv6 SLAAC



ICMPv6

Router Solicitation (RS)



ICMPv6

Router Advertisement (RA)



prefix: 2001:db8::/64

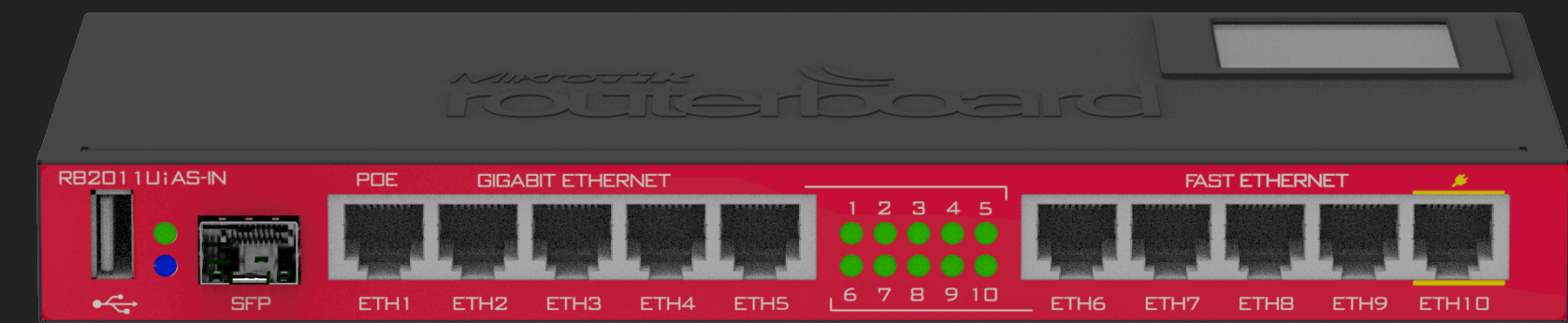
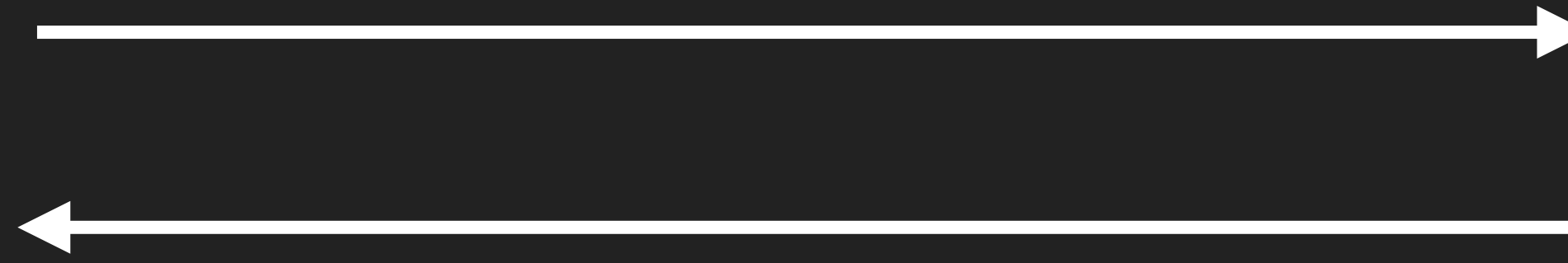
IPv6 SLAAC



IPv6 = prefix + EUI-64

ICMPv6

Router Solicitation (RS)



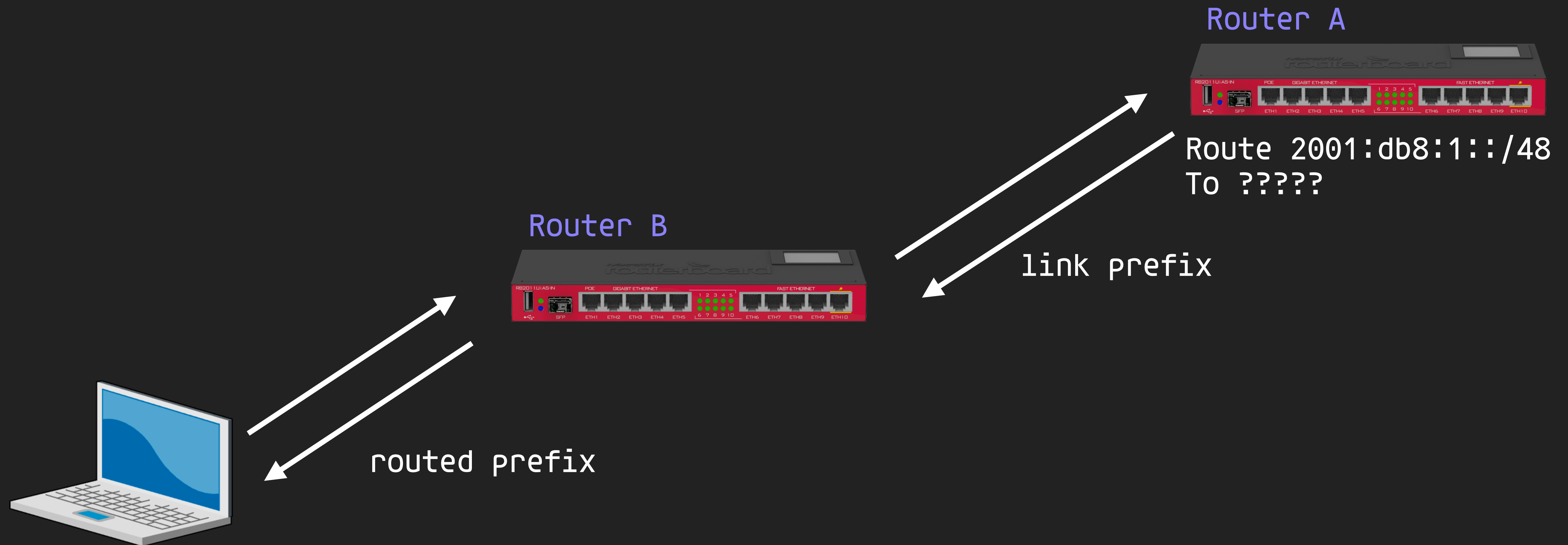
ICMPv6

Router Advertisement (RA)

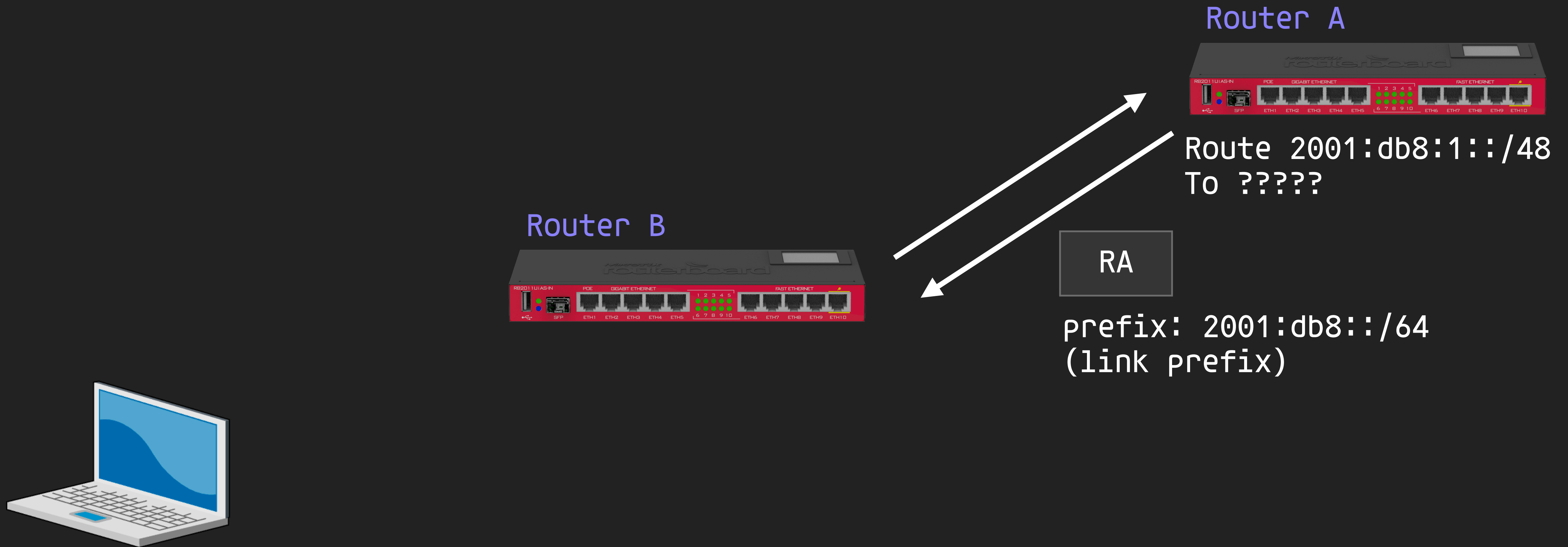


prefix: 2001:db8::/64

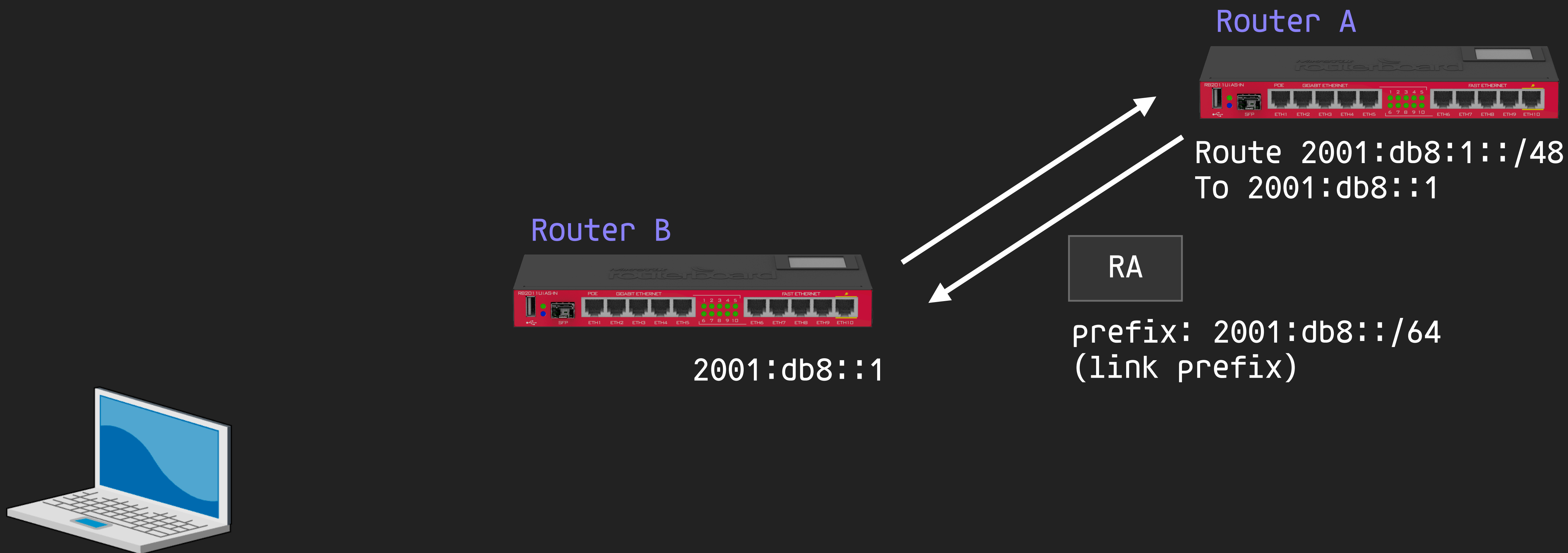
IPv6 SLAAC



IPv6 SLAAC

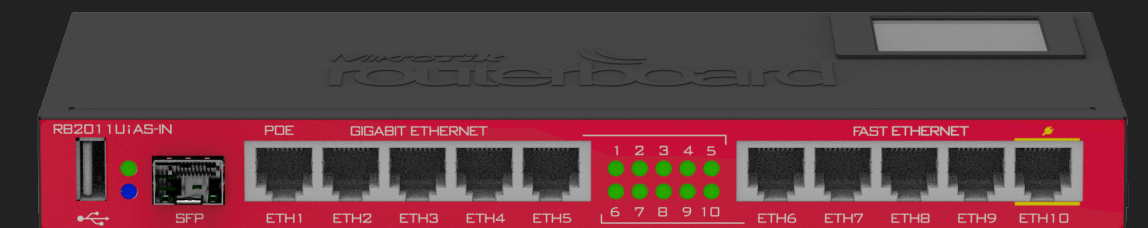


IPv6 SLAAC



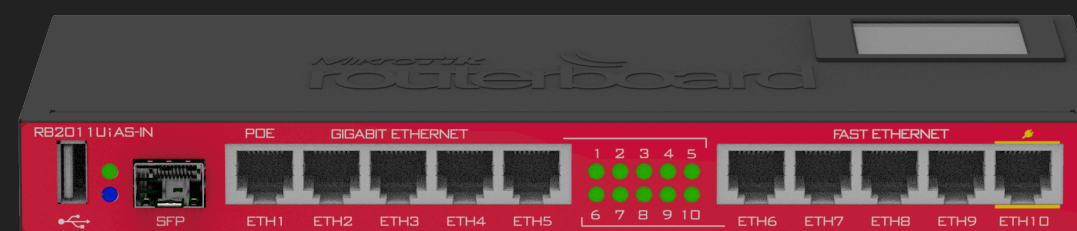
IPv6 SLAAC

Router A

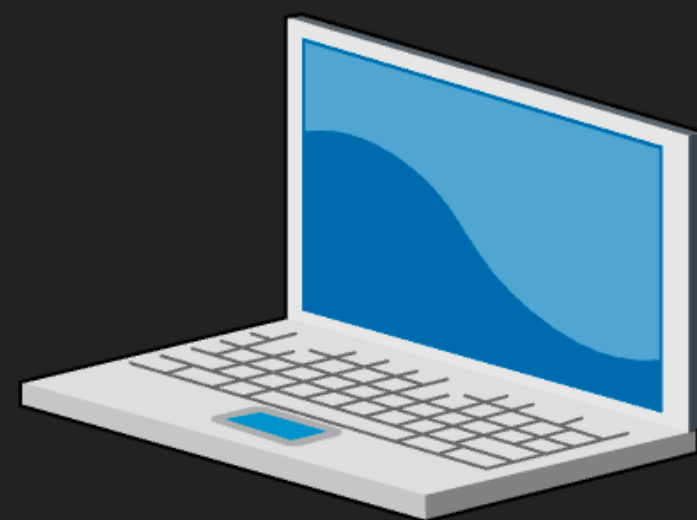


Route 2001:db8:1::/48
To 2001:db8::1

Router B



2001:db8::1

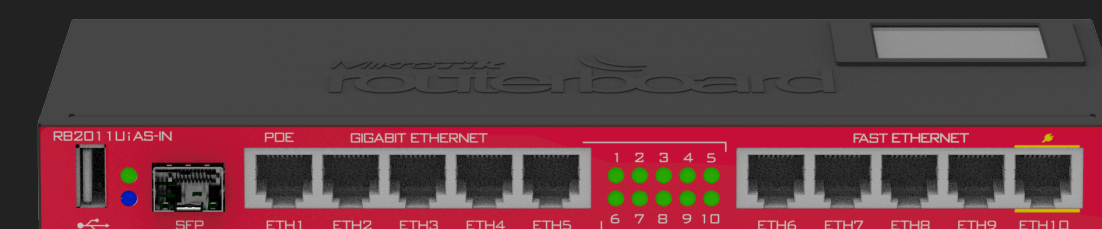


RA

prefix: 2001:db8:1::/48
(routed prefix)

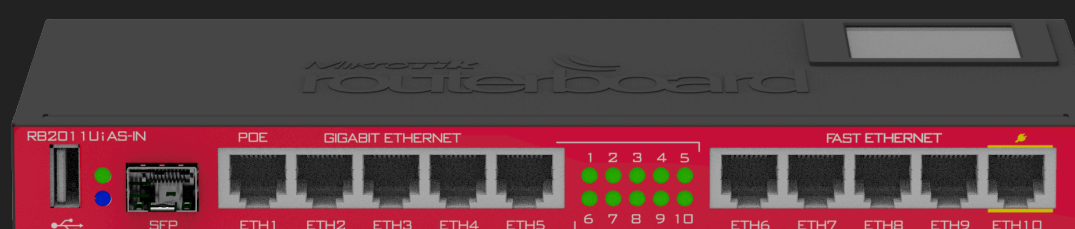
IPv6 SLAAC

Router A



Route 2001:db8:1::/48
To 2001:db8::1

Router B



2001:db8::1



2001:db8:1::1

RA

prefix: 2001:db8:1::/48
(routed prefix)

