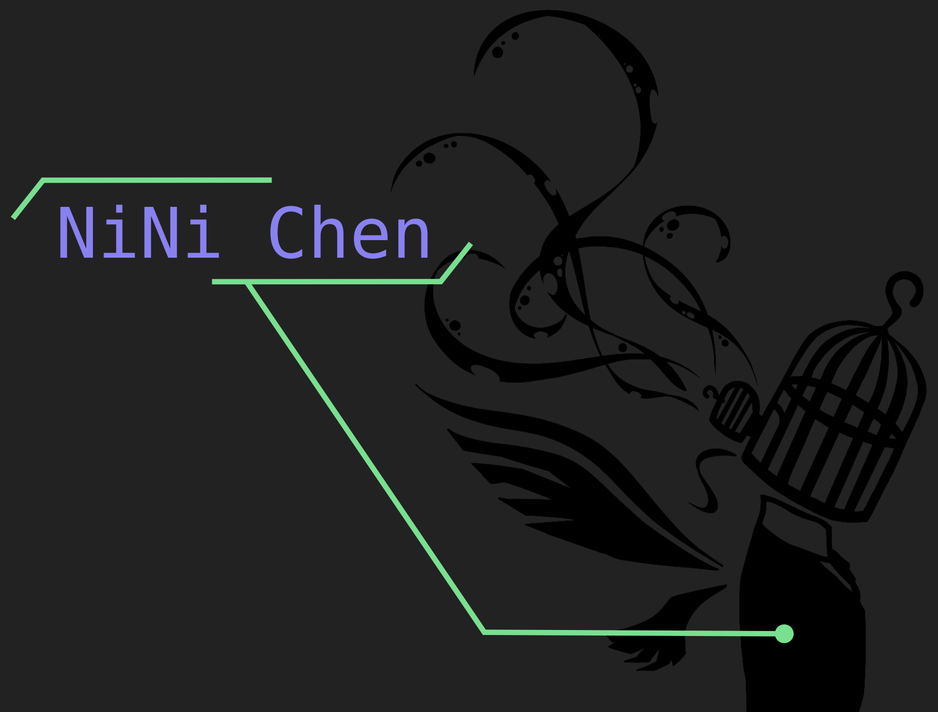
A Comprehensive Review on the Less-Traveled Road

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

DEVCORE

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

DEVCORE

Why?