/bnd1/ipv6/nova/bin/radvd

main

Execution flow of radvd

nv::ThinRunner::addSocket

Register a pair of
Socket and callback

Looper

Fds

socket	onMsg..
socket	callback

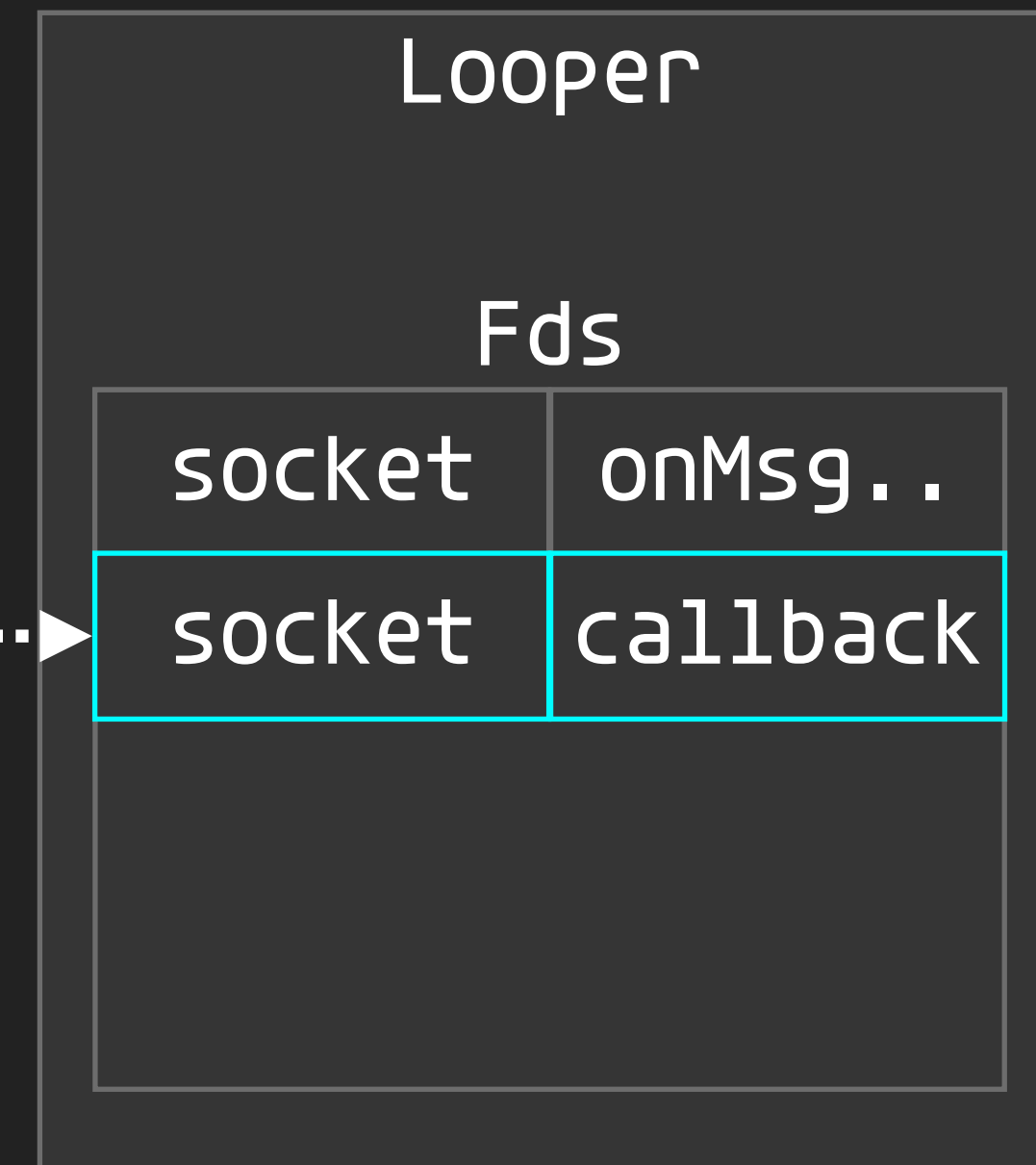
/bnd1/ipv6/nova/bin/radvd

main

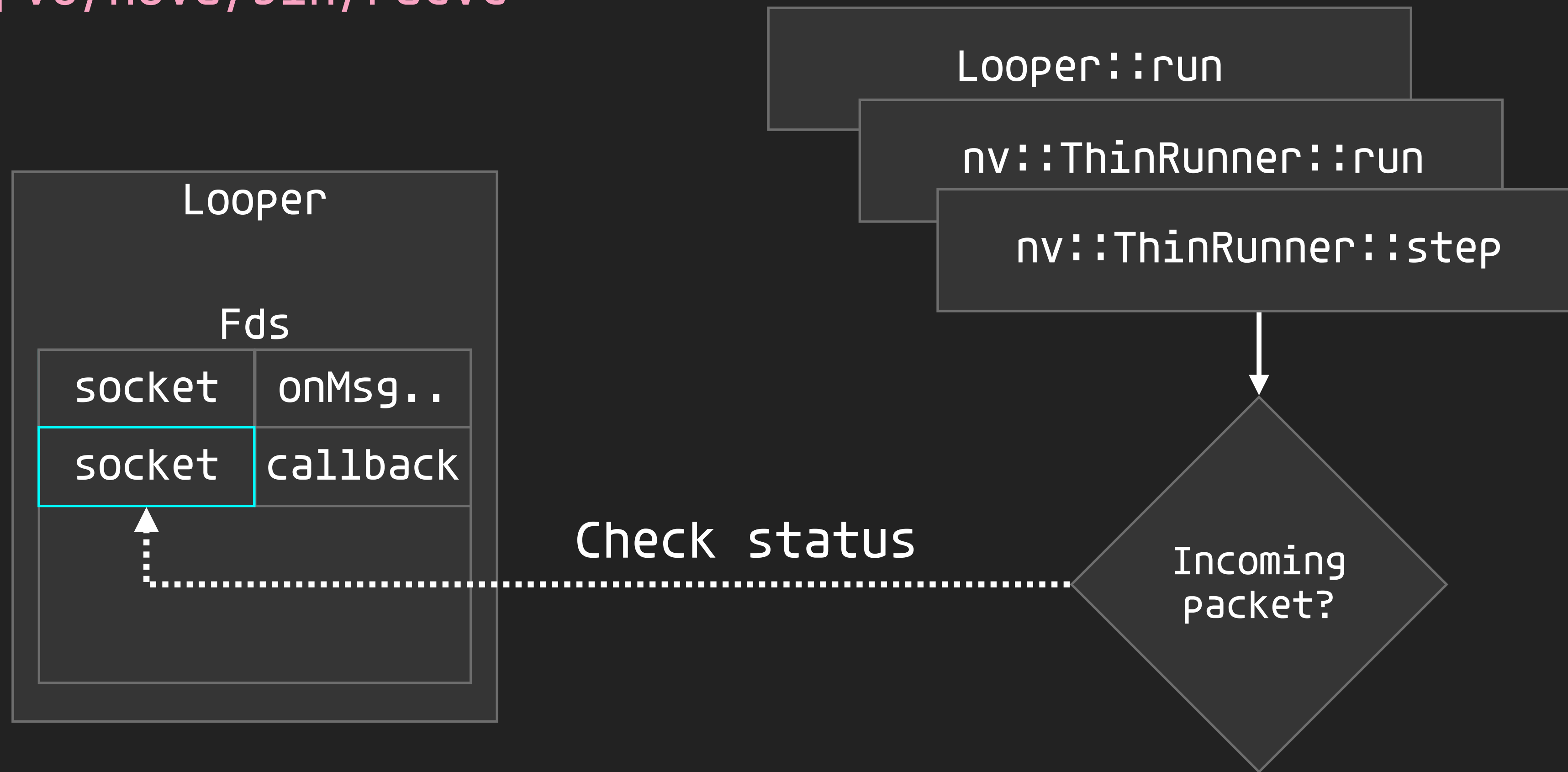


nv::ThinRunner::addSocket

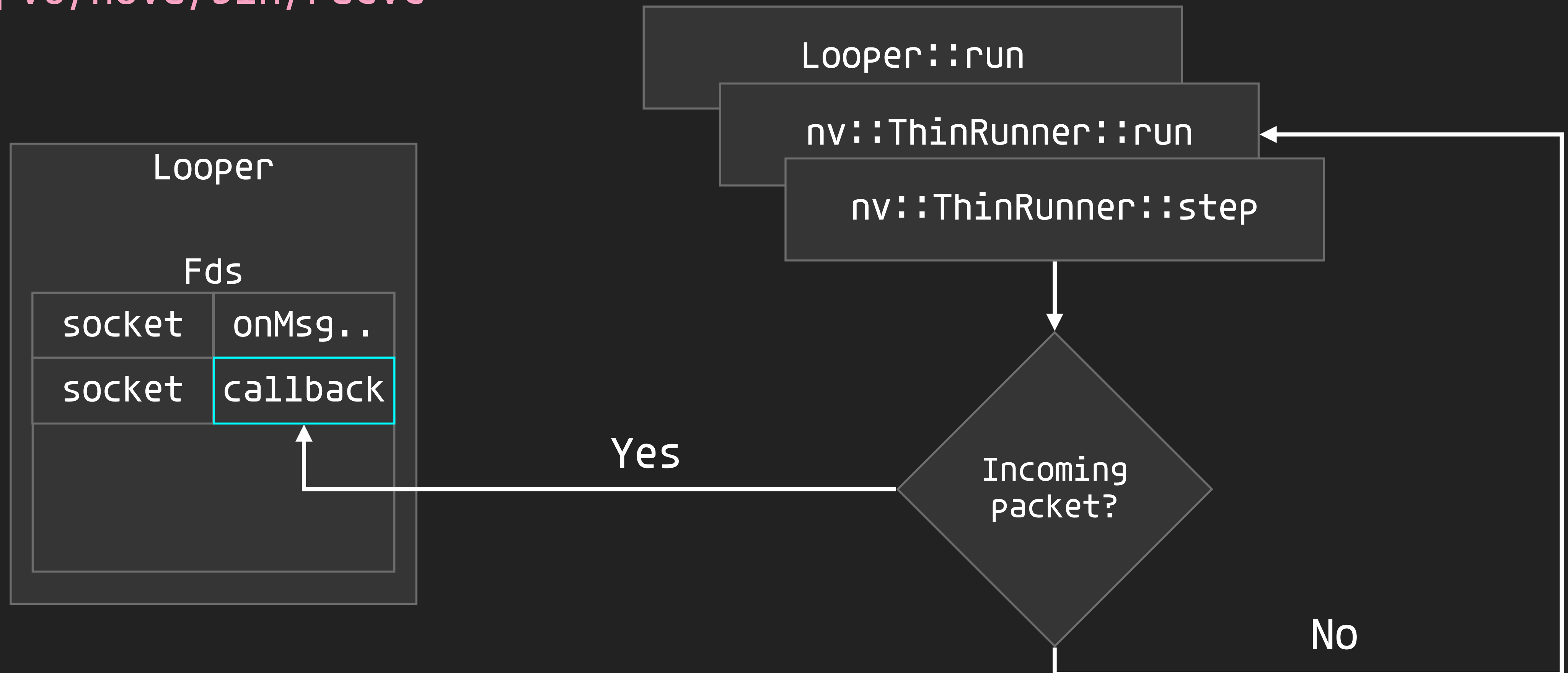
Register a pair of
Socket and callback



`/bnd1/ipv6/nova/bin/radvd`



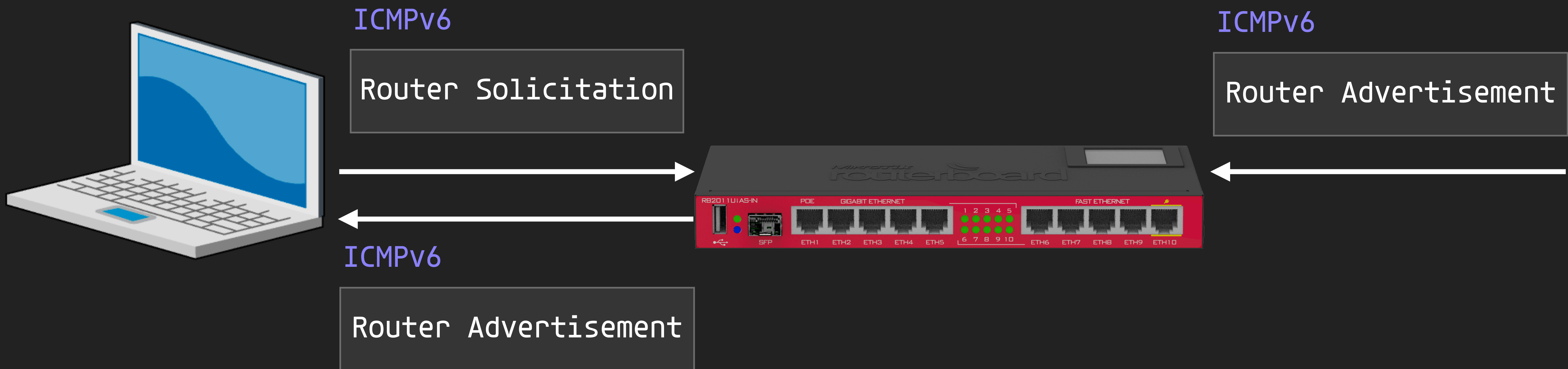
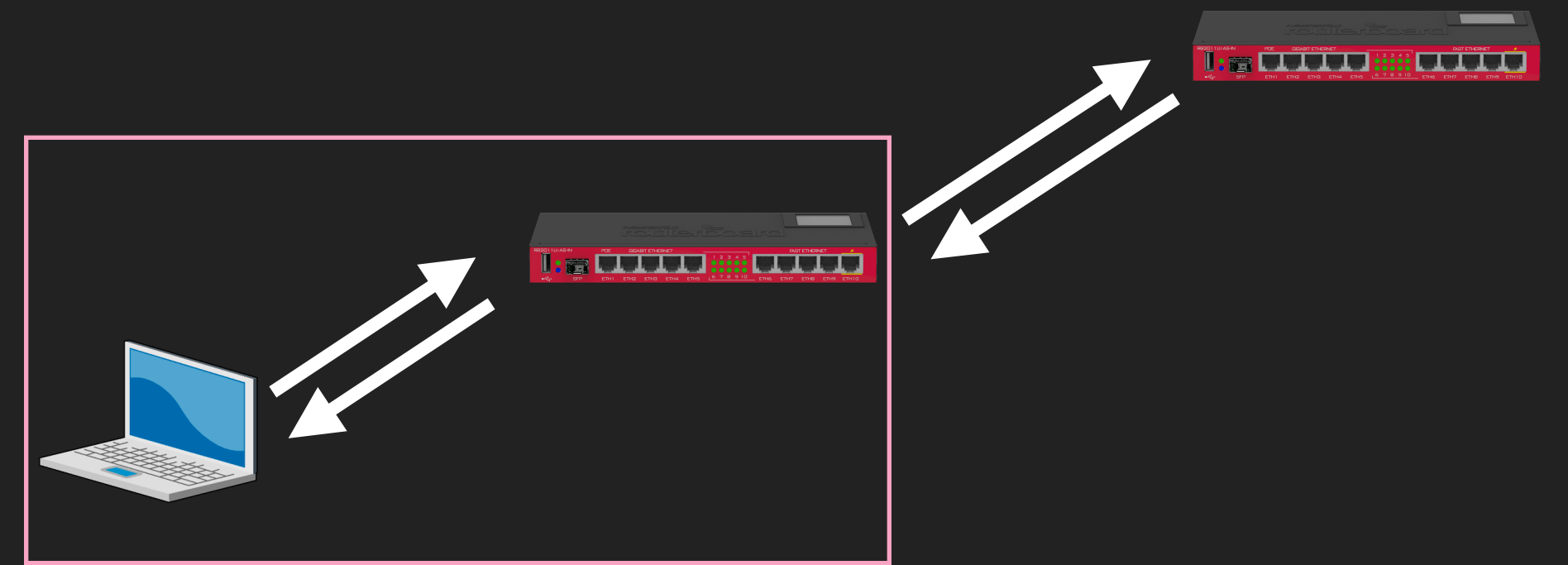
/bnd1/ipv6/nova/bin/radvd



RADVD

- In callback:
 - Check if the packet is a valid RA or a valid RS
 - If it is RA
 - Store information in handler 1 (AMap)
 - If it is RS
 - Multicast RA

IPv6 SLAAC



RADVD

- In callback:
 - Check if the packet is a valid RA or a valid RS
 - If it is an RA
 - Store information in handler 1 (AMap)
 - If it is an RS
 - Multicast RA



RADVD

- Where does the radvd construct RA?
 - radvd sends RA right after it receives an RS

```
Jiffies_now = nv::getJiffies();
jiffies_start = a1->jiffies_start;
if ( jiffies_start && Jiffies_now - jiffies_start >= (unsigned int)(100 * a1->ndsetting->RADelay_d3) )
{
    sendRA_407810(a1); ← What we really care
    addNextTimer_407308(a1);
    return 0;
}
```

RADVD

- Where does the radvd construct RA?
 - The handler 1 registers a timer to send RA periodically

```
if ( handler_1->ndsetting )
{
    notify(handler_1, 1);
    p_Runner64 = &nv::getLooper()->Runner64;
    callback.function_ptr = RAroutine; ←
    callback.arg = handler_1;
    if ( ((unsigned __int8)handler_1 & 1) != 0 )
        abort();
    nv::ThinRunner::addTimer(&timer_, p_Runner64, 0x32u, (int *)&callback);
    timer = timer_;
    clean_callback(&callback);
    handler_1->timer = timer;
}
```

```
1 void __fastcall RAroutine_407F00(interface *a1)
2 {
3     a1->timer = 0;
4     sendRA_407810(a1); ←
5     addNextTimer_407308(a1);
6 }
```

What we really care

RADVD

```
if ( v23->enable_advisory )
{
    lifetime = v23->lifetime;
    length = handler_1->DNS_tree.length;
    if ( length )
        length = addDNS((int)&RA_raw[pos], &handler_1->DNS_tree, (lifetime >> 1) + lifetime);
    expire_pos = length + pos;
    v32 = handler_1->expired_DNS_tree.length;
    if ( v32 )
        v32 = addDNS((int)&RA_raw[expire_pos], &handler_1->expired_DNS_tree, 0);
    pos = v32 + expire_pos;
    tree_begin = a1->prefix_tree.tree_begin;
}
else
```

Stack buffer with size 0x1000

```
int __fastcall addDNS(int a1, tree_base *a2, int lifetime)
{
    // [COLLAPSED LOCAL DECLARATIONS. PRESS KEYPAD CTRL-"+" TO EXPAND]
    BYTE *)a1 = 25;
    a2->length;
    + 2) = 0;
    *(_D
    v14.iter
    v8 = 8;
    for ( i = (tree_base *)tree_begin; i != (tree_base *)&a2->tree_end; i = (tree_base *)v14.iter )
    {
        if ( sub_406610() )
        {
            operator<<((int)&logger, (int)"adding DNS server option, address=");
            v10 = operator<<();
            v11 = " (expired)";
            if ( lifetime )
                v11 = "";
            v12 = (ostream *)operator<<(v10, (int)v11);
            endl(v12);
        }
        memcpy((void *) (a1 + v8), &v14.iter->data, 0x10u);
        v8 += 16;
        tree_iterator_base::incr(&v14);
    }
    return v8;
}
```

Stack buffer with size 0x1000

```
int __fastcall addDNS(int a1, tree_base *a2, int lifetime)
{
    // [COLLAPSED LOCAL DECLARATIONS. PRESS KEYPAD CTRL-"+" TO EXPAND]
    BYTE *)a1 = 25;
    a2->length;
    + 2) = 0;
    *(_D
    v14.iter
    v8 = 8;
    for ( i = (tree_base *)tree_begin; i != (tree_base *)&a2->tree_end; i = (tree_base *)v14.iter )
    {
        if ( sub_406610() )
        {
            operator<<((int)&logger, (int)"adding DNS server option, address=");
            v10 = operator<<();
            v11 = " (expired)";
            if ( lifetime )
                v11 = "";
            v12 = (ostream *)operator<<(v10, (int)v11);
            endl(v12);
        }
        memcpy((void *) (a1 + v8), &v14.iter->data, 0x10u);
        v8 += 16;
        tree_iterator_base::incr(&v14);
    }
    return v8;
}
```

Stack buffer with size 0x1000

No boundary check,
overflow if the tree is big enough

5.1. Recursive DNS Server Option

The RDNSS option contains one or more IPv6 addresses of RDNSSes. All of the addresses share the same Lifetime value. If it is desirable to have different Lifetime values, multiple RDNSS options can be used. Figure 1 shows the format of the RDNSS option.

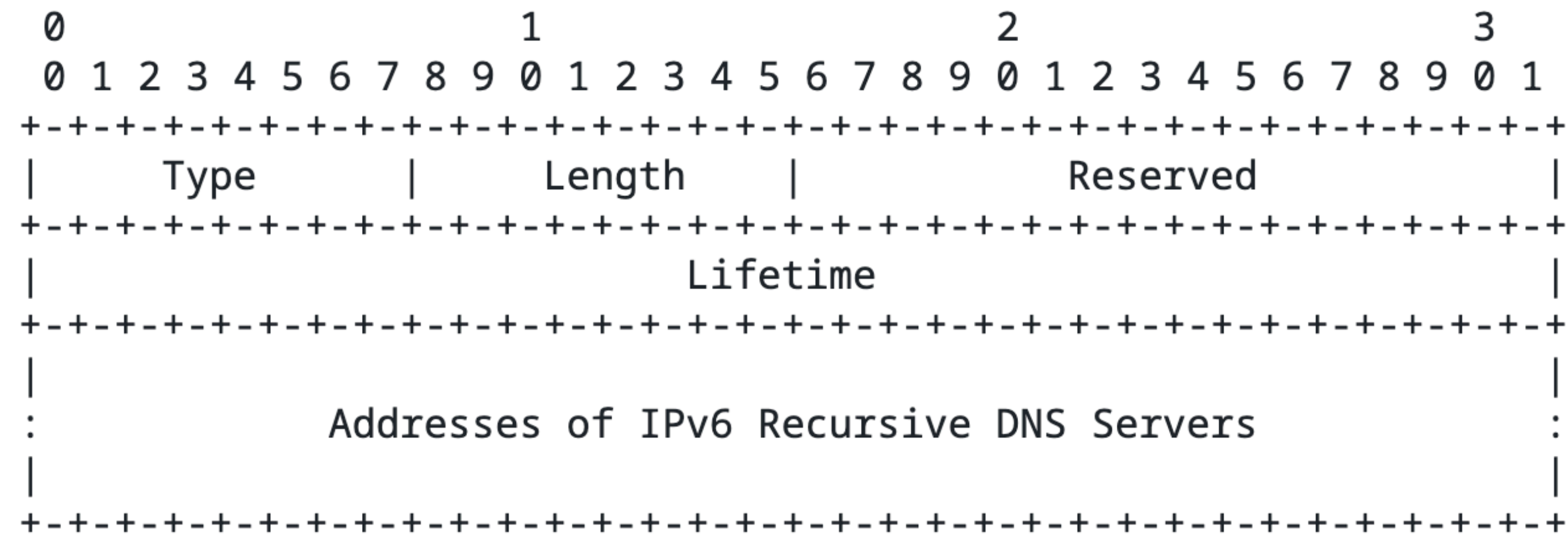


Figure 1: RDNSS Option Format

Fields:

Type 8-bit identifier of the RDNSS option type as assigned by IANA: 25

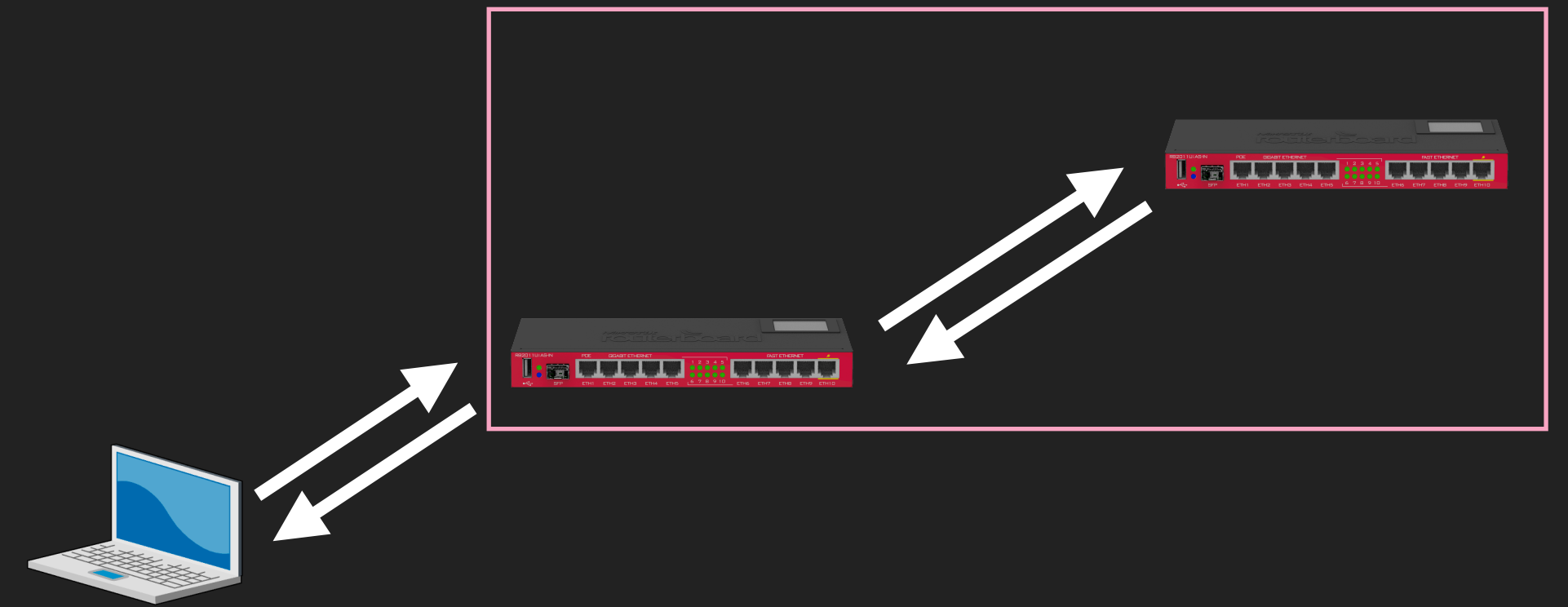
Length 8-bit unsigned integer. The length of the option (including the Type and Length fields) is in units of 8 octets. The minimum value is 3 if one IPv6 address is contained in the option. **Every additional RDNSS address increases the length by 2.** The Length field is used by the receiver to determine the number of IPv6 addresses in the option.

RADVD

```
if ( v23->enable_advisory )
{
    lifetime = v23->lifetime;
    length = handler_1->DNS_tree.length;
    if ( length )
        length = addDNS((int)&RA_raw[pos], &handler_1->DNS_tree, (lifetime >> 1) + lifetime);
    expire_pos = length + pos;
    v32 = handler_1->expired_DNS_tree.length;
    if ( v32 )
        v32 = addDNS((int)&RA_raw[expire_pos], &handler_1->expired_DNS_tree, 0);
    pos = v32 + expire_pos;
    tree_begin = a1->prefix_tree.tree_begin;
}
else
```

Stack buffer with size 0x1000

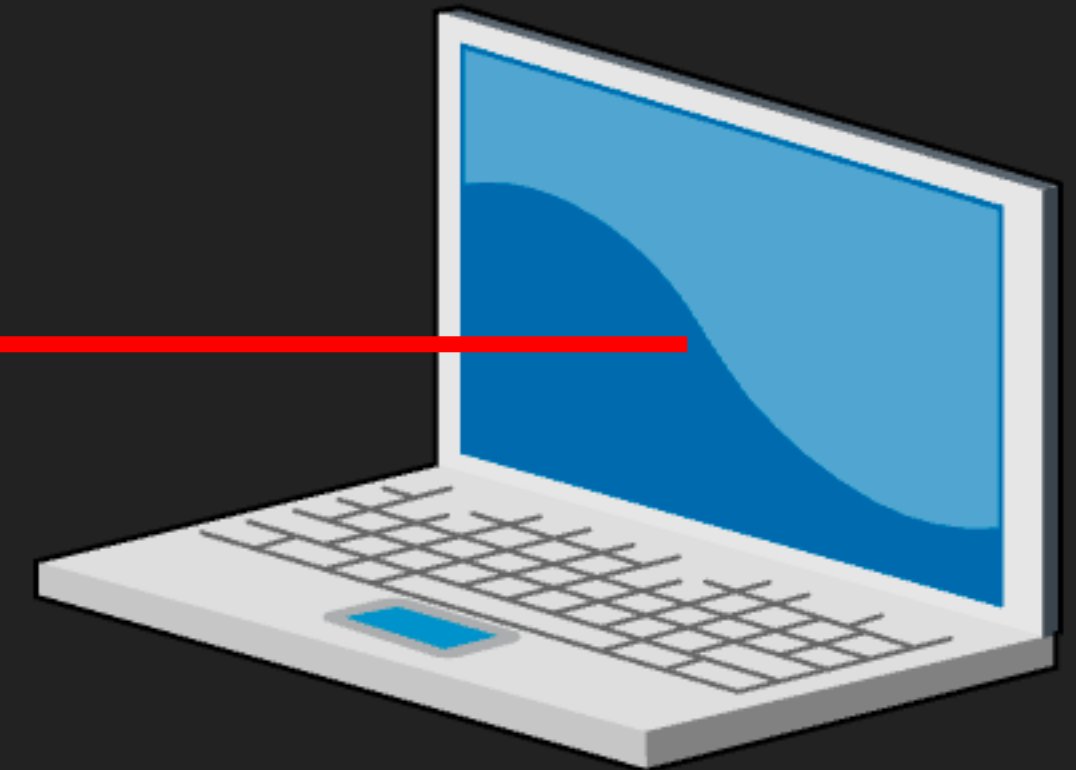
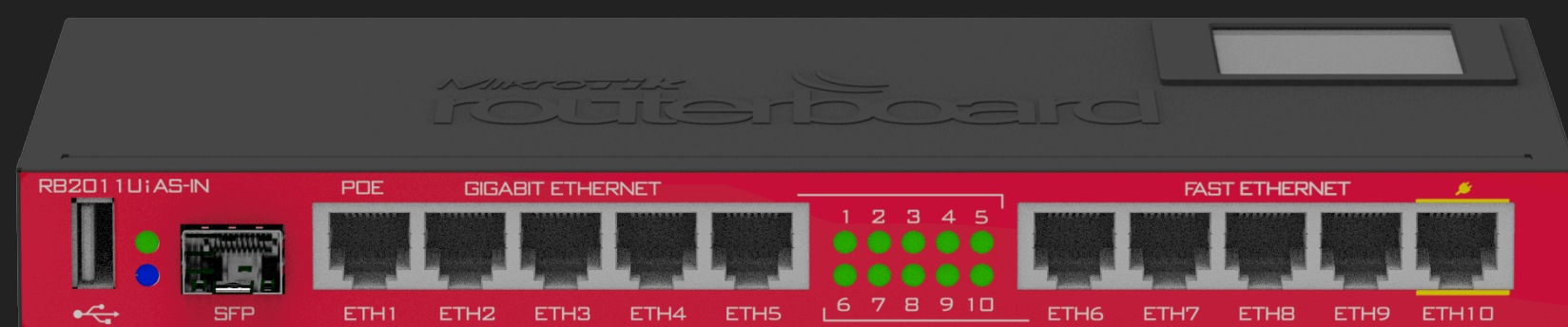
IPv6 SLAAC



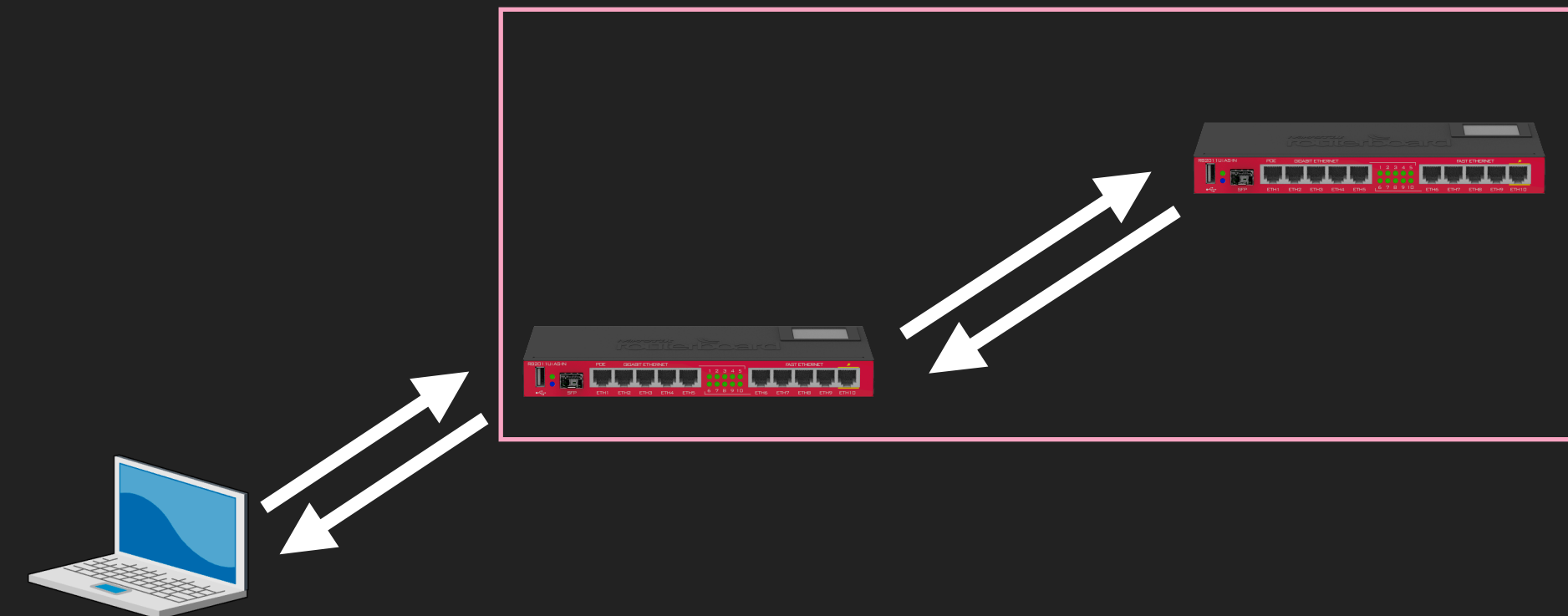
ICMPv6

Router Advertisement (RA)

RDNSS: <a big list>



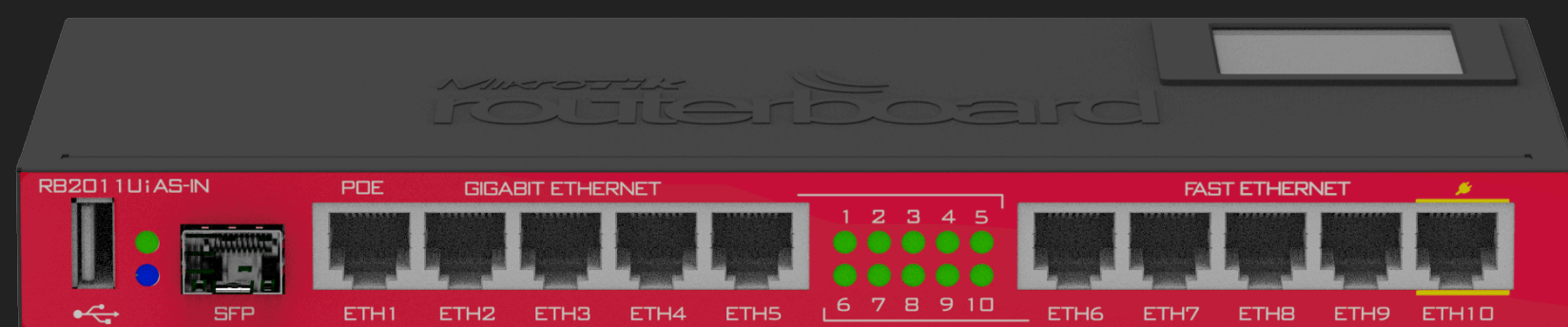
IPv6 SLAAC



ICMPv6

Router Advertisement (RA)

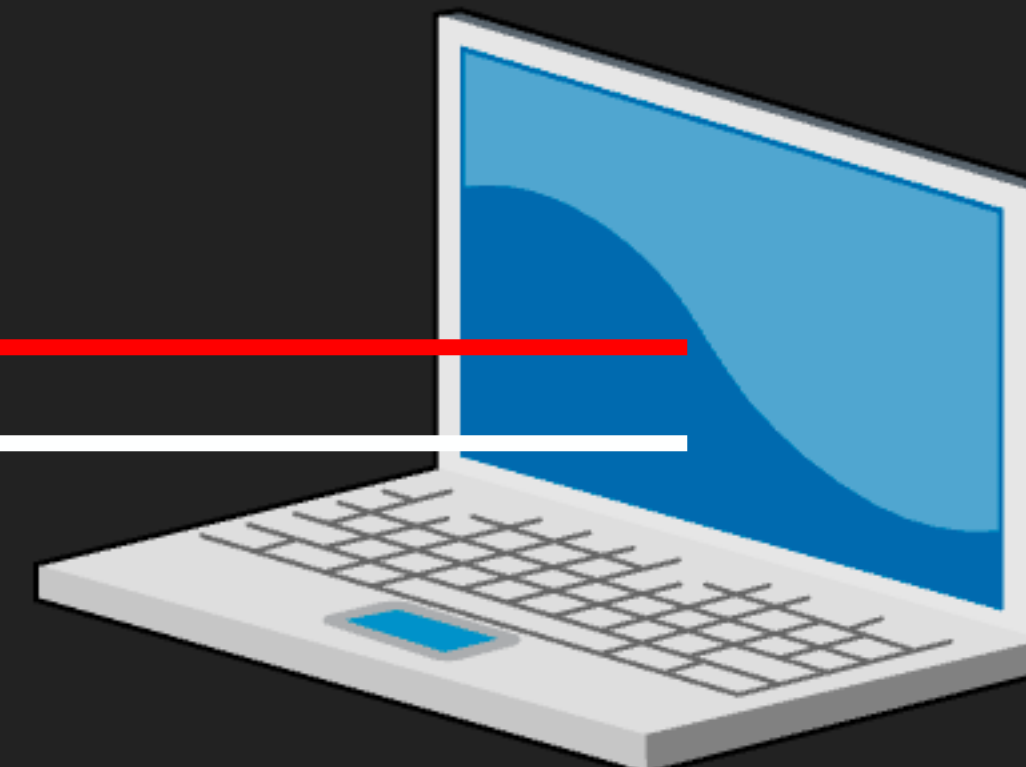
RDNSS: <a big list>



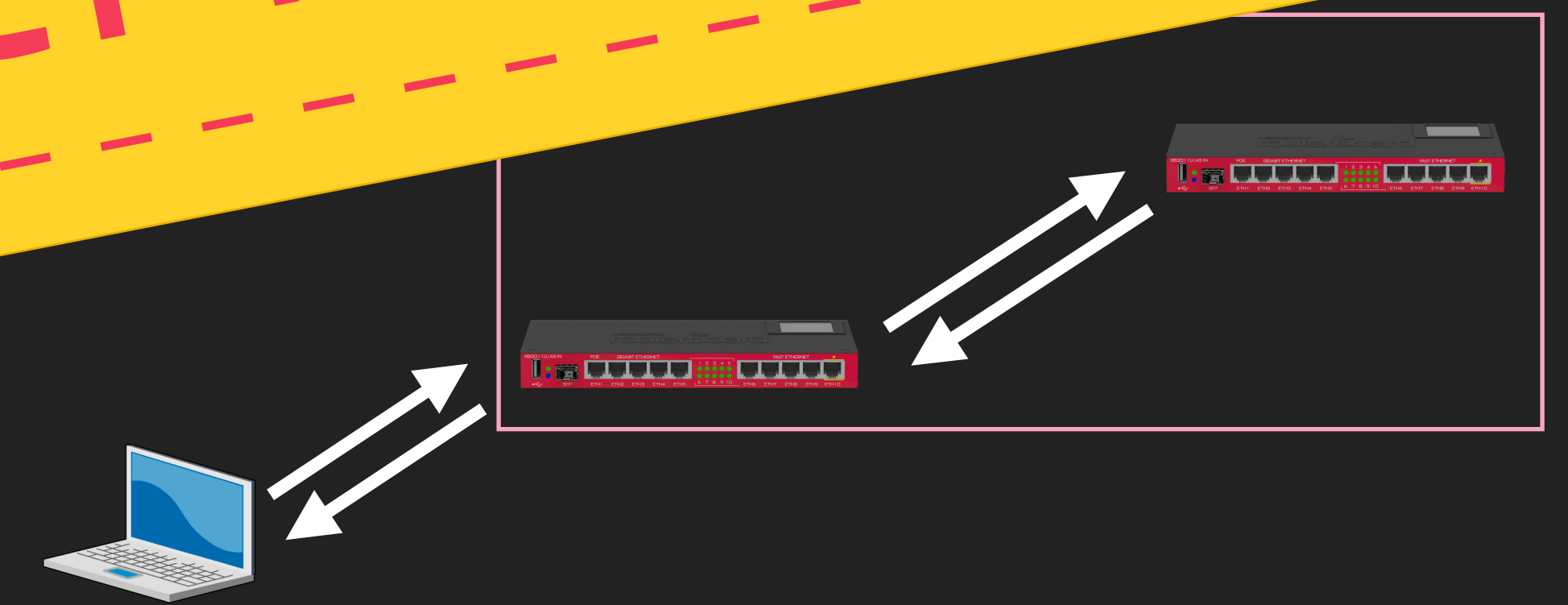
ICMPv6

Router Advertisement (RA)

RDNSS: <a big list>



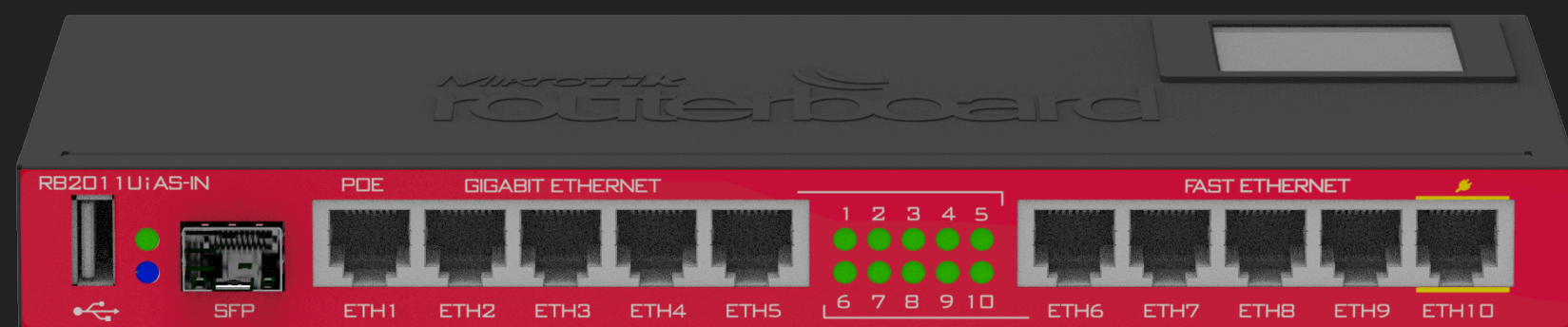
overflow! overflow! overflow!



ICMPv6

Router Advertisement (RA)

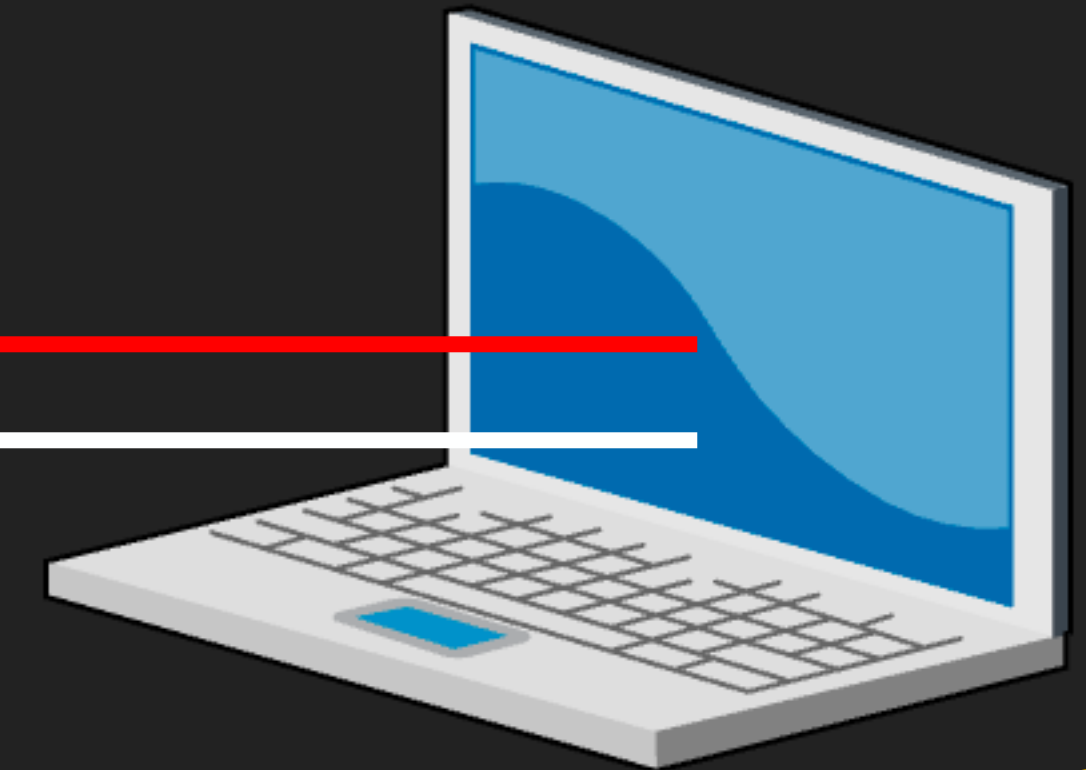
RDNSS: <a big list>



ICMPv6

Router Advertisement (RA)

RDNSS: <a big list>



overflow! overflow! overflow!

```
[*] '/tmp/radvd'  
Arch:      mips-32-big  
RELRO:     No RELRO  
Stack:     No canary found  
NX:        NX disabled  
PIE:       No PIE (0x400000)  
RWX:       Has RWX segments  
ELF('/tmp/radvd')
```



**Just overflow the
shellcode on the
stack and jump to it**

***sure grandma let's
get you to bed***

Credit: @__ammar2__

R0P gadgets

```
0:1:xxxx:xxxx:xxxx:xxxx:xxxx:xxxx  
0:2:xxxx:xxxx:xxxx:xxxx:xxxx:xxxx  
0:3:xxxx:xxxx:xxxx:xxxx:xxxx:xxxx
```

...

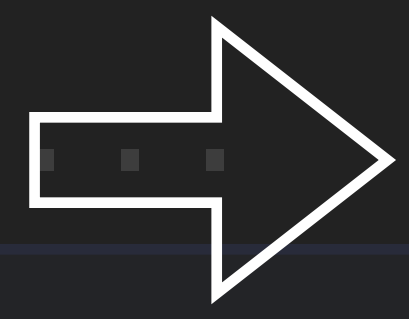
```
0:20:xxxx:xxxx:xxxx:xxxx:jump $+8  
0:21:xxxx:xxxx:xxxx:xxxx:jump $+8  
0:22:xxxx:xxxx:xxxx:xxxx:jump $+8
```

shellcode

R0P gadgets

0:1:xxxx:xxxx:xxxx:xxxx:xxxx:xxxx
0:2:xxxx:xxxx:xxxx:xxxx:xxxx:xxxx
0:3:xxxx:xxxx:xxxx:xxxx:xxxx:xxxx
0:4:xxxx:xxxx:xxxx:xxxx:xxxx:xxxx
0:5:xxxx:xxxx:xxxx:xxxx:xxxx:xxxx
0:6:xxxx:xxxx:xxxx:xxxx:xxxx:xxxx
0:7:xxxx:xxxx:xxxx:xxxx:xxxx:xxxx
0:8:xxxx:xxxx:xxxx:xxxx:xxxx:xxxx
0:9:xxxx:xxxx:xxxx:xxxx:xxxx:xxxx
0:10:xxxx:xxxx:xxxx:xxxx:xxxx:xxxx
0:11:xxxx:xxxx:xxxx:xxxx:xxxx:xxxx
0:12:xxxx:xxxx:xxxx:xxxx:xxxx:xxxx
0:13:xxxx:xxxx:xxxx:xxxx:xxxx:xxxx
0:14:xxxx:xxxx:xxxx:xxxx:xxxx:xxxx
0:15:xxxx:xxxx:xxxx:xxxx:xxxx:xxxx
0:16:xxxx:xxxx:xxxx:xxxx:xxxx:xxxx
0:17:xxxx:xxxx:xxxx:xxxx:xxxx:xxxx
0:18:xxxx:xxxx:xxxx:xxxx:xxxx:xxxx
0:19:xxxx:xxxx:xxxx:xxxx:xxxx:xxxx
0:20:xxxx:xxxx:xxxx:xxxx:xxxx:xxxx
0:21:xxxx:xxxx:xxxx:xxxx:xxxx:xxxx
0:22:xxxx:xxxx:xxxx:xxxx:xxxx:xxxx

```
li $s5, sub_405A50  
li $s4, j_free  
addiu $a0, $sp, 0x70+var_58  
jal sub_406644  
move $a1, $s1  
(Delay slot)
```



```
li $s5, sub_405A50  
li $s4, j_free  
addiu $a0, $sp, 0x70+var_58  
move $a1, $s1  
jal sub_406644
```

shellcode

R0P gadgets

```
0:1:xxxx:xxxx:xxxx:xxxx:xxxx:xxxx  
0:2:xxxx:xxxx:xxxx:xxxx:xxxx:xxxx  
0:3:xxxx:xxxx:xxxx:xxxx:xxxx:xxxx
```

...

```
0:20:xxxx:xxxx:jump $+8:xxxx:xxxx  
0:21:xxxx:xxxx:jump $+8:xxxx:xxxx  
0:22:xxxx:xxxx:jump $+8:xxxx:xxxx
```

shellcode

```
addi s8, s0, 1
```

R0P gadgets

```
221e:1:xxxx:xxxx:xxxx:xxxx:xxxx:xxxx
```

```
221e:2:xxxx:xxxx:xxxx:xxxx:xxxx:xxxx
```

```
221e:3:xxxx:xxxx:xxxx:xxxx:xxxx:xxxx
```

...

```
221e:20:xxxx:xxxx:xxxx:xxxx:xxxx:xxxx
```

```
221e:21:xxxx:xxxx:xxxx:xxxx:xxxx:xxxx
```

```
221e:22:xxxx:xxxx:xxxx:xxxx:xxxx:xxxx
```

shellcode

ROP gadgets

221e:1:xxxx:xxxx:xxxx:xxxx:xxxx:xxxx

221e:2:xxxx:xxxx:xxxx:xxxx:xxxx:xxxx

221e:3:xxxx:xxxx:xxxx:xxxx:xxxx:xxxx

...

221e:20:xxxx:xxxx:xxxx:xxxx:xxxx:xxxx

221e:21:xxxx:xxxx:xxxx:xxxx:xxxx:xxxx

221e:22:xxxx:xxxx:xxxx:xxxx:xxxx:xxxx

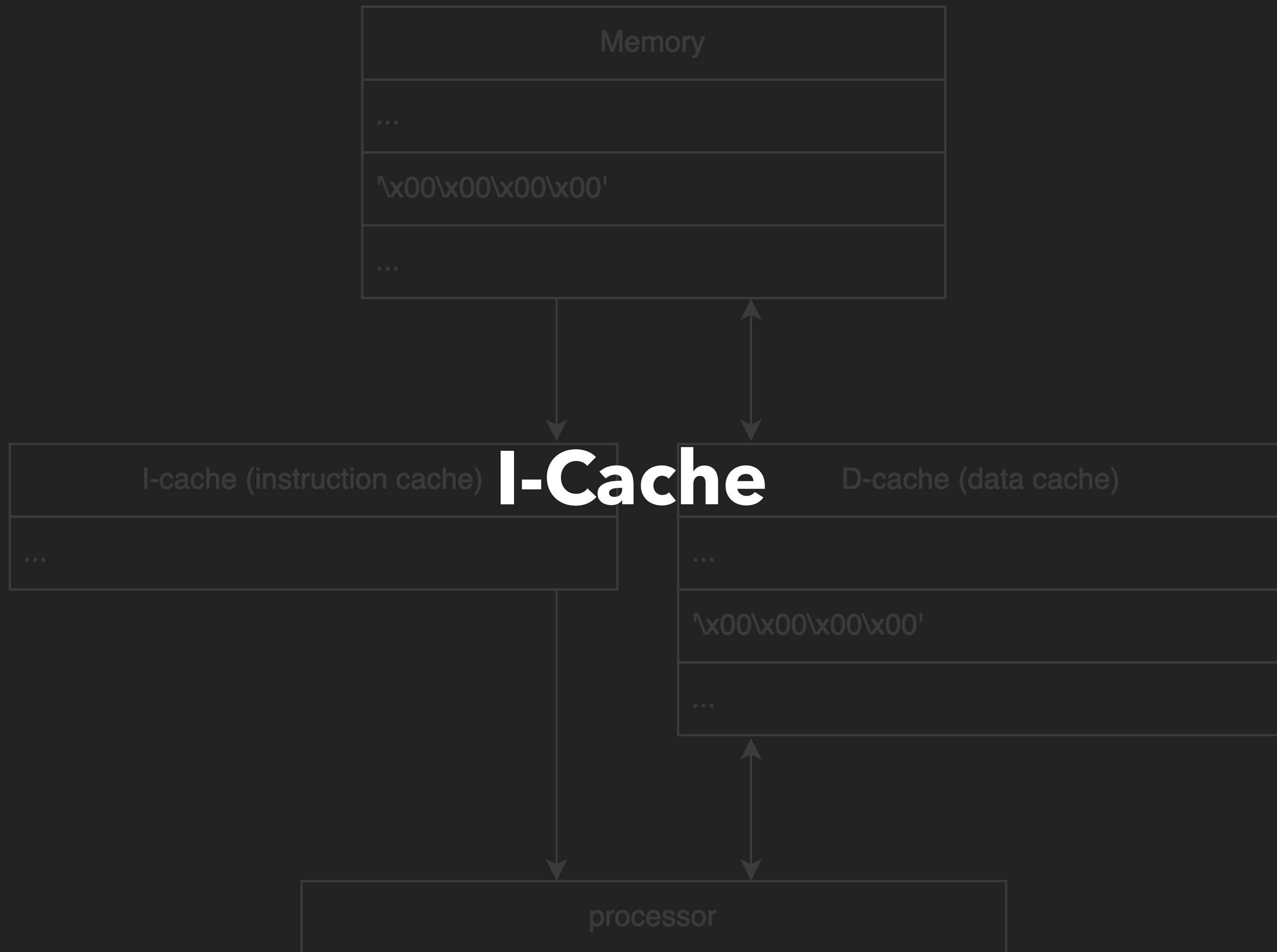
1. write

2. jump

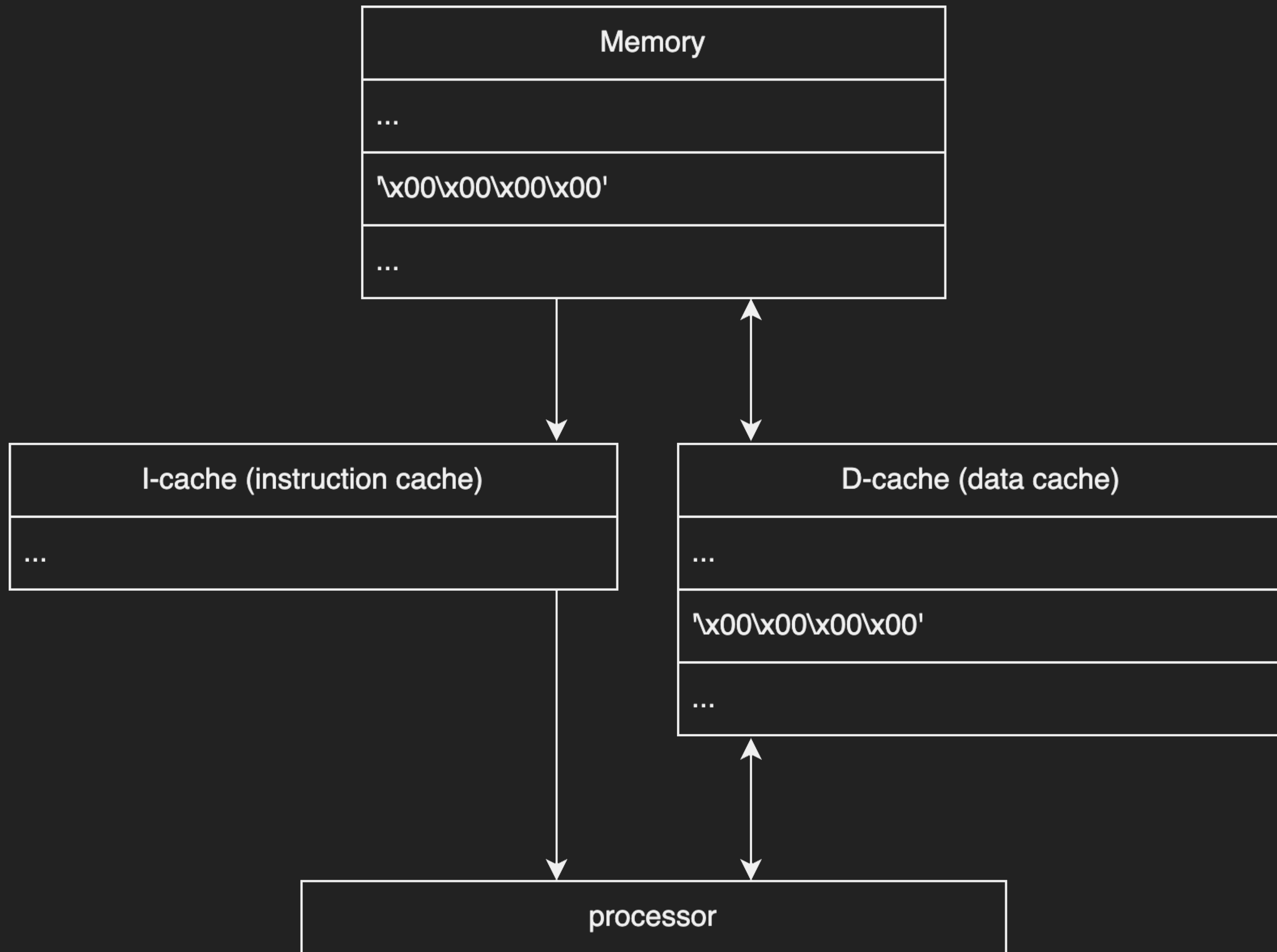
3. jump

jalr \$sp

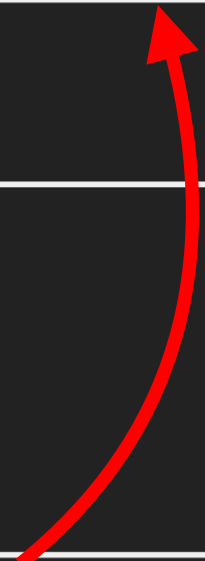
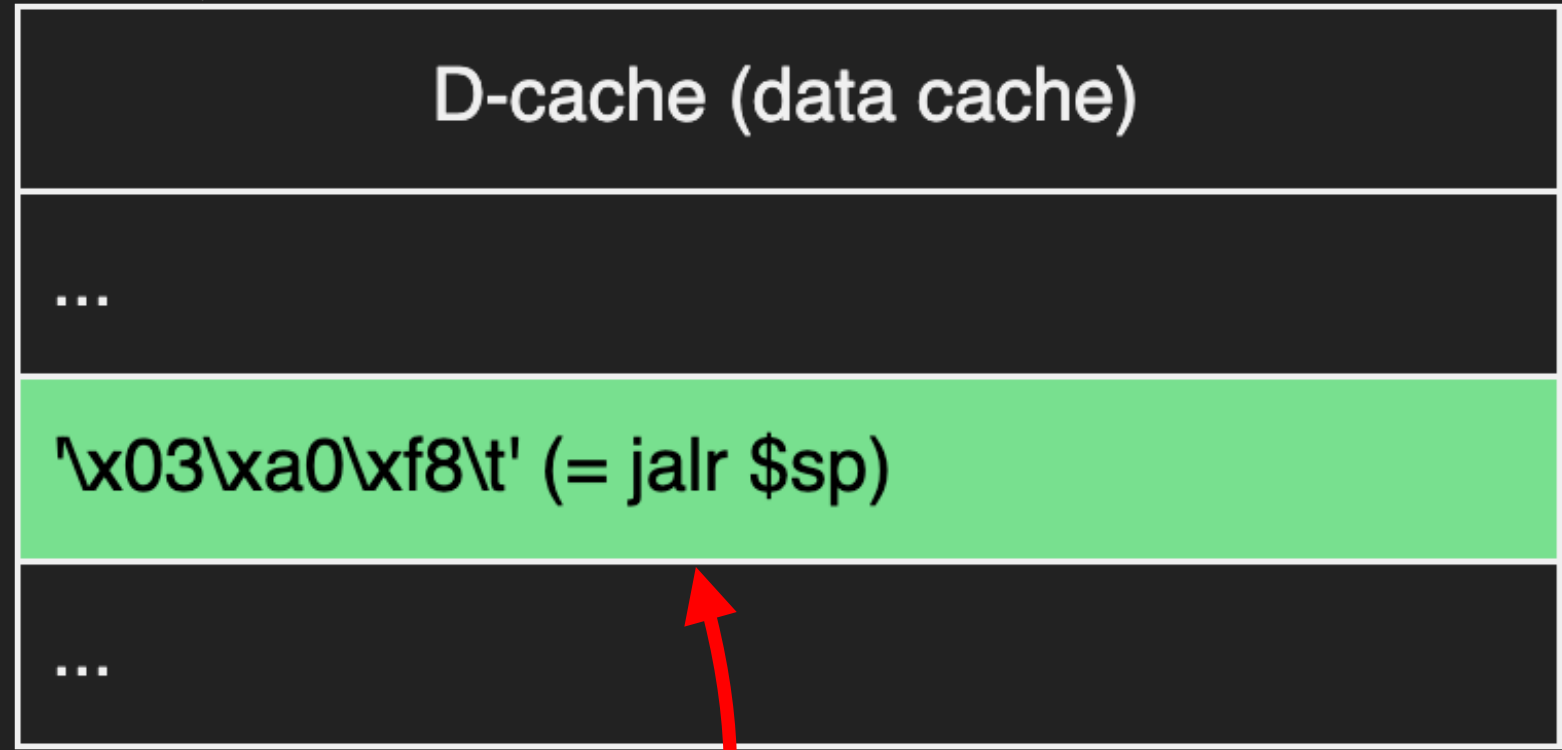
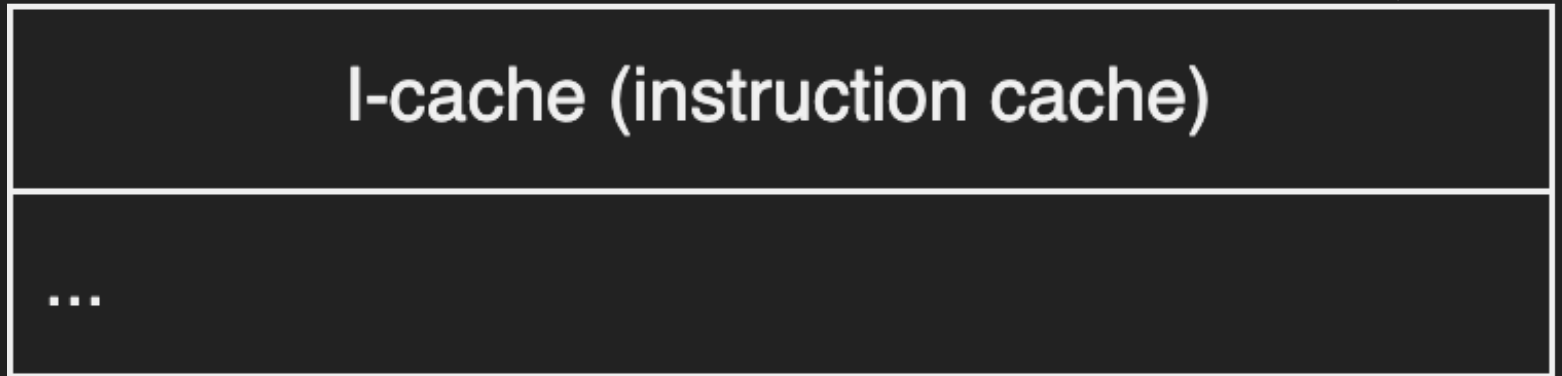
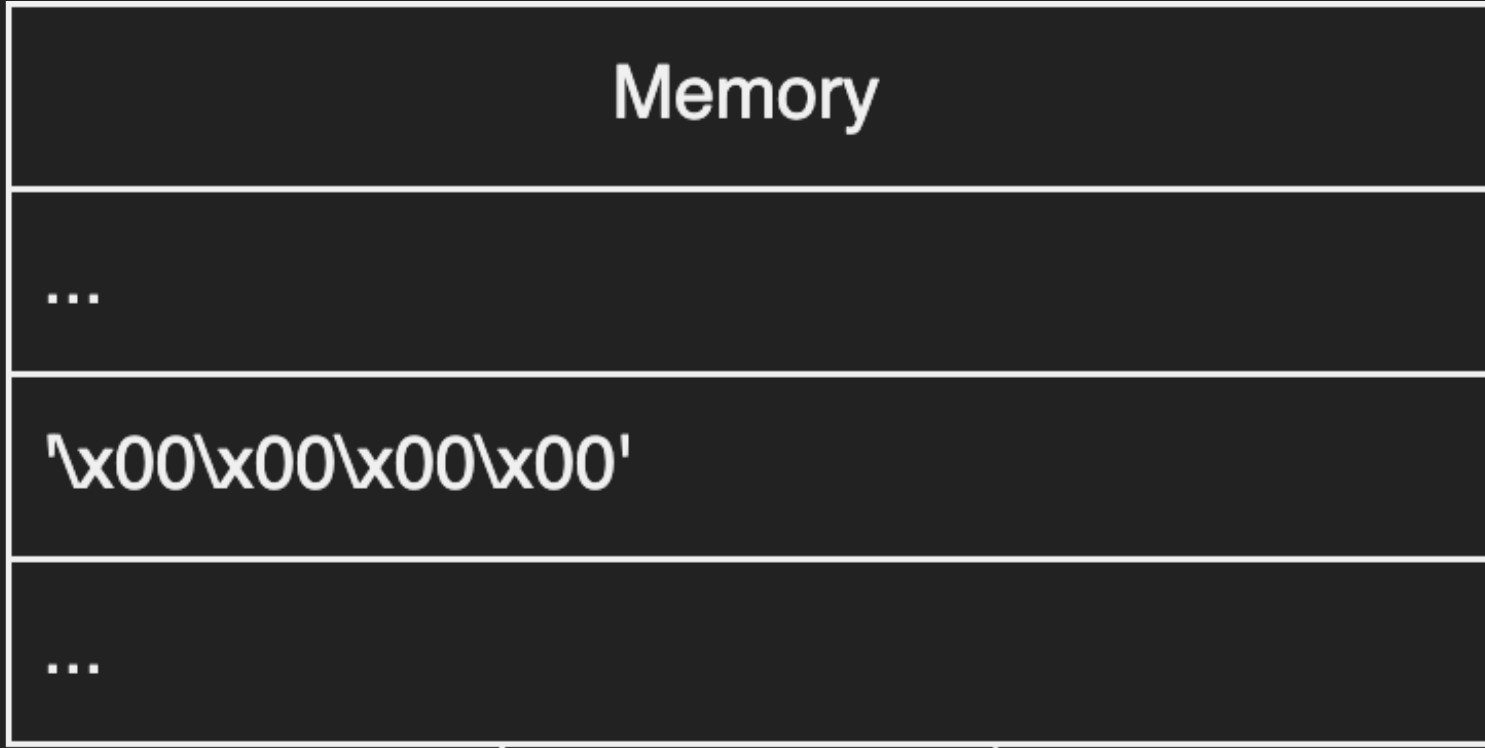
shellcode



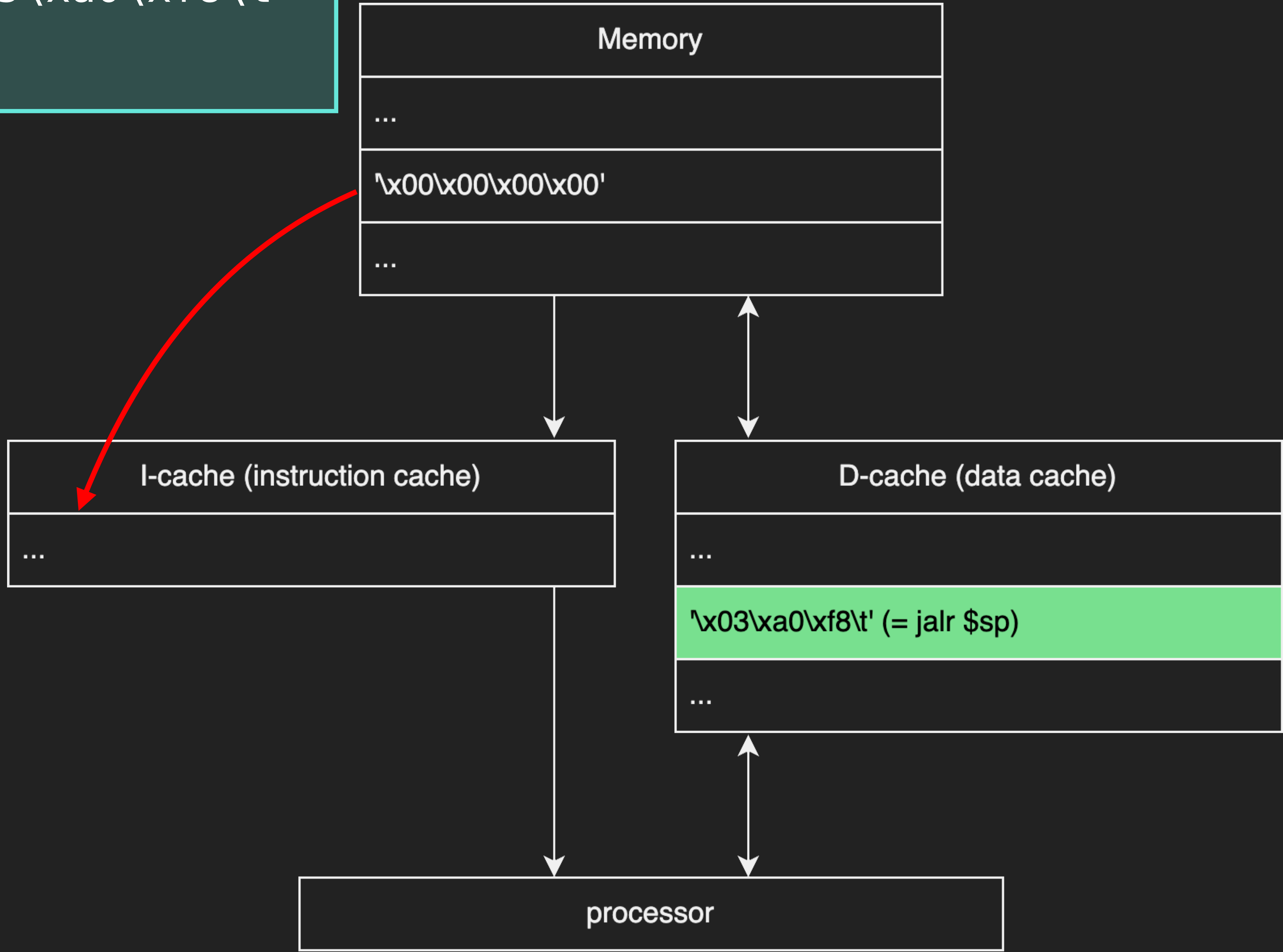
I-Cache



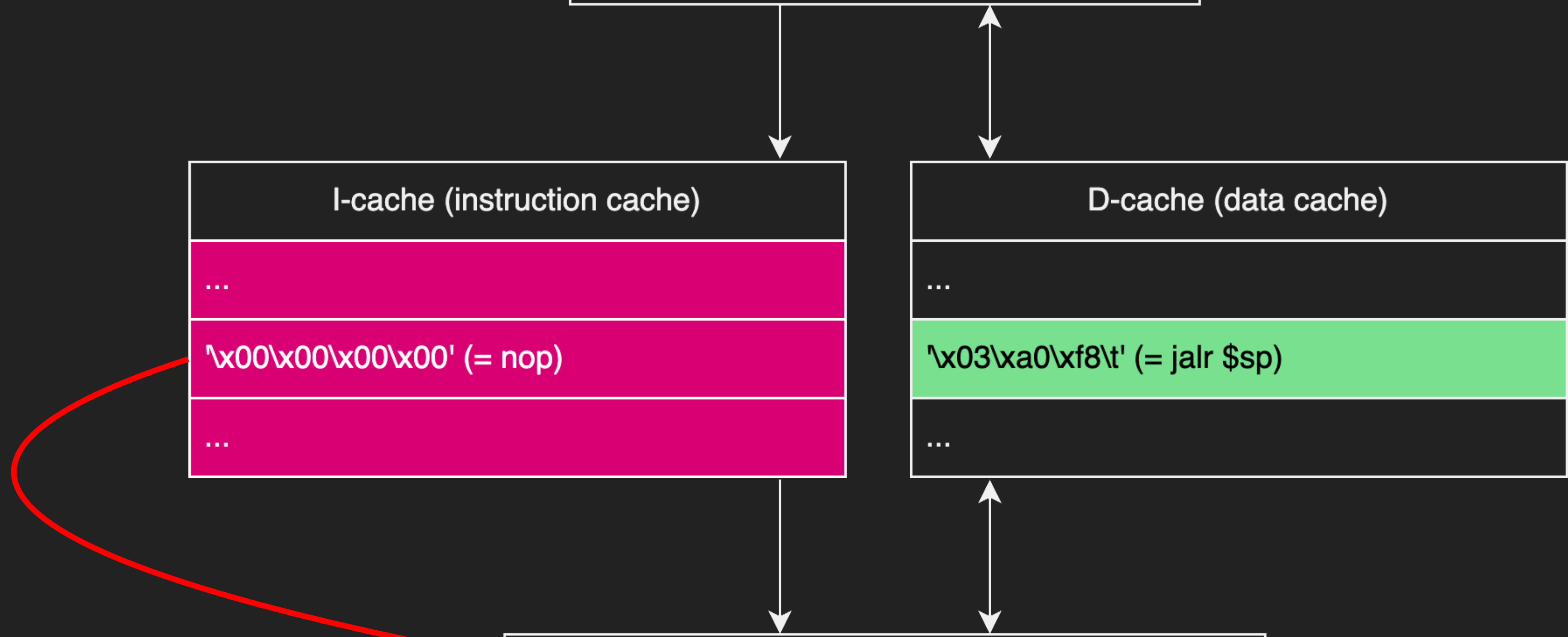
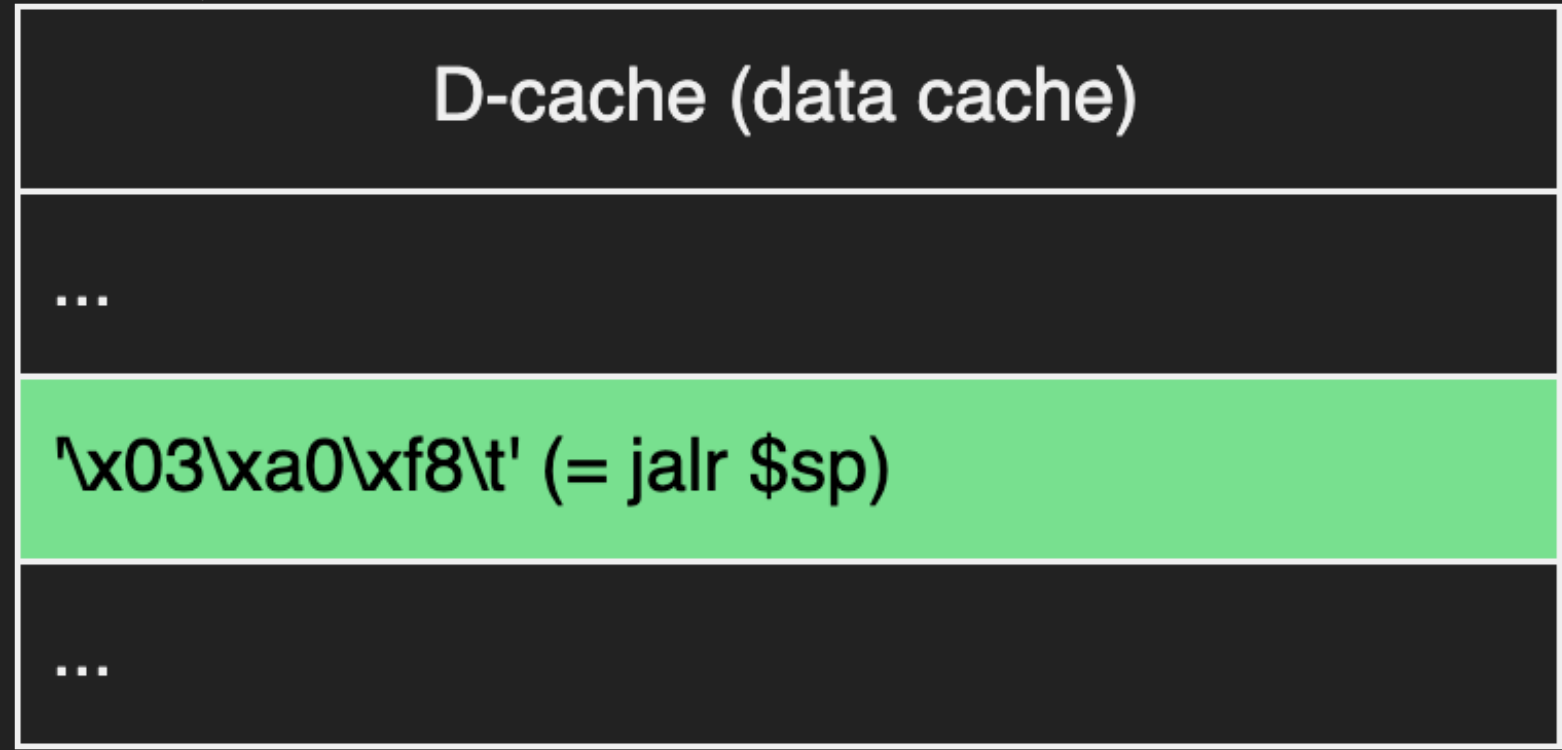
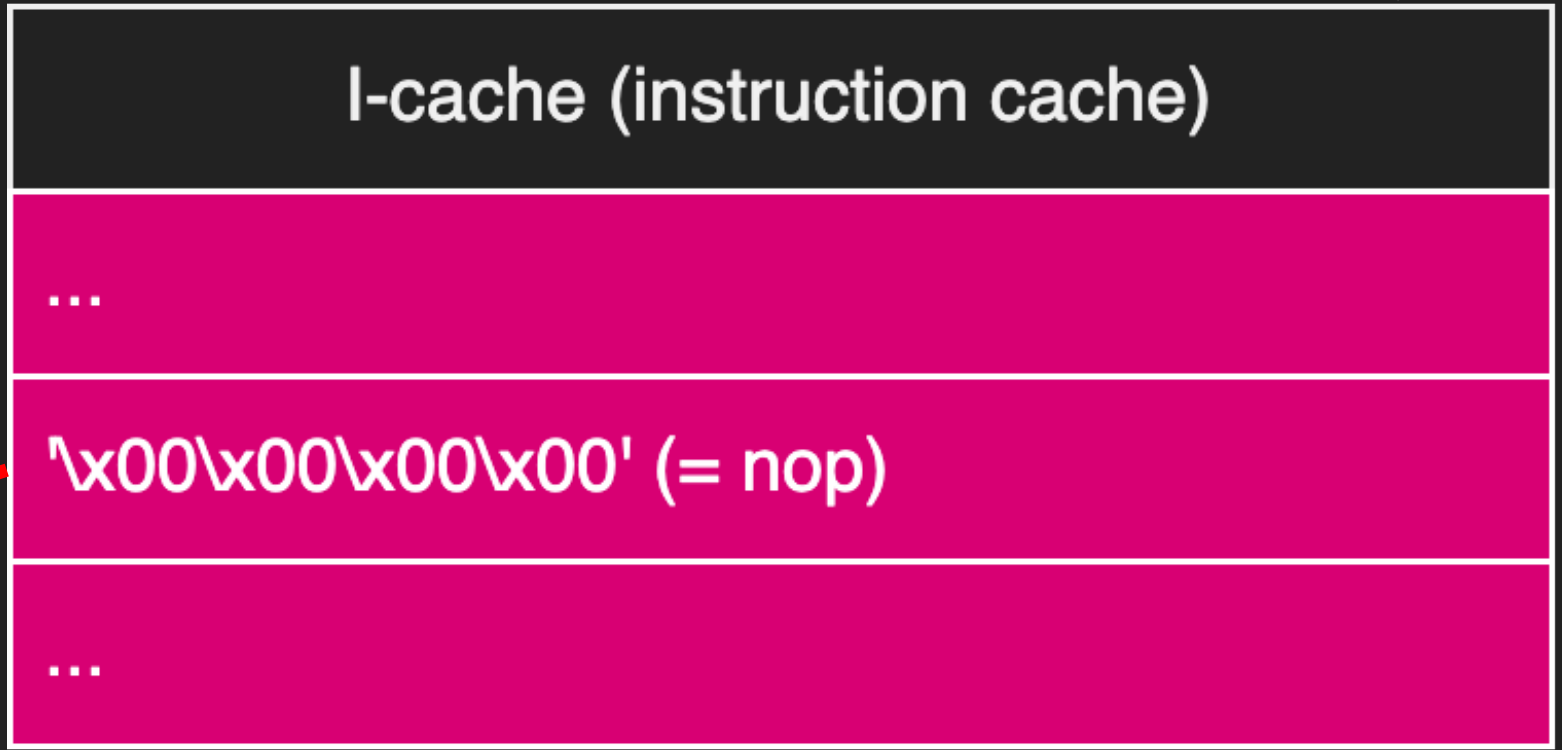
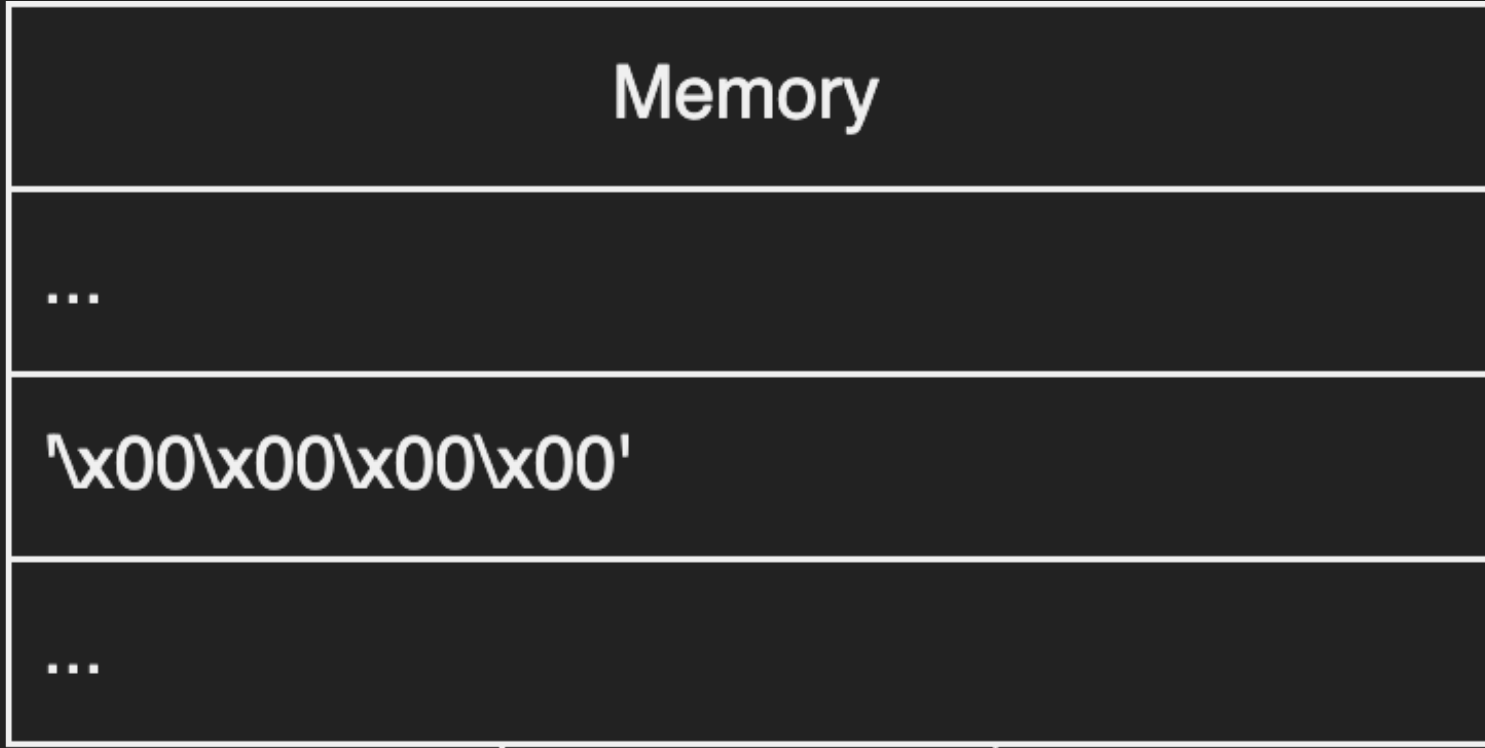
```
write adr, '\x03\xa0\xf8\t'
```



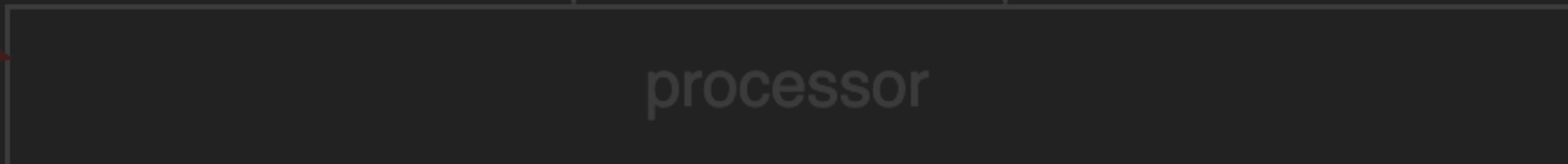
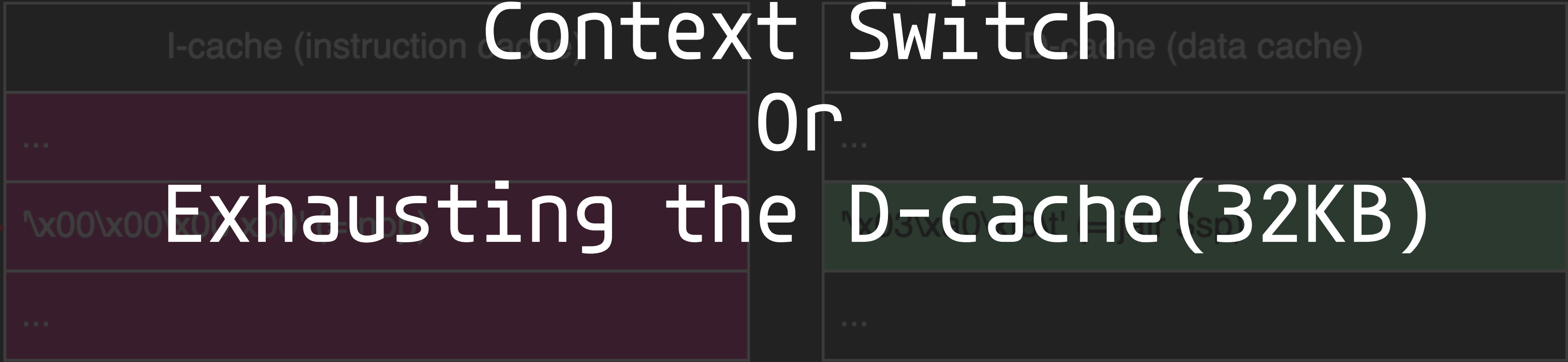
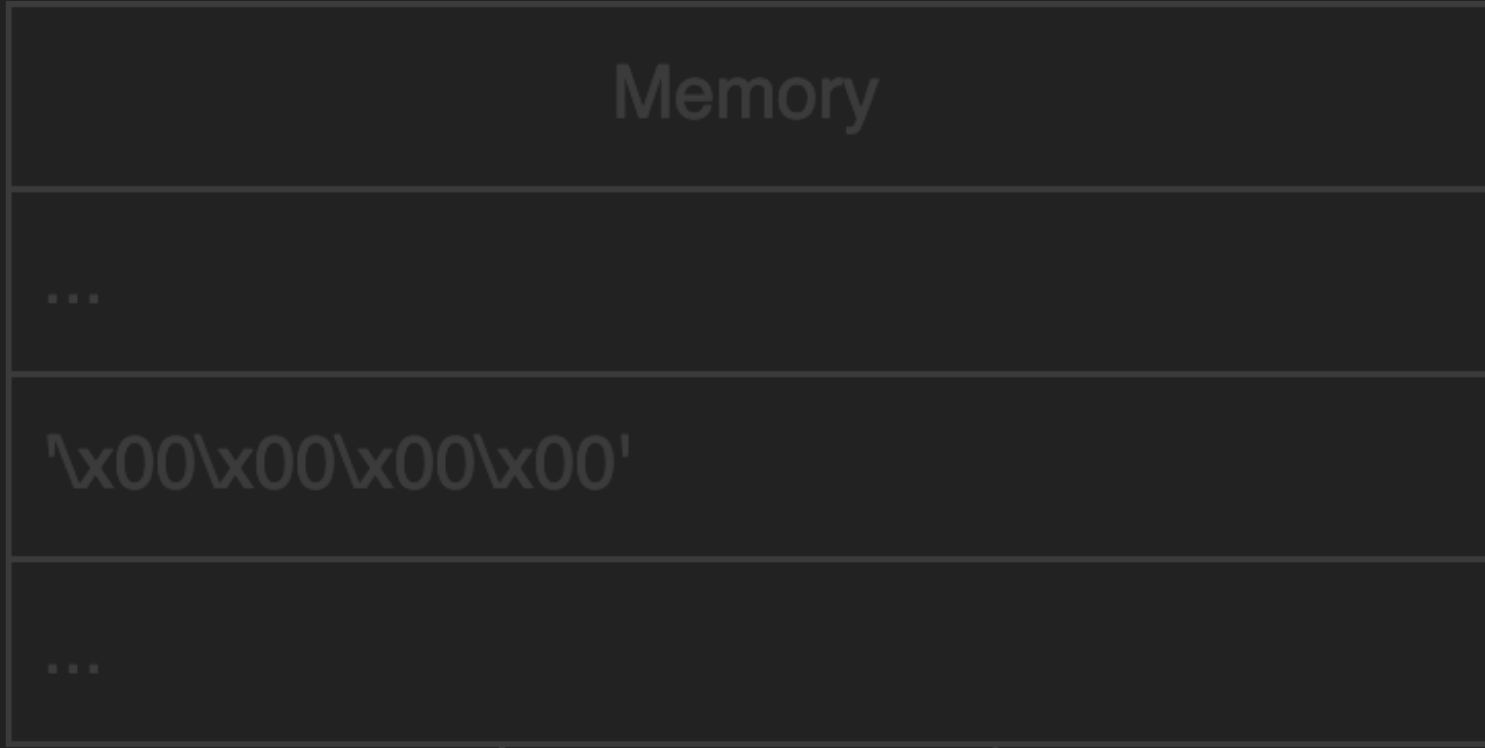
```
write adr, '\x03\xa0\xf8\t'  
jump adr
```



```
write adr, '\x03\xa0\xf8\t'  
jump adr
```



```
write adr, '\x03\xa0\xf8\t'  
jump adr
```





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

```
[pwndbg> vmmmap  
LEGEND: STACK | HEAP | CODE | DATA | RWX | RODATA  
0x400000 0x40d000 r-xp d000 0 /ram/pckg/ipv6/nova/bin/radvd  
0x41c000 0x41d000 rw-p 1000 c000 /ram/pckg/ipv6/nova/bin/radvd  
0x41d000 0x427000 rwxp a000 0 [heap]  
0x77031000 0x77078000 r-xp 47000 0 /lib/libuClibc-0.9.33.2.so  
0x77078000 0x77087000 ---p f000 0 [anon_77078]  
0x77087000 0x77088000 r--p 1000 46000 /lib/libuClibc-0.9.33.2.so  
0x77088000 0x77089000 rw-p 1000 47000 /lib/libuClibc-0.9.33.2.so  
0x77089000 0x7708b000 rw-p 2000 0 [anon_77089]  
0x7708b000 0x770b9000 r-xp 2e000 0 /lib/libgcc_s.so.1  
0x770b9000 0x770c8000 ---p f000 0 [anon_770b9]  
0x770c8000 0x770c9000 rw-p 1000 2d000 /lib/libgcc_s.so.1
```

RADVD

```
if ( v23->enable_advisory )
{
    lifetime = v23->lifetime;
    length = handler_1->DNS_tree.length;
    if ( length )
        length = addDNS((int)&RA_raw[pos], &handler_1->DNS_tree, (lifetime >> 1) + lifetime);
    expire_pos = length + pos;
    v32 = handler_1->expired_DNS_tree.length;
    if ( v32 )
        v32 = addDNS((int)&RA_raw[expire_pos], &handler_1->expired_DNS_tree, 0);
    pos = v32 + expire_pos;
    tree_begin = a1->prefix_tree.tree_begin;
}
else
```

```
[pwndbg> vmmmap
LEGEND: STACK | HEAP | CODE | DATA | RWX | RODATA
0x400000 0x40d000 r-xp d000 0 /ram/pckg/ipv6/nova/bin/radvd
0x41c000 0x41d000 rw-p 1000 c000 /ram/pckg/ipv6/nova/bin/radvd
0x41d000 0x46f000 rwxp 52000 0 [heap]
0x77931000 0x77978000 r-xp 47000 0 /lib/libuClibc-0.9.33.2.so
0x77978000 0x77987000 ---p f000 0 [anon_77978]
0x77987000 0x77988000 r--p 1000 46000 /lib/libuClibc-0.9.33.2.so
0x77988000 0x77989000 rw-p 1000 47000 /lib/libuClibc-0.9.33.2.so
0x77989000 0x7798b000 rw-p 2000 0 [anon_77989]
0x7798b000 0x779b9000 r-xp 2e000 0 /lib/libgcc_s.so.1
0x779b9000 0x779c8000 ---p f000 0 [anon_779b9]
```



Terminal

/tmp

```
[% python3 exploit.py ]
```

```
[*] [Exploit Script]: Starting Exploit
```

```
[*] [Exploit Script]: Generating shellcode
```

```
[*] [Exploit Script]: Sending RA with RDNSS
```

```
[*] [Exploit Script]: Exploit done
```

```
[*] Switching to interactive mode
```

```
>>> / # / # / # uname -a
```

```
Linux MikroTik 3.3.5 #1 Tue Oct 11 14:41:12 UTC 2022 mips GNU/Linux
```

CVE-2023-32154

- Fixed at:
 - Long-term Release 6.48.7
 - Stable Release 6.49.8, 7.10
 - Testing Release 7.10rc6
- The vulnerable code has existed at least since v6.0

CVE-2023-32154

- Fixed at:

- Long-term Release 6.48.7
- Stable Release 6.49.8, 7.10
- Testing Release 6.49.9

Release 6.0

2013-05-20

- The vulnerable code has existed at least since v6.0

CVE-2023-32154

• Fixed at: **No one with sanity would like to dive into the details of Nova Binary**

• Long-term Release 6.48.7, 7.10

• Stable Release 6.49.8, 7.10

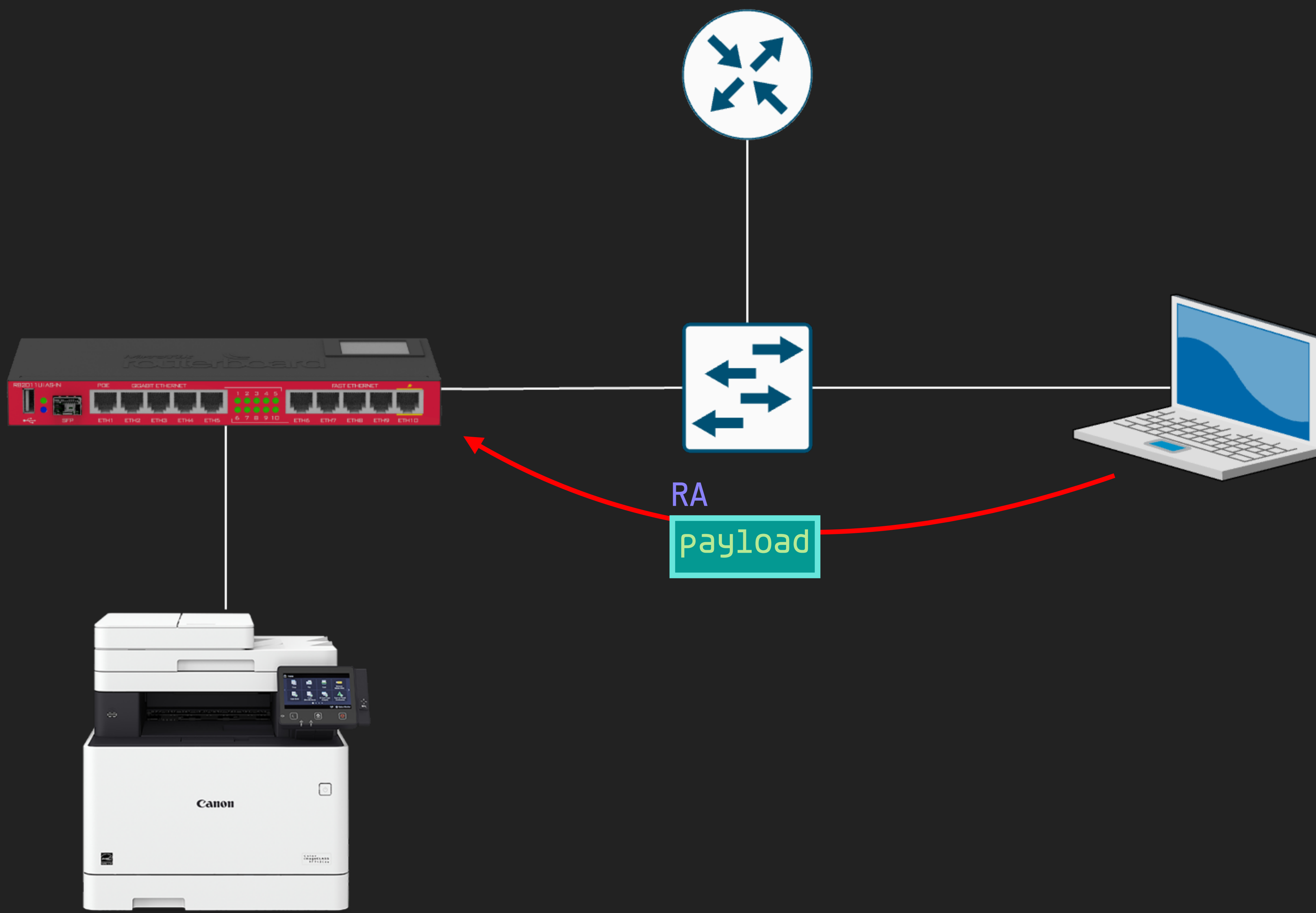
Release 6.0

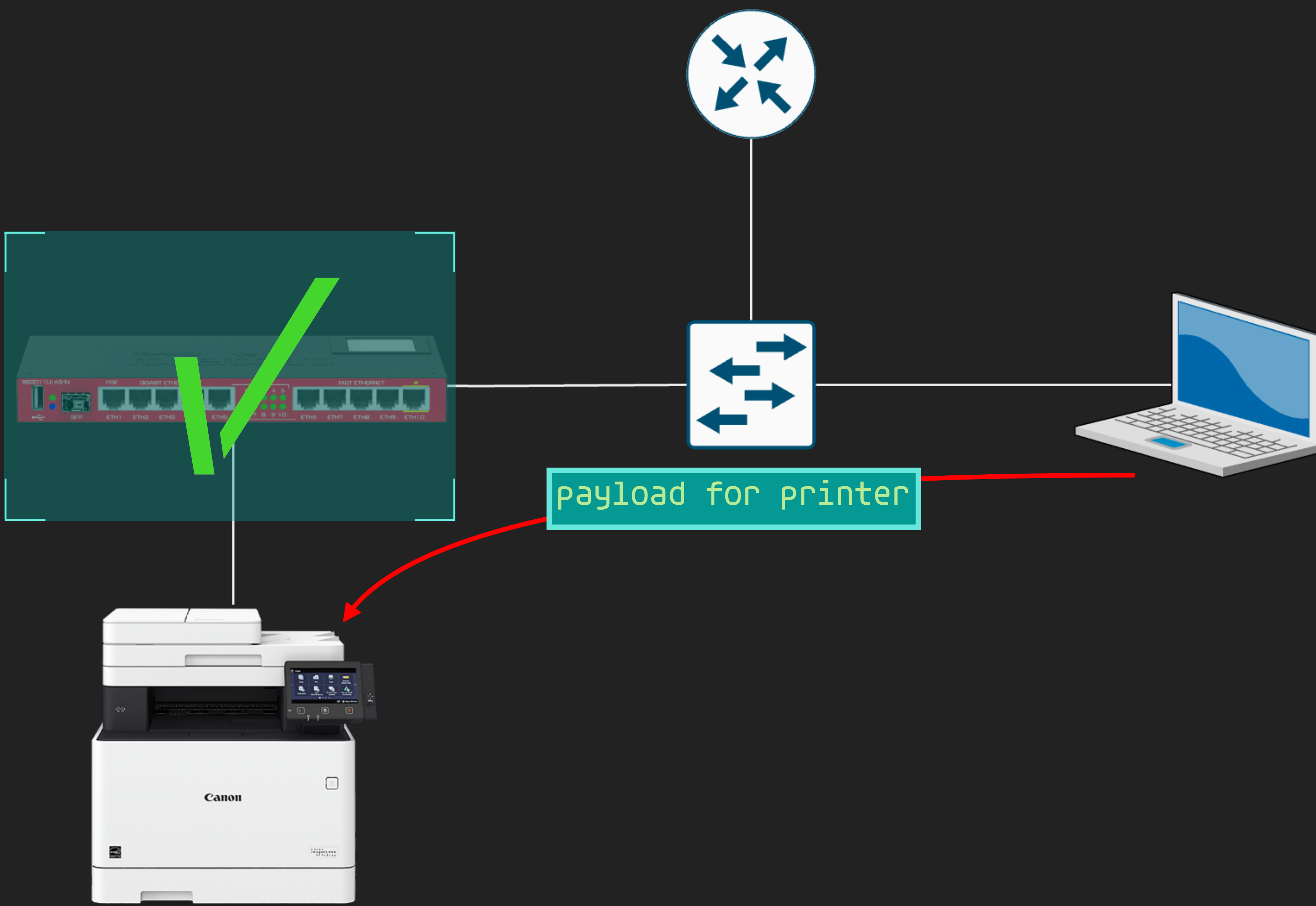
2013-05-20

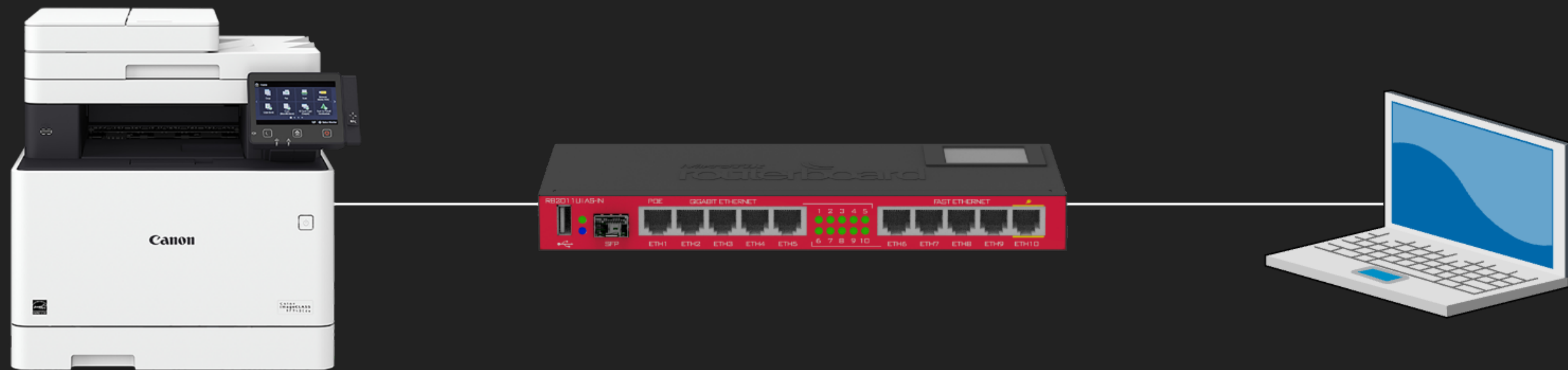
• Testing Release 6.49.8, 7.10

• The vulnerable code has existed at least since v6.0

Q.E.D.









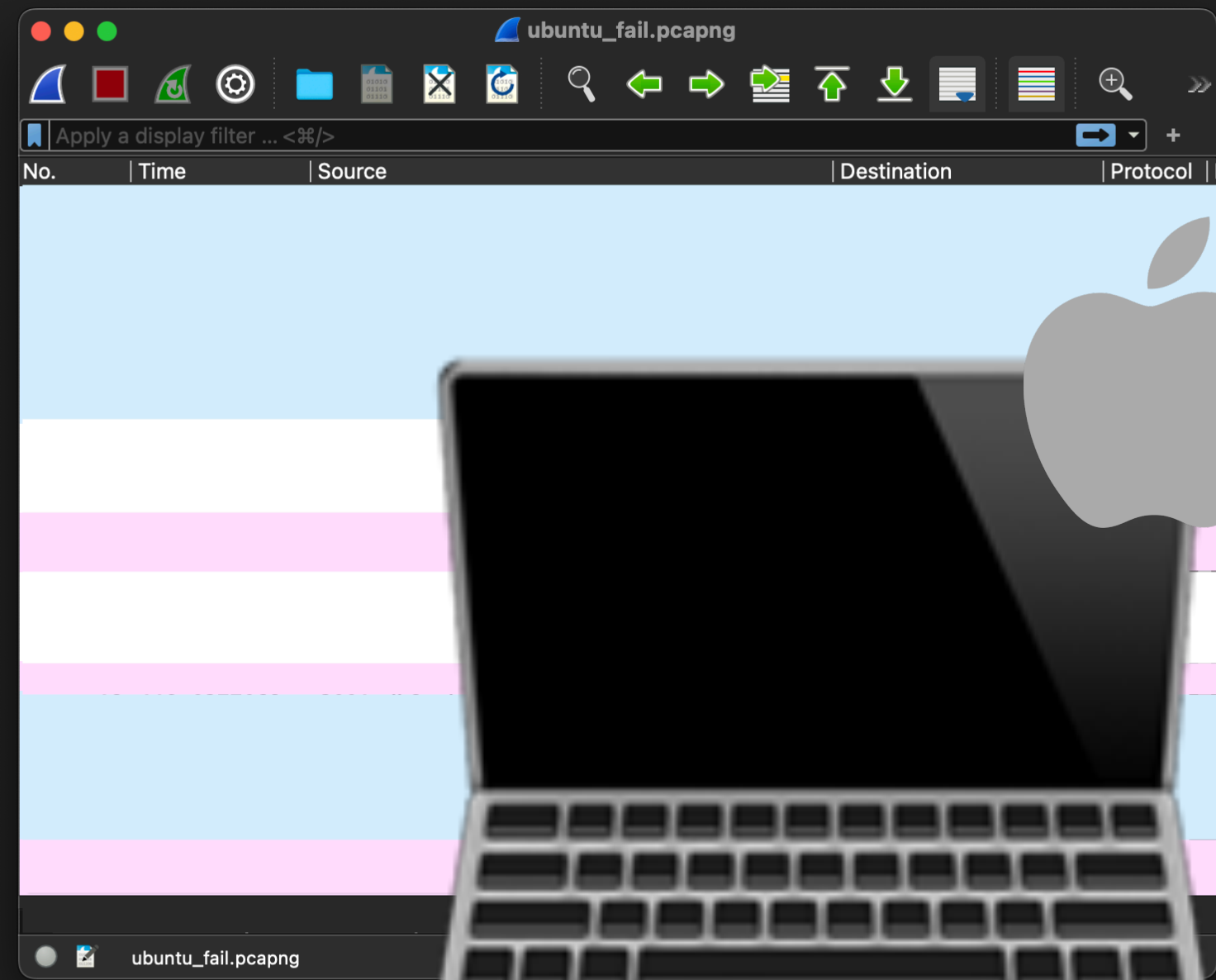
\$100,000

But...

But

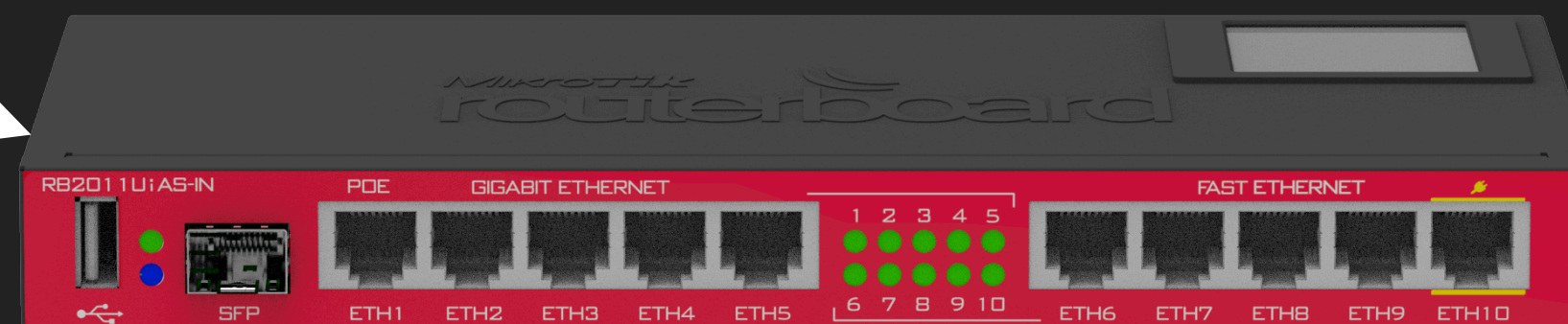
- Our exploit only worked on MacOS and failed on Ubuntu, whether it's a VM or not.



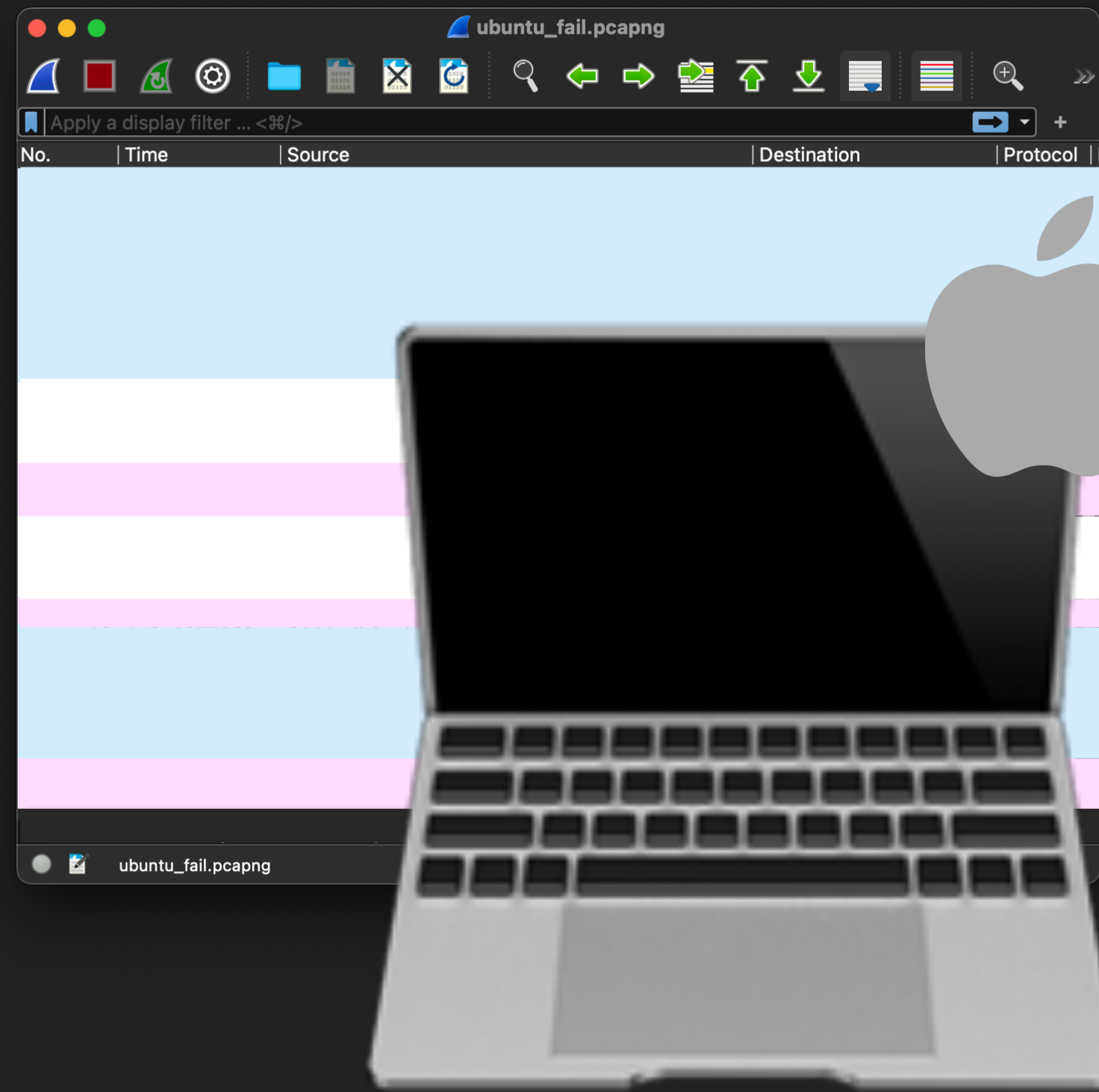


1. exploit & record

Succeed!

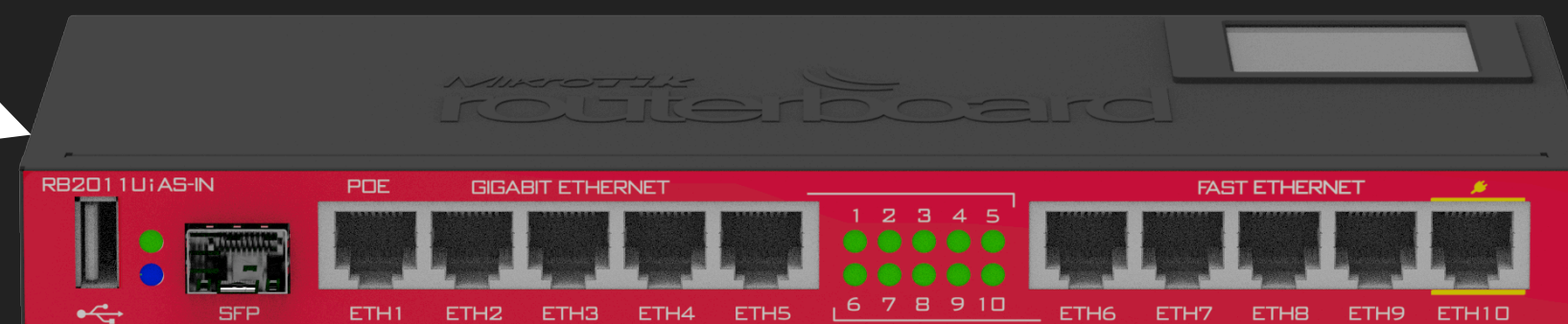


2. Dump traffics

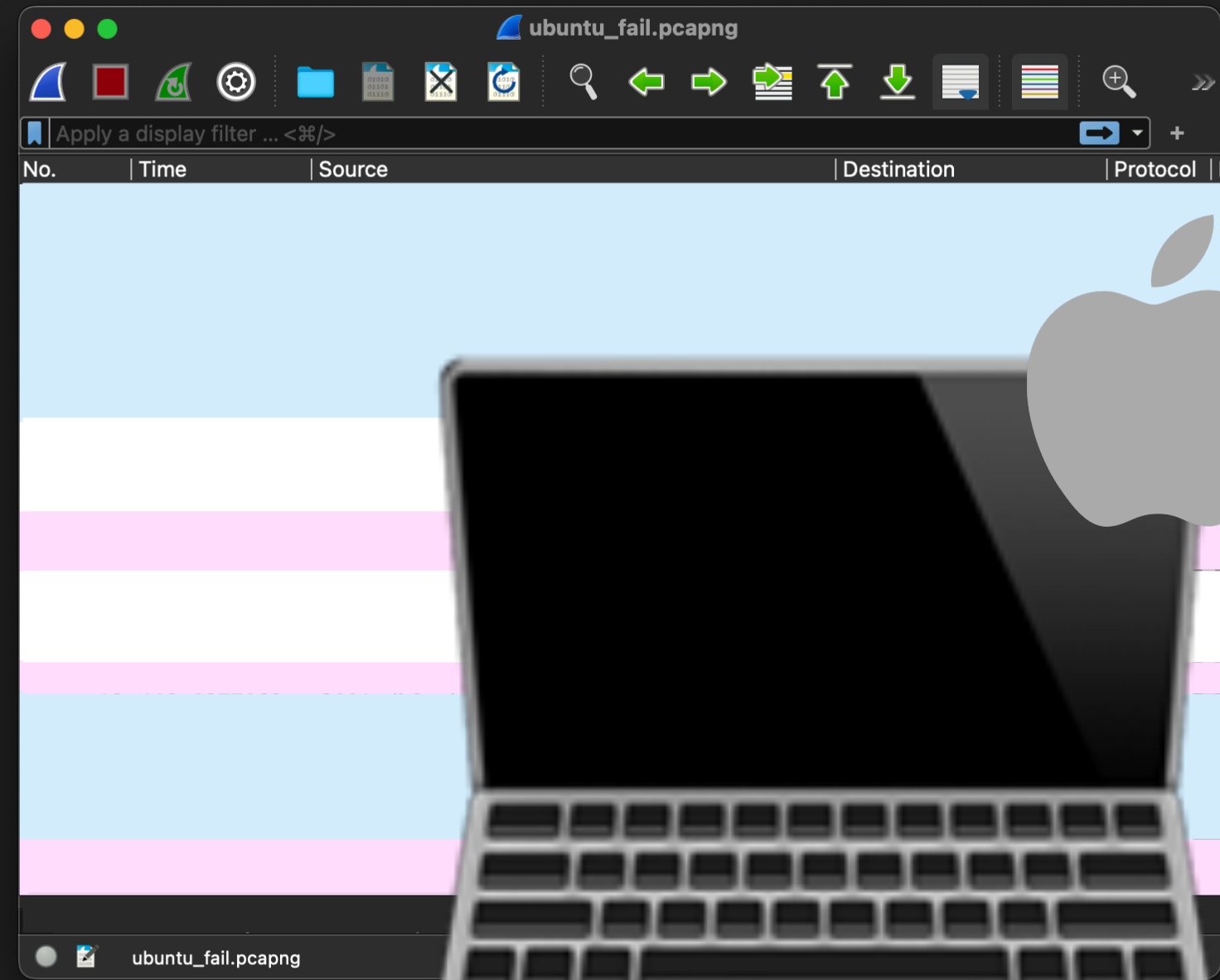


1. exploit & record

Succeed!



2. Dump traffics

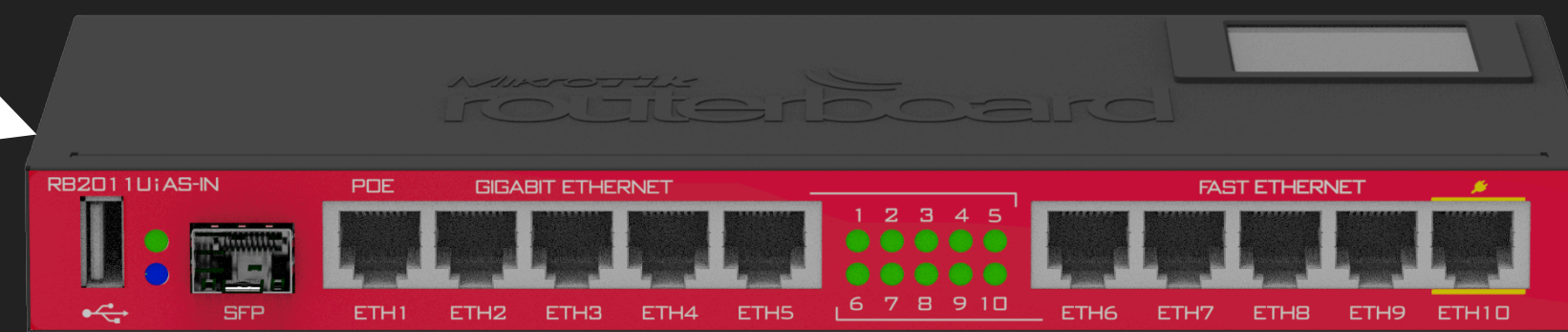


1. exploit & record

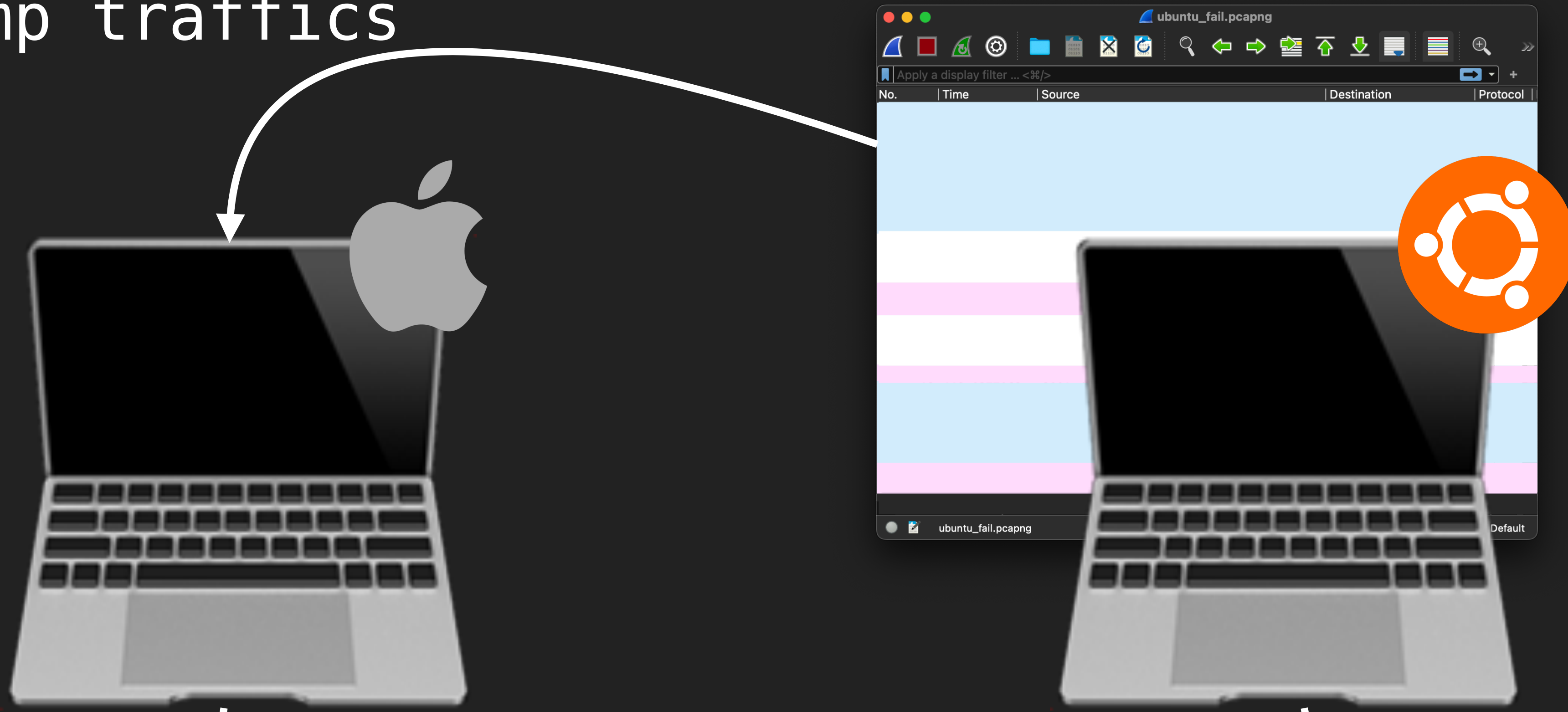
Succeed!

Fail!

3. replay



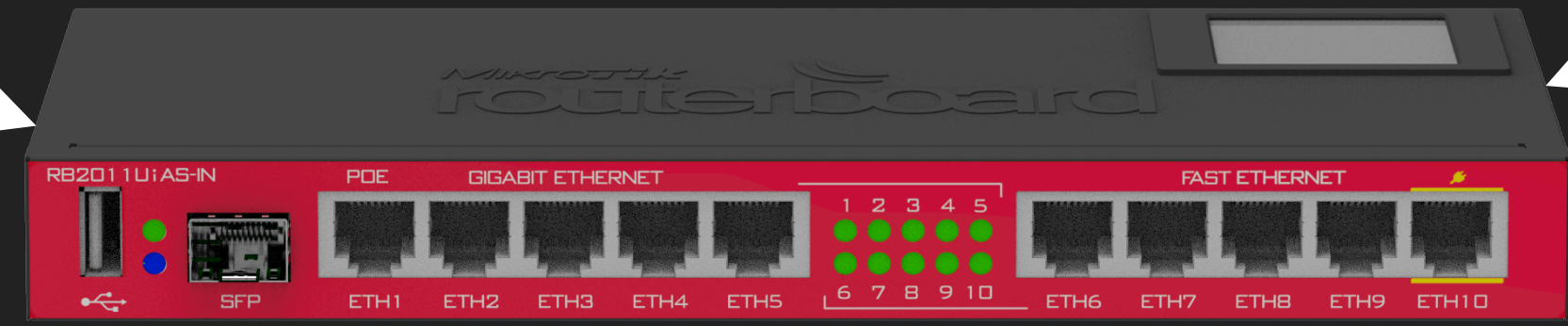
2. Dump traffics



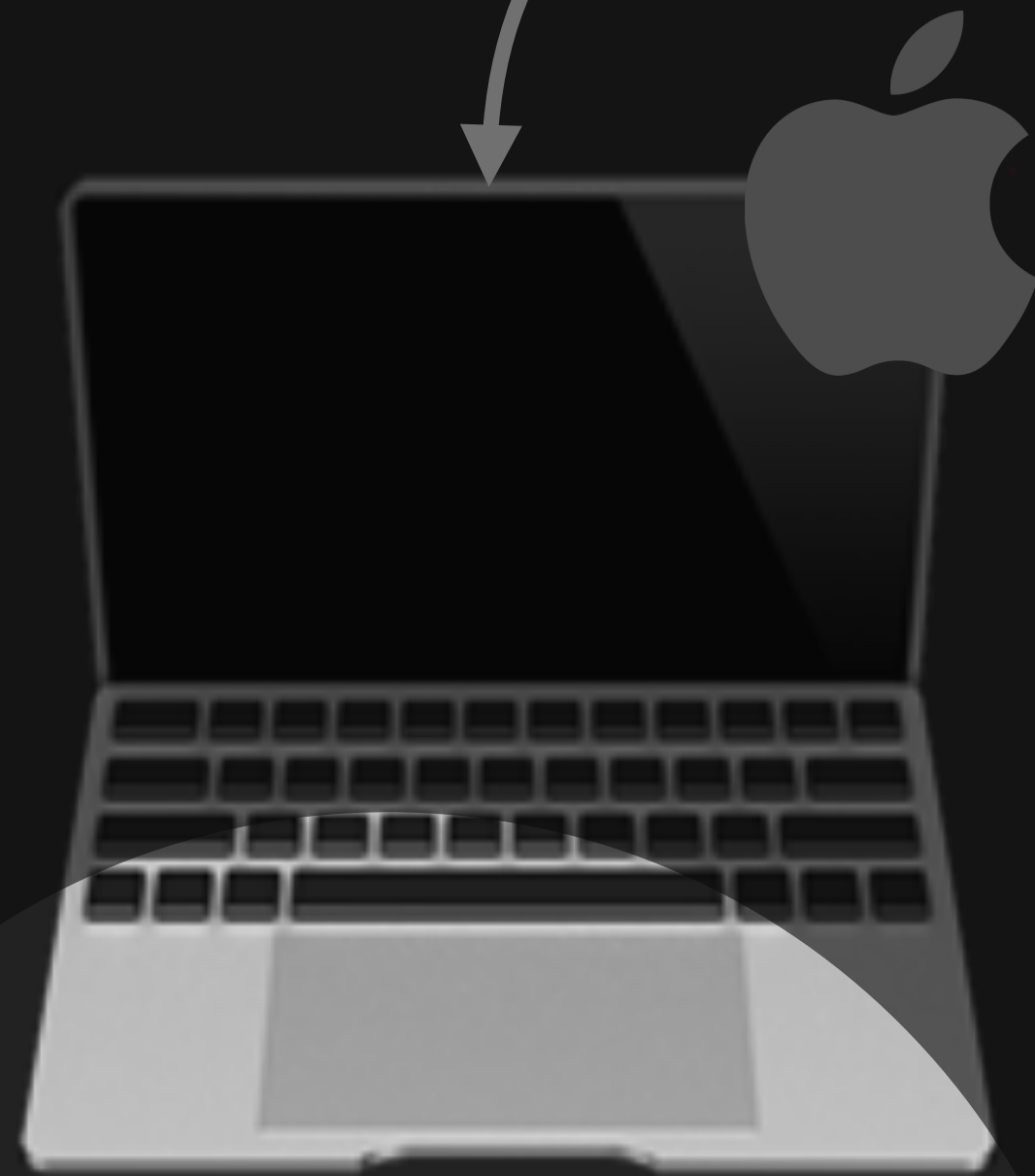
1. exploit & record

Fail!

3. replay



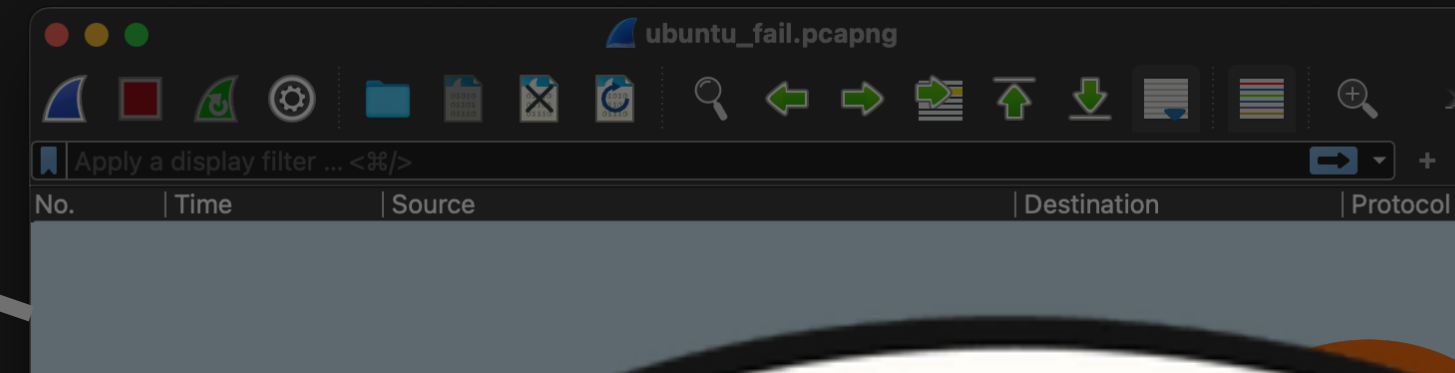
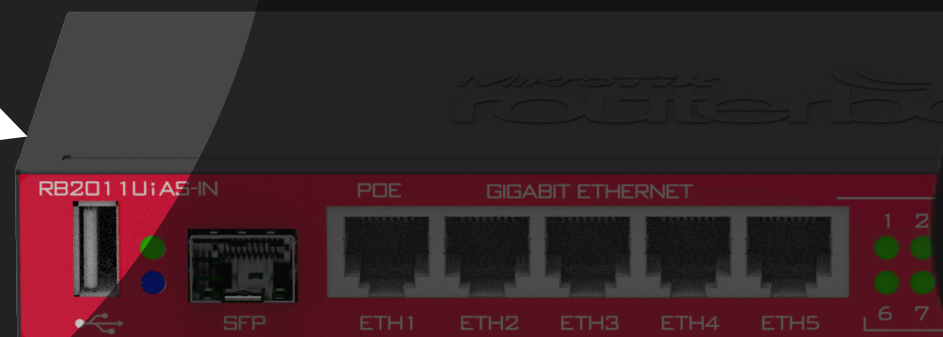
2. Dump traffics



1. explo

Succeed!

3. replay



Guess:

An OS reorders the packets

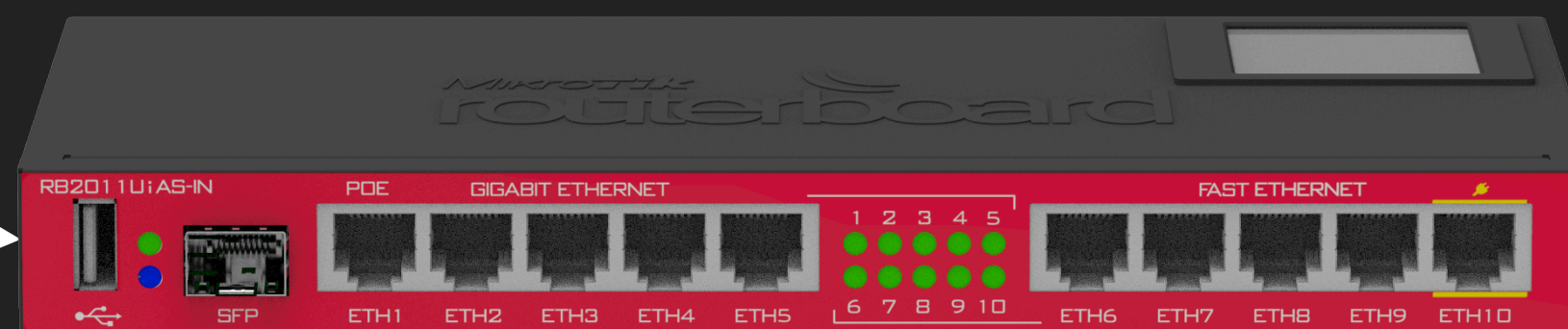


exploit



exploit

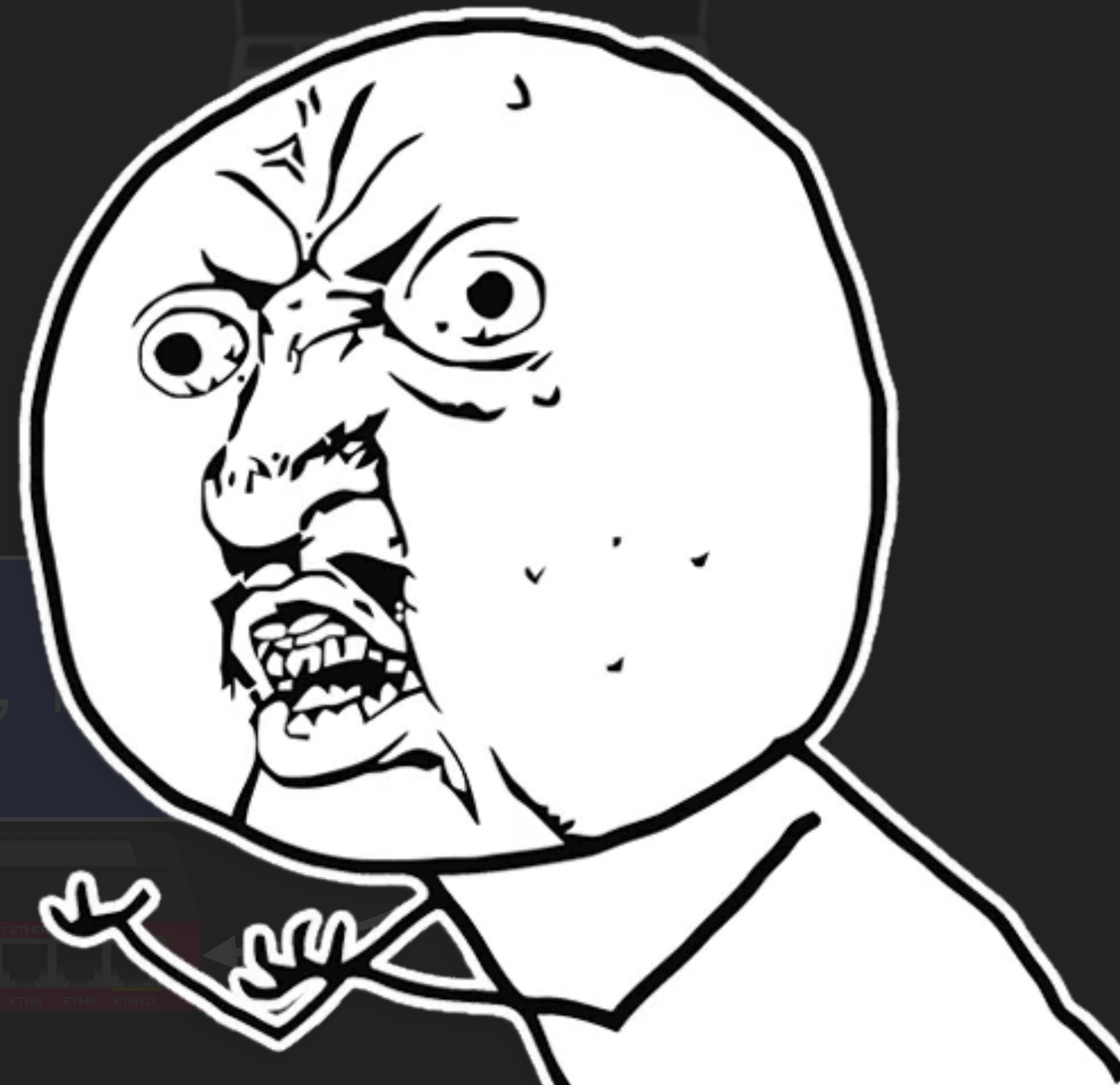
```
...  
int s = socket(AF_PACKET, SOCK_RAW, htons(ETH_P_IPV6)) ;  
...
```

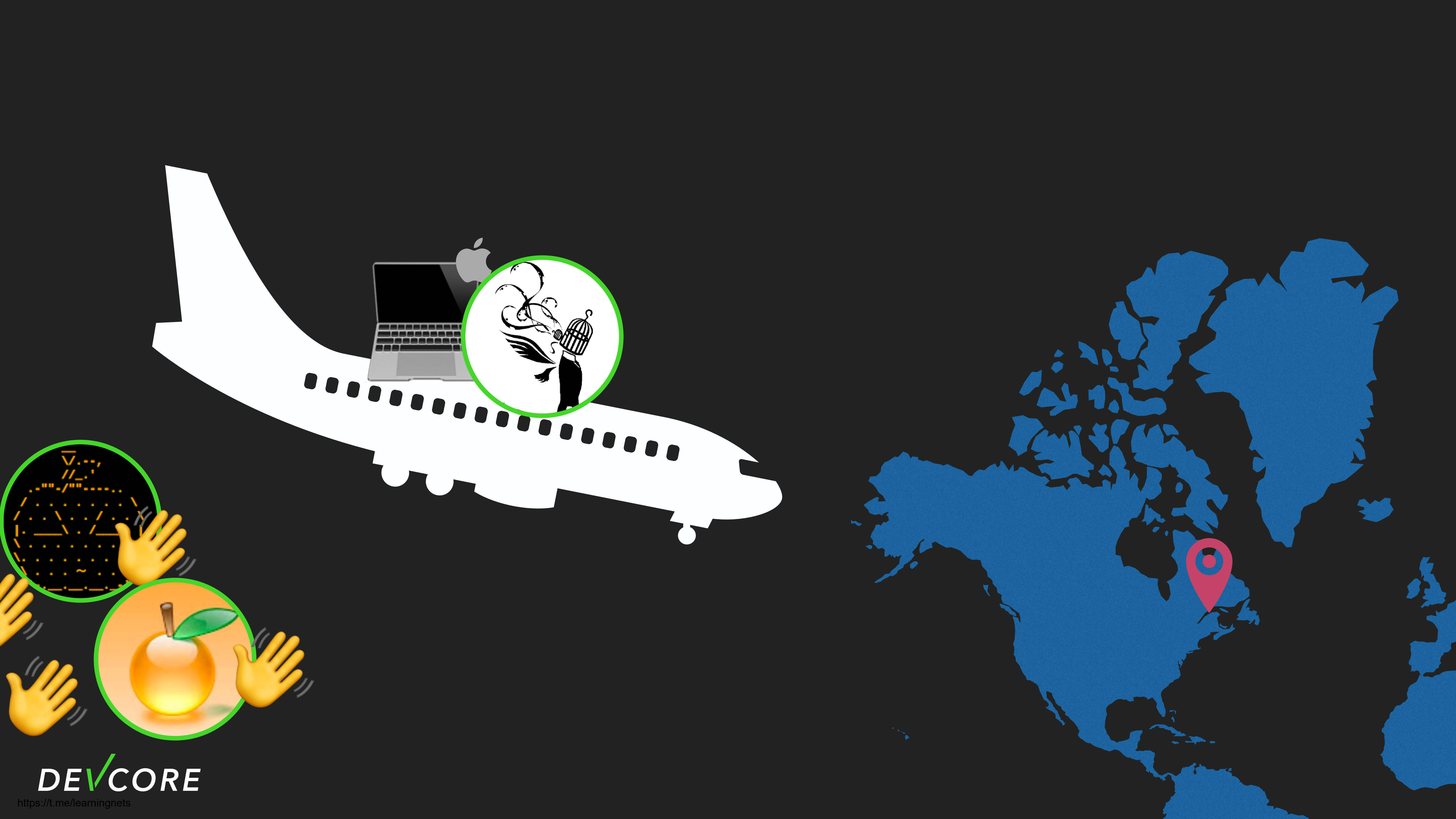


exploit

Look the same

```
int s = socket(AF_PACKET, SOCK_RAW,
```





DEV/CORE

<https://t.me/learningnets>



0.0004



39	149.193503	fe80::
40	149.193659	fe80::
41	149.193788	fe80::
42	149.193898	fe80::
43	149.194025	fe80::
44	149.194151	fe80::
45	149.194257	fe80::
46	149.194373	fe80::

0.1560



9	117.1194004...	fe80::
10	117.1489369...	fe80::
11	117.2049854...	fe80::
12	117.2609559...	fe80::
13	117.2731308...	::
14	117.2828079...	fe80::
15	117.3249979...	fe80::
16	117.3729505...	fe80::
17	117.4169443...	fe80::

390x!!



In callback function (recv RA from WAN)

```
if ( DNS_vector_now == DNS_vector_end )
{
    if ( !operator==(IPAddr6)((int *)&new_DNS_vector, (int *)&handler_1->DNS_raw) )
    {
        clean_remoteObj(handler_1);
        vector_base::swap_raw(&new_DNS_vector, &handler_1->DNS_raw);
        v163 = (void **)handler_1->DNS_raw.end;
        for ( IPAddr6 = handler_1->DNS_raw.start; IPAddr6 != v163; IPAddr6 += 4 )
        {
            *(_DWORD *)v181 = 0;
            v183[0] = 0;
            v191[0] = 0;
            v191[1] = 0;
        }
    }
}
```

In RAroutine (send RA to LAN)

```
lifetime = ndsetting->lifetime;
length = handler_1->DNS_treeE4.length;
if ( length )
    length = addDNS((int)&raw_packet[v18], &handler_1->DNS_treeE4, (lifetime >> 1) + lifetime);
```

RADVD

- In callback:
 - Check if the packet is a valid RA or a valid RS
 - If it is an RA
 - Store information in handler 1 (AMap)
 - If it is an RS
 - Multicast RA

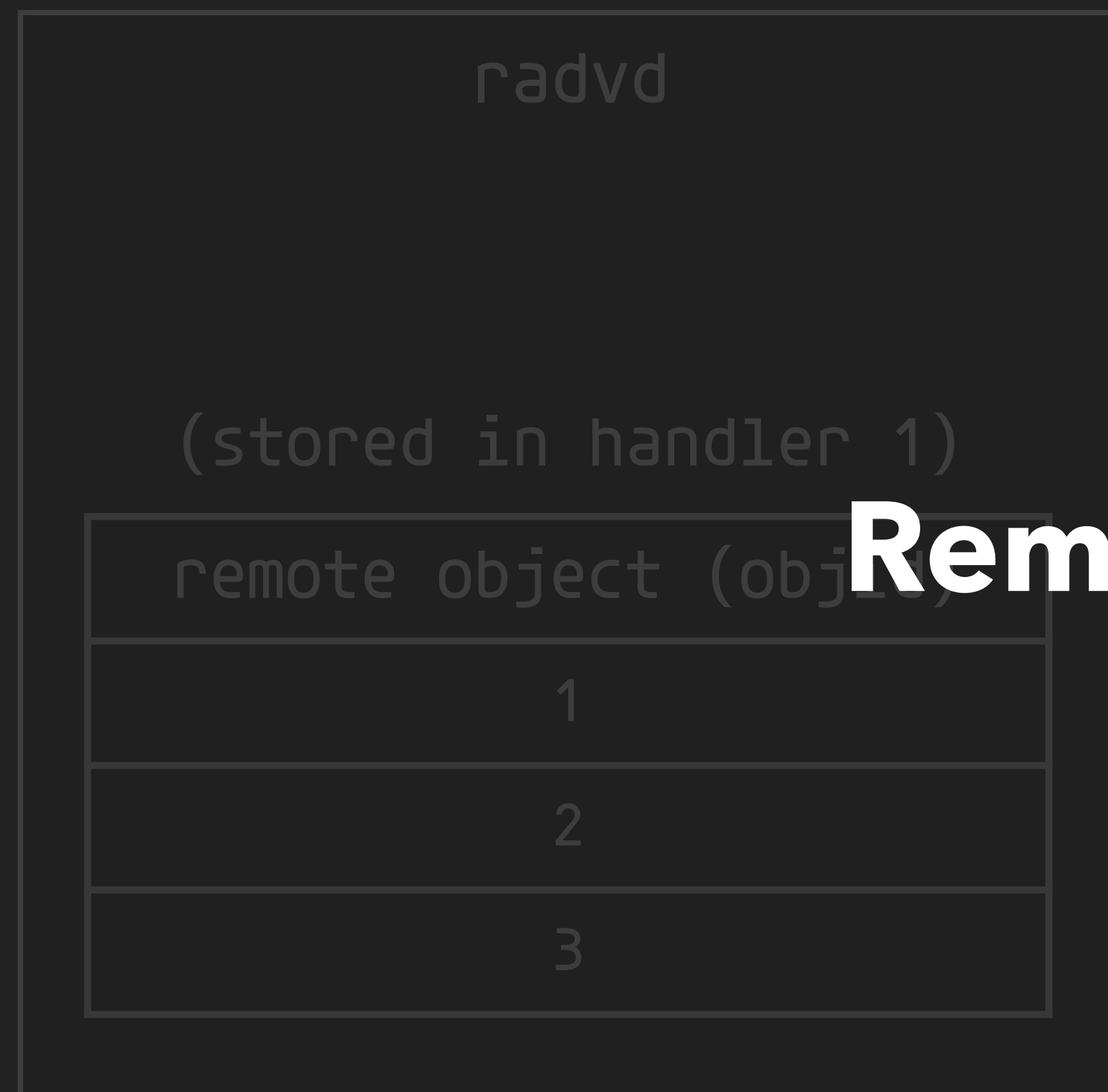


In callback function (recv RA from WAN)

```
if ( DNS_vector_now == DNS_vector_end )
{
    if ( !operator==<IPAddr6>((int *)&new_DNS_vector, (int *)&handler_1->DNS_raw) )
    {
        → clean_remoteObj(handler_1);
        vector_base::swap_raw(&new_DNS_vector, &handler_1->DNS_raw);
        v163 = (void **)handler_1->DNS_raw.end;
        for ( IPAddr6 = handler_1->DNS_raw.start; IPAddr6 != v163; IPAddr6 += 4 )
        {
            *(_DWORD *)v181 = 0;
            v183[0] = 0;
            v191[0] = 0;
            v191[1] = 0;
            → v165 = (nv::RemoteObject *)nv::roDNS((int)IPAddr6, 0, v191);
            v166 = handler_1->DNS_remoteObject.start;
            v167 = v165;
            v168 = handler_1->DNS_remoteObject.end - (void *)v166;
            if ( v168 >> 2 == sizeofAllocatedMem(v166) >> 2 )
```

```
for obj in DNS_remoteObject:
    obj.remote_remove()
DNS_remoteObject = list(map(nv::roDNS, new_DNS_vector))
```

Remote Object



radvd

(stored in handler 1)

remote object (objid)
1
2
3

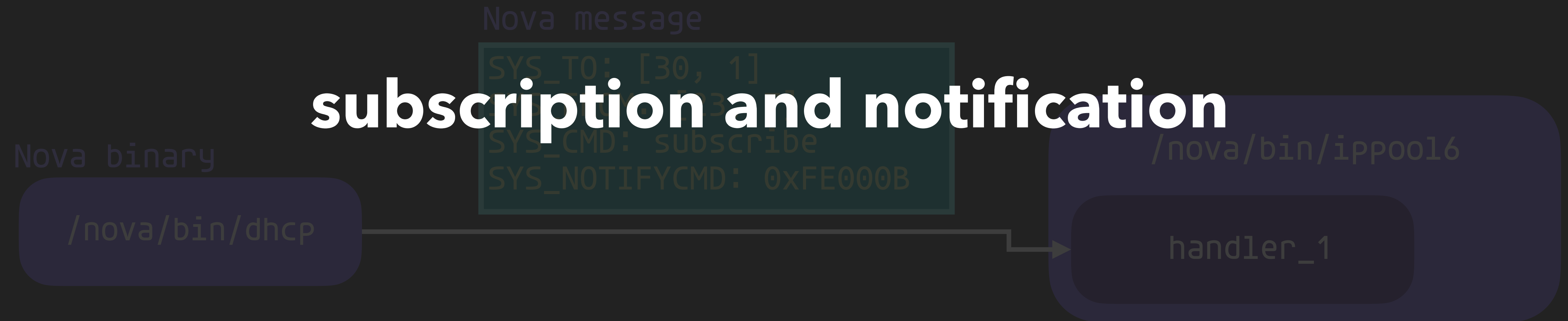
resolver

(stored in handler 2)

remote object (object)
221e:1:::1
221e:1:::2
221e:1:::3

Subscription

subscription and notification



Subscription

Nova binary

/nova/bin/dhcp

handler_7

Nova message

```
SYS_TO: [30, 1]  
SYS_FROM: [23, 7]  
SYS_CMD: subscribe  
SYS_NOTIFYCMD: 0xFE000B
```

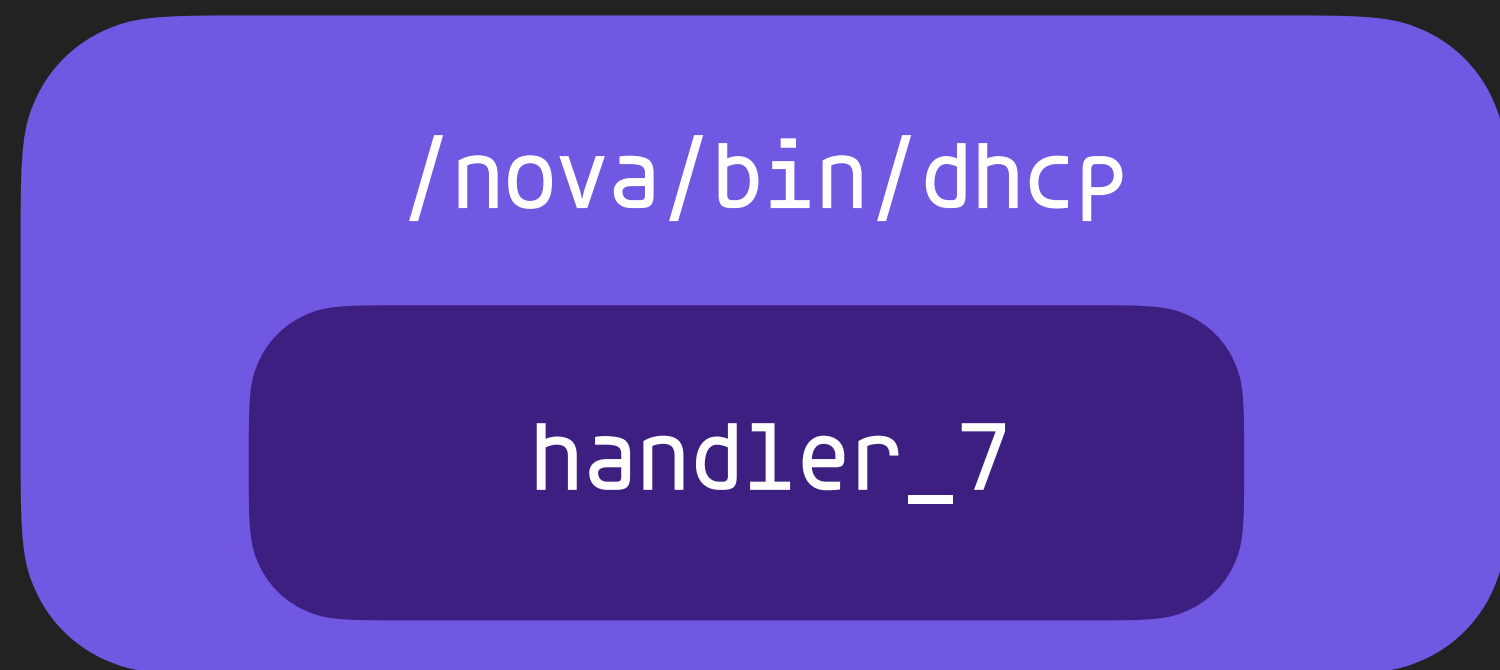
/nova/bin/ippool16

handler_1



Subscription

Nova binary



Nova message

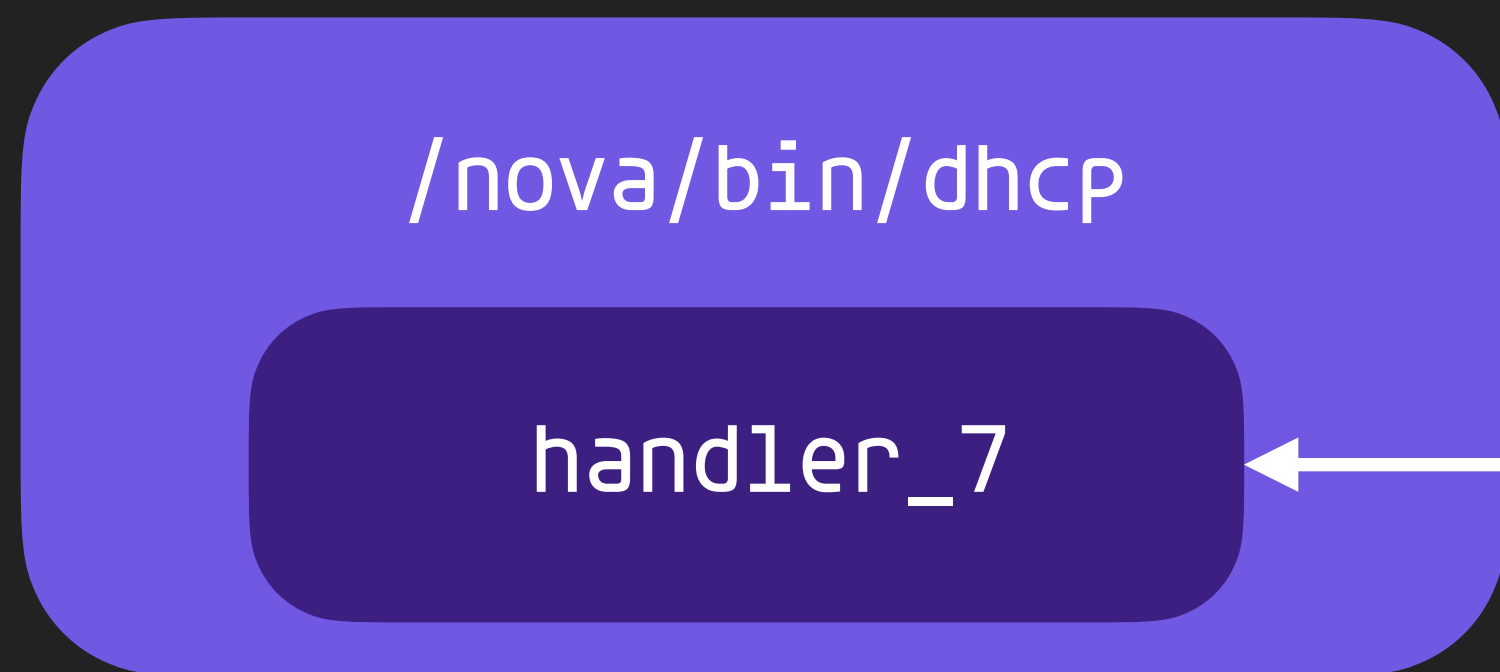
```
SYS_TO: [30, 1]  
SYS_FROM: [??]  
SYS_CMD: AddObj
```

Nova binary



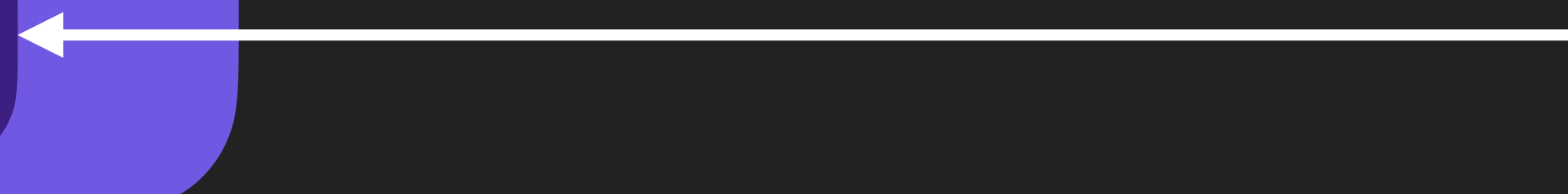
Subscription

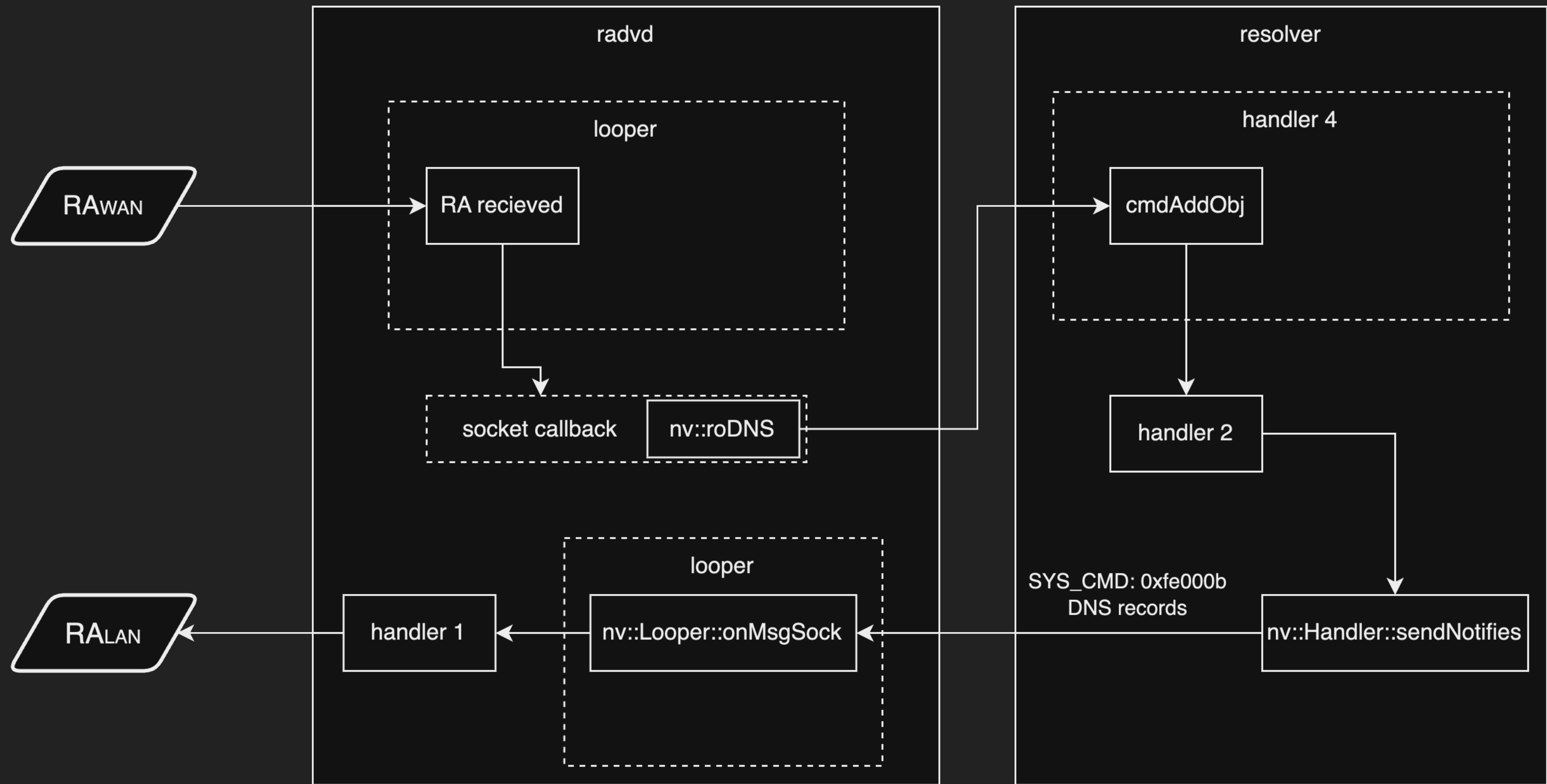
Nova binary



Nova message

```
SYS_T0: [23, 7]  
SYS_FROM: [30, 1]  
SYS_CMD: 0xFE000B  
...
```





In callback function (recv RA from WAN)

```
if ( DNS_vector_now == DNS_vector_end )
{
  if ( !operator==(IPAddr6)((int *)&new_DNS_vector, (int *)&handler_1->DNS_raw) )
  {
    clean_remoteObj(handler_1->DNS_remoteObject, new_DNS_vector, handler_1->DNS_raw);
    vector_base::swap(handler_1->DNS_remoteObject, new_DNS_vector);
    v163 = (void **)handler_1->DNS_raw.end;
    for ( IPAddr6 = handler_1->DNS_raw.start; IPAddr6 != v163; IPAddr6 += 4 )
    {
      *(_DWORD *)v183[0] = 0;
      v191[0] = 0;
      v191[1] = 0;
      → v165 = (nv::RemoteObject *)nv::roDNS((int)IPAddr6, 0, v191);
      v166 = handler_1->DNS_remoteObject.start;
      v167 = v165;
      v168 = handler_1->DNS_remoteObject.end - (void *)v166;
      if ( v168 >> 2 == sizeofAllocatedMem(v166) >> 2 )
    }
  }
}
```

nv::Handler::postMessage (Non-blocking)

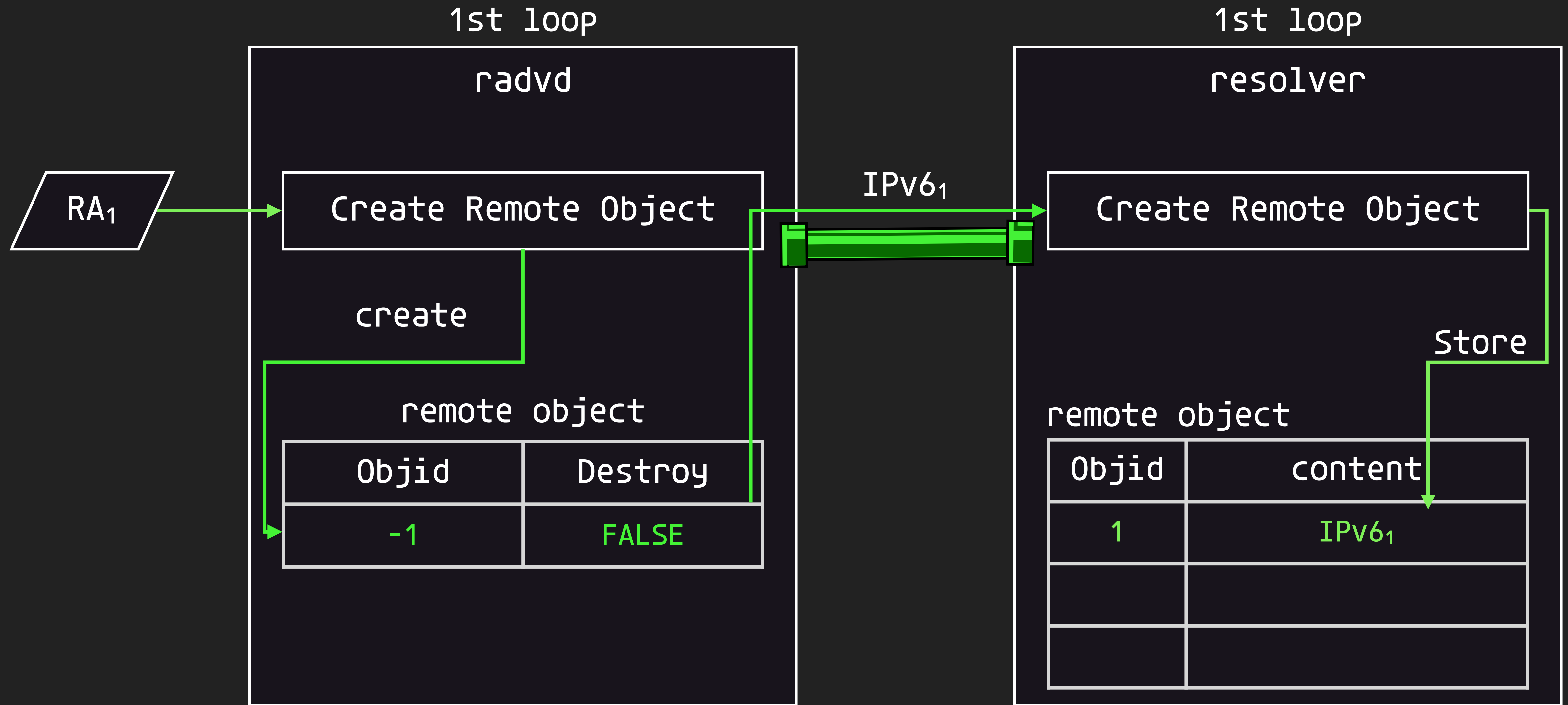
nv::RemoteObjectBackend::request

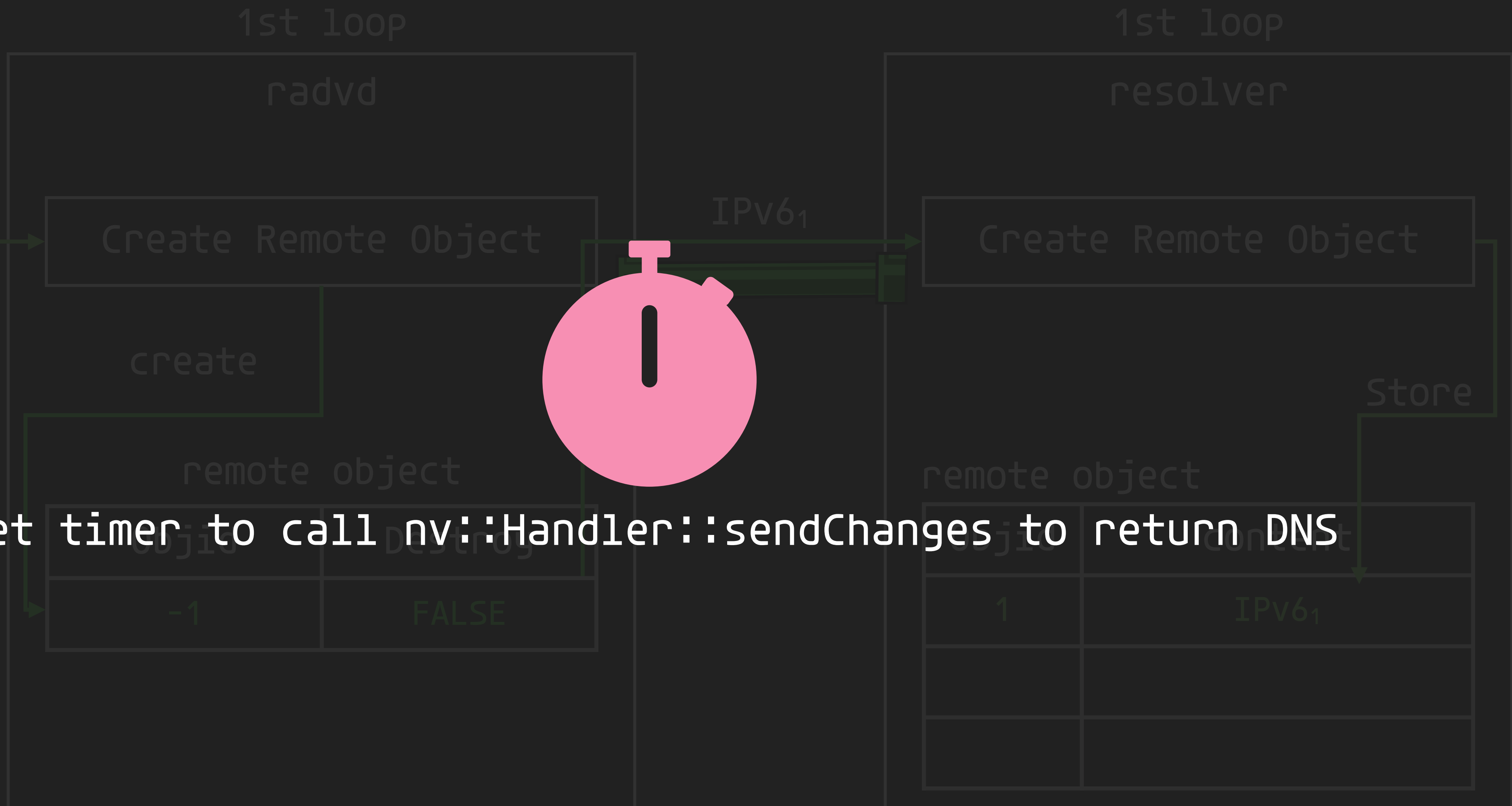
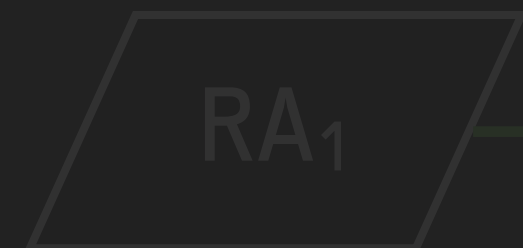
```
for obj in DNS_remoteObject:
  obj.remote_remove()
DNS_remoteObject = list(map(nv::roDNS, new_DNS_vector))
```

In callback function (recv RA from WAN)

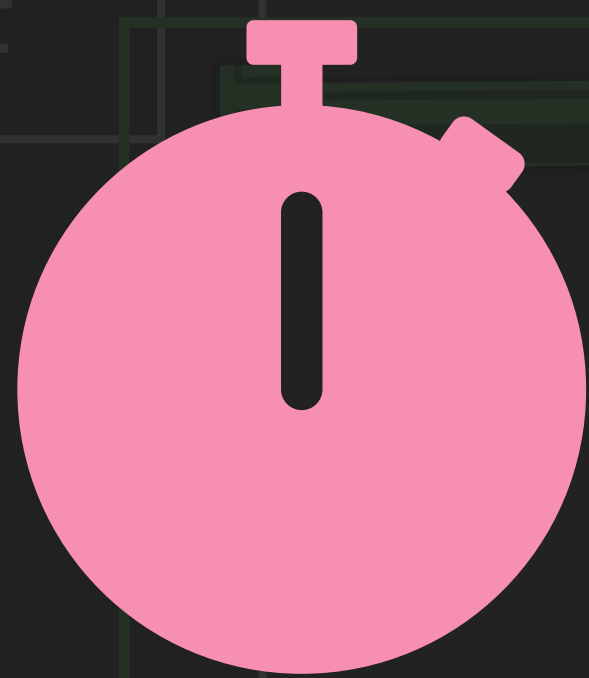
```
if ( DNS_vector_now == DNS_vector_end )
{
  if ( !operator==(IPAddr6)((int *)&new_DNS_vector, (int *)&handler_1->DNS_raw) )
  {
    → clean_remoteObj(handler_1); → nv::RemoteObject::~~RemoteObject
    vector_base::swap_raw(&new_DNS_vector, &handler_1->DNS_raw);
    v163 = (void **)handler_1->DNS_raw.end;
    for ( IPAddr6 = handler_1->DNS_raw.start; IPAddr6 != v163; IPAddr6 += 4 )
    {
      *(_DWORD *)v181 void __fastcall nv::RemoteObjectBackend::cleanup(nv::RemoteObjectBackend *this)
      v183[0] = 0; {
      v191[0] = 0;   this->destroy = 1;
      v191[1] = 0;   if ( this->objid != -1 )
      v165 = (nv::Rem  nv::RemoteObjectBackend::postRemove(this); (Non-blocking)
      v166 = handler_ }
      v167 = v165;
      v168 = handler_1->DNS_remoteObject.end - (void *)v166;
      if ( v168 >> 2 == sizeofAllocatedMem(v166) >> 2 )
    }
  }
}
```

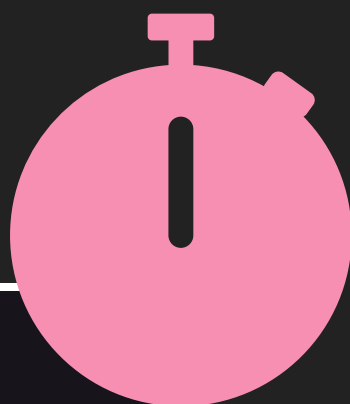
```
for obj in DNS_remoteObject:
  obj.remote_remove()
DNS_remoteObject = list(map(nv::roDNS, new_DNS_vector))
```





Set timer to call `nv::Handler::sendChanges` to return DNS





1st loop

1st loop

radvd

resolver

RA₁

Create Remote Object

Create Remote Object

IPV6₁

ResponseHandler

Store

update
(Not yet)

remote object

remote object

Objid	Destroy
-1	FALSE

Objid	content
1	IPV6 ₁

Response
(Not yet)



2nd loop

radvd

RA₂

Delete Remote Object

ignore

set

remote object

Objid	Destroy
-1	TRUE

1st loop

resolver

Create Remote Object

Store

remote object

Objid	content
1	IPv6 ₁



2nd loop

radvd

RA₂

Delete Remote Object

Create Remote Object

remote object

Objid	Destroy
-1	TRUE



1st loop

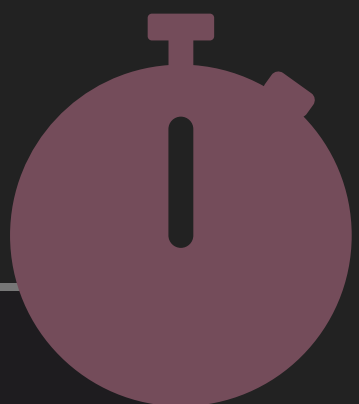
resolver

Create Remote Object

Store

remote object

Objid	content
1	IPv6 ₁



2nd loop

radvd

RA₂

Delete Remote Object

Create Remote Object

remote object

Objid	Destroy
-1	TRUE
-1	FALSE

create

IPv6₂

1st loop

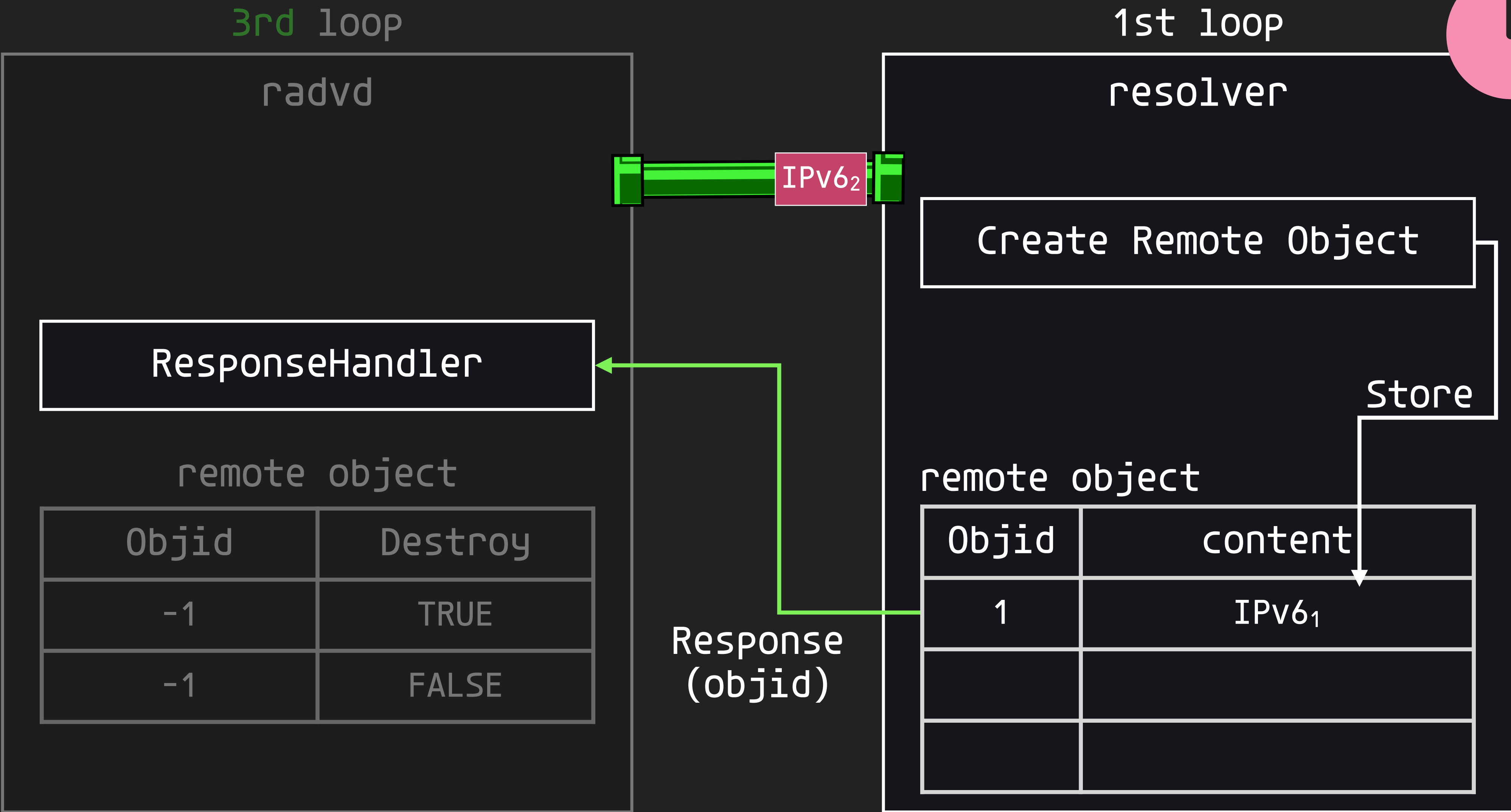
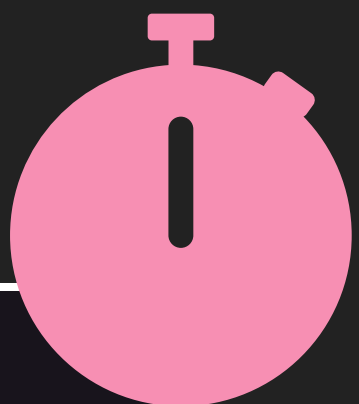
resolver

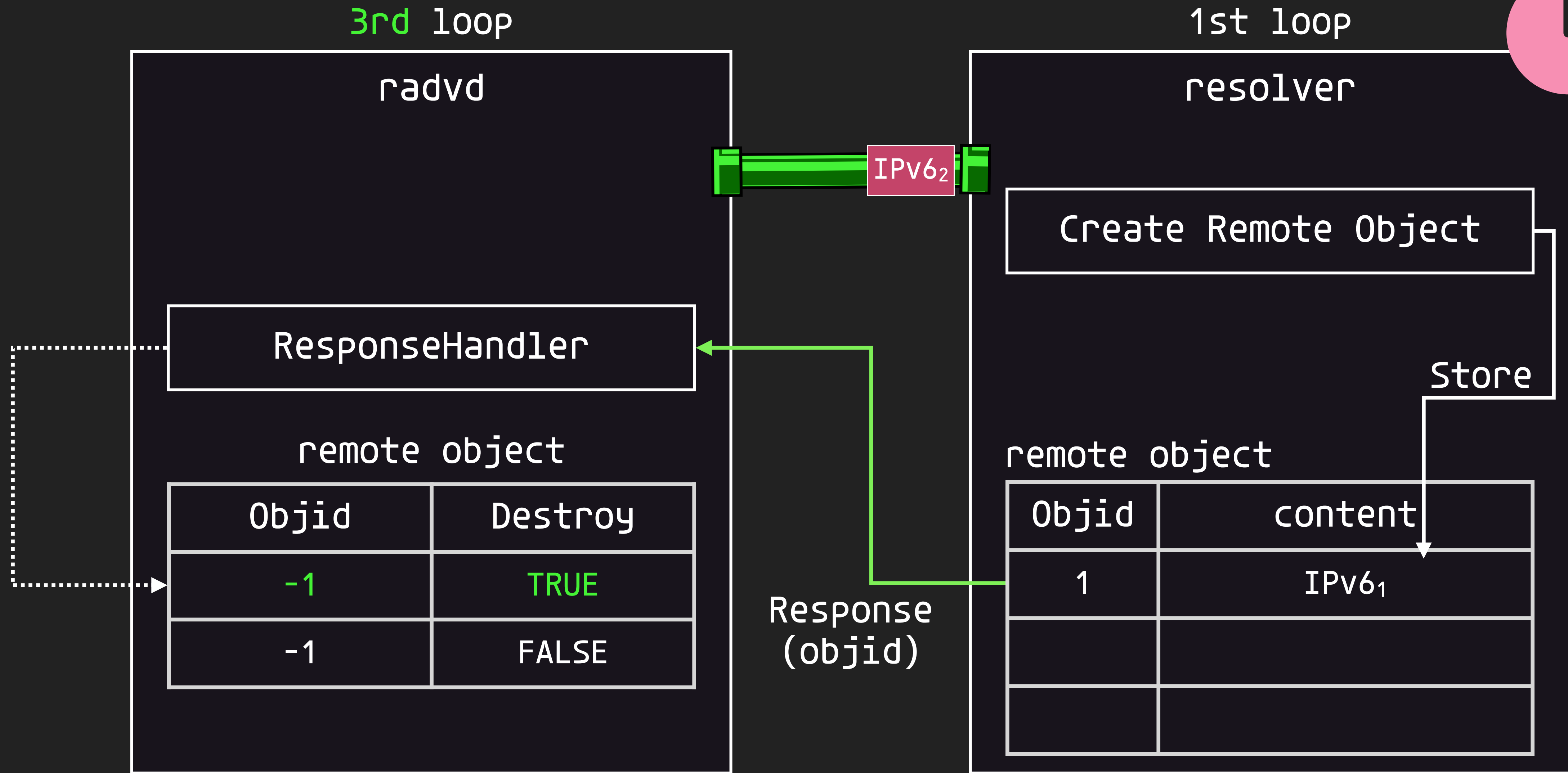
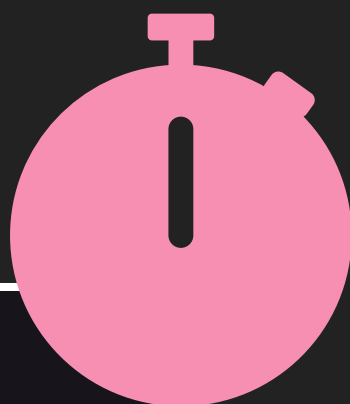
Create Remote Object

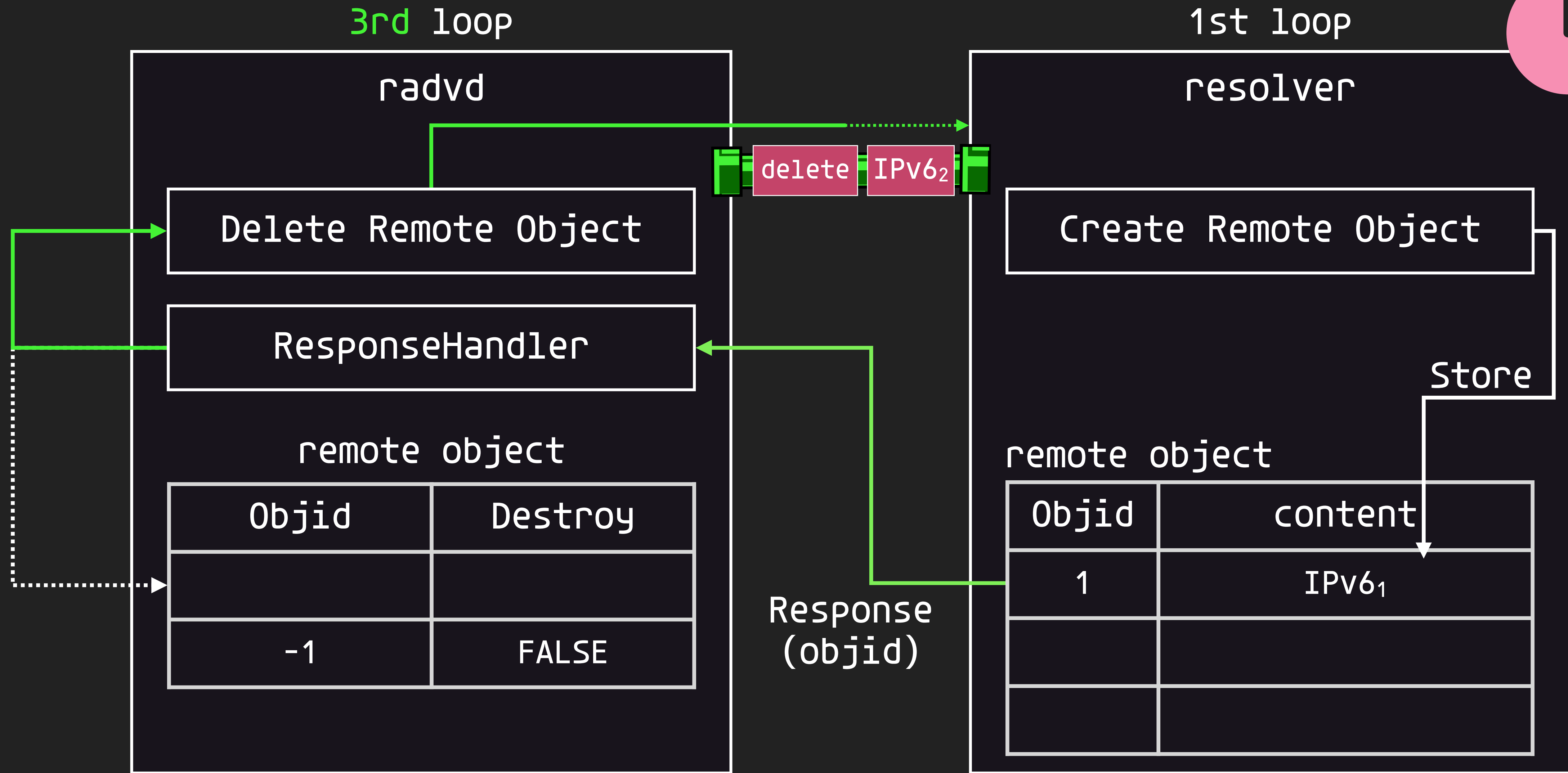
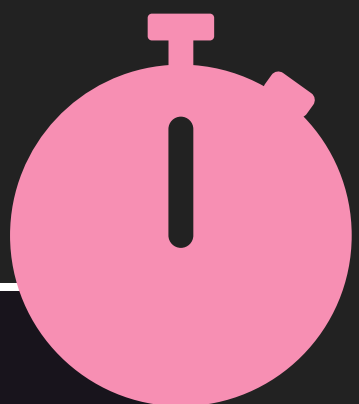
Store

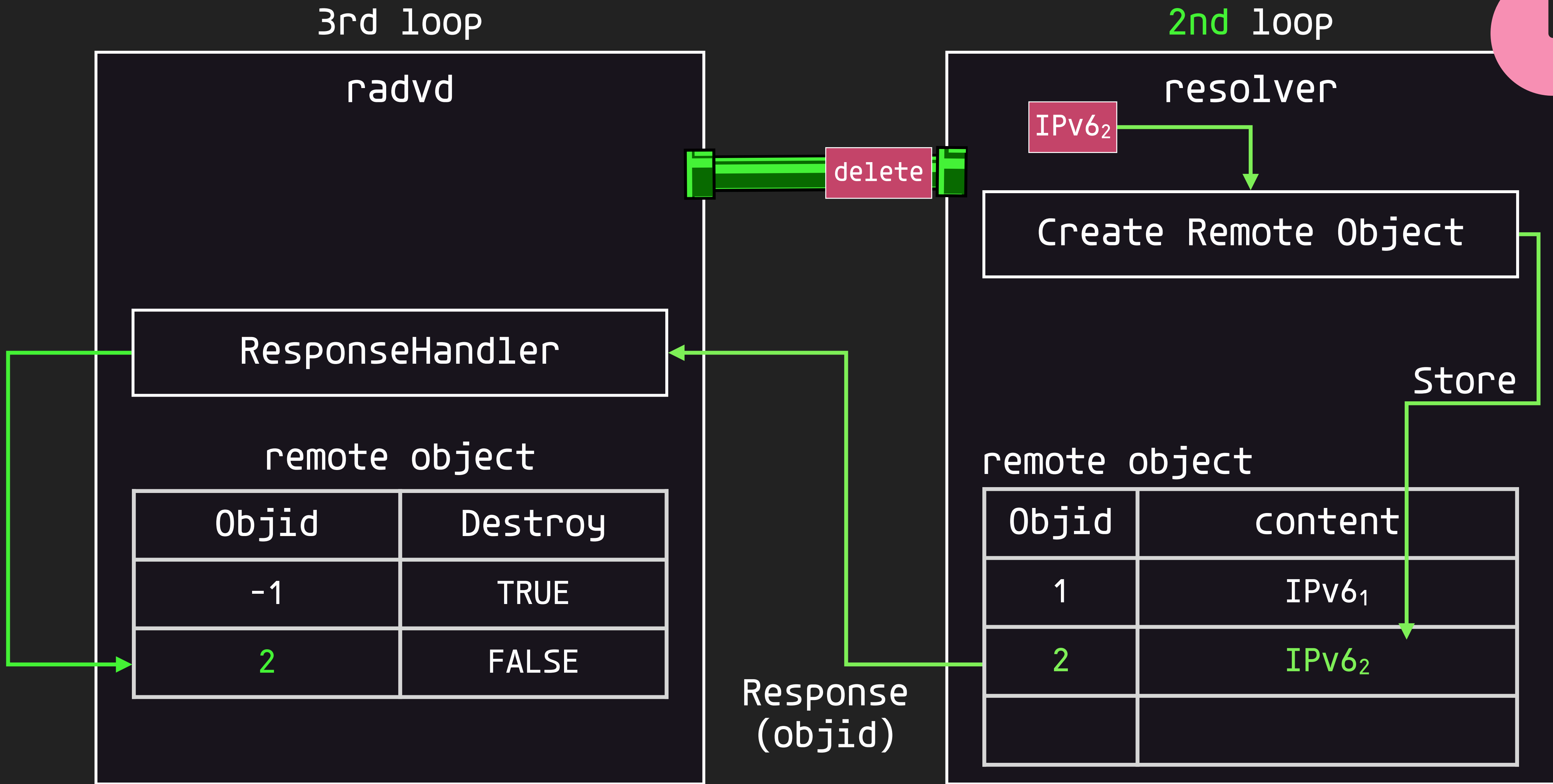
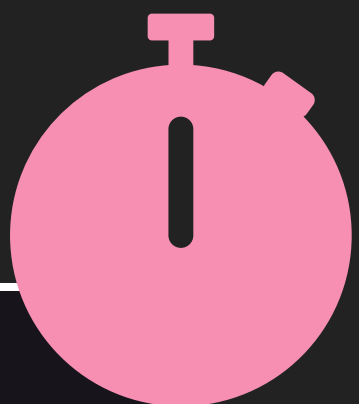
remote object

Objid	content
1	IPv6 ₁



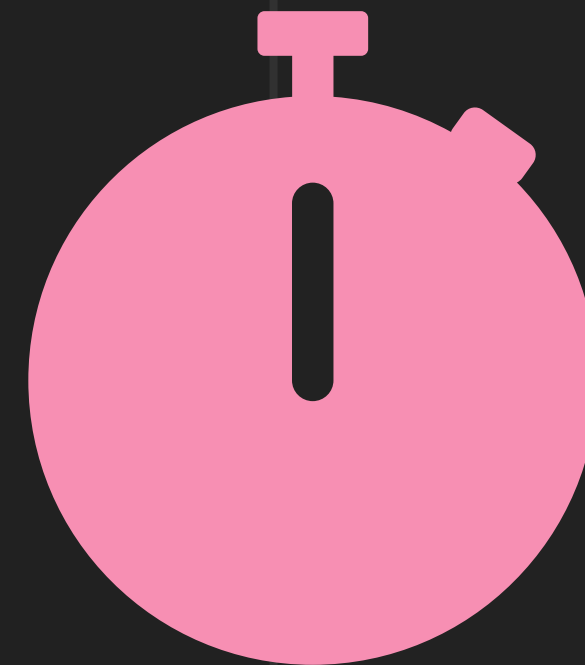






3rd loop

radvd



delete

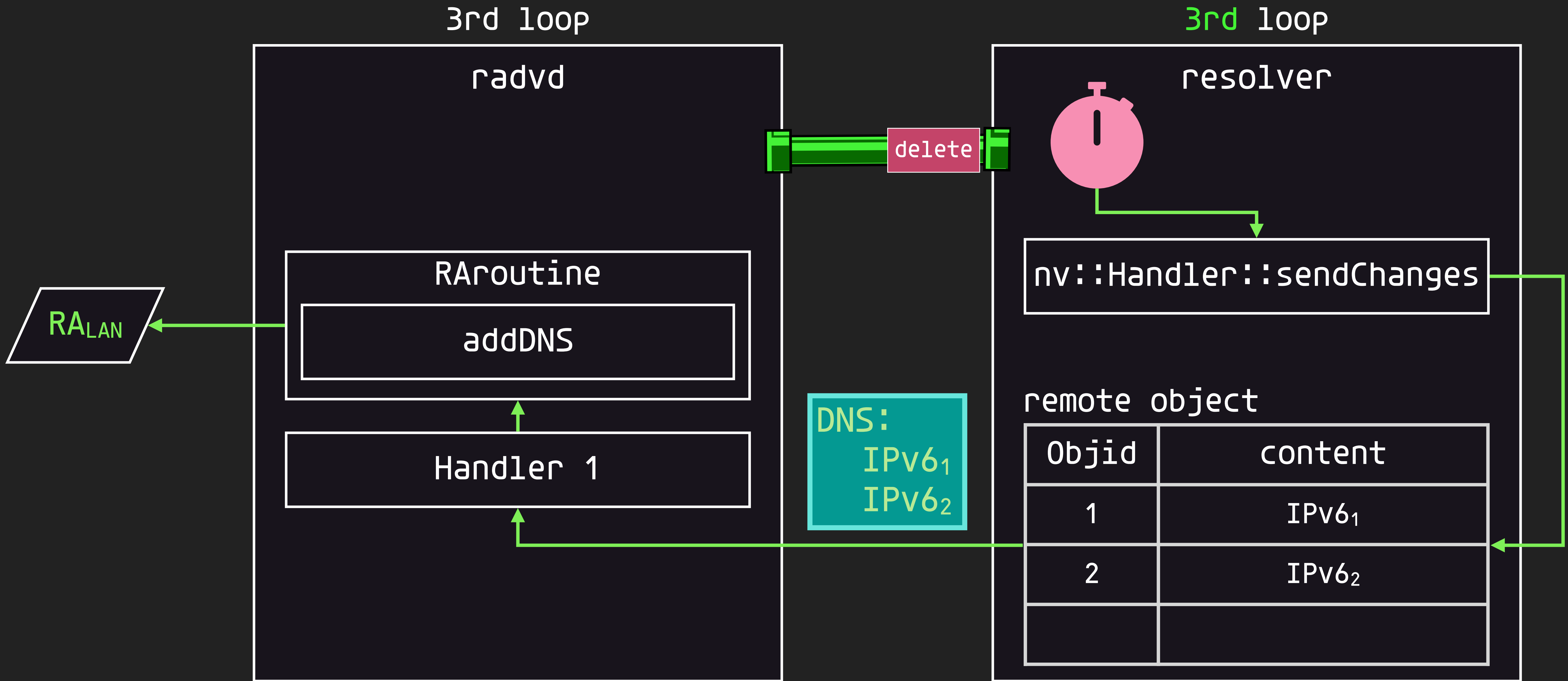
3rd loop

resolver



Create Remote Object

call `nv::Handler::sendChanges` to return DNS



Race Condition

- Pattern
 - Use non-blocking methods to create/delete the remote object
 - Subscribe to the remote object
- Impact:
 - Maybe it can be used to bypass some checks

Race Condition

- Pattern

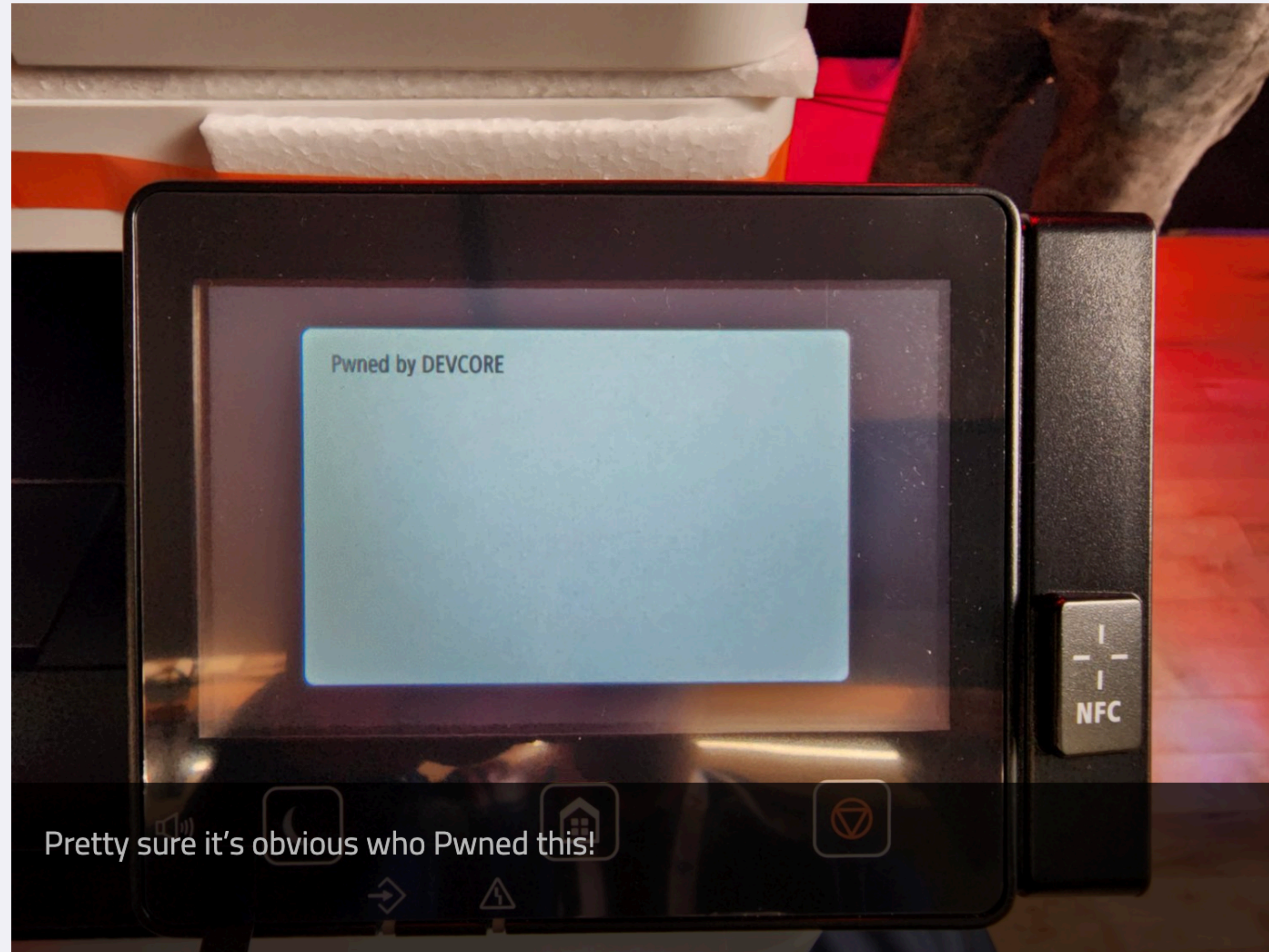
- Use non-blocking methods to create remote object
- Subscribe to the

- Impact:

- Maybe can be used in checks



SUCCESS - DEVCORE becomes the first team ever to successfully execute two different Stack-based buffer overflow attacks against a Mikrotik router and a Canon printer in the brand new SOHO SMASHUP category. They earn a cool \$100K cash and 10 Master of Pwn points.



Pretty sure it's obvious who Pwned this!

Summary

- MikroTik reimplements everything with its own designed IPC.
 - The business logic is scattered all over.
- A pre-auth RCE on WAN has existed for nine years.
- Race condition in remote objects due to non-blocking methods.
- The tools to ease reversing will be available at:
 - <https://github.com/terrynini/routeros-tools>

Q&A

DEV✓**CORE**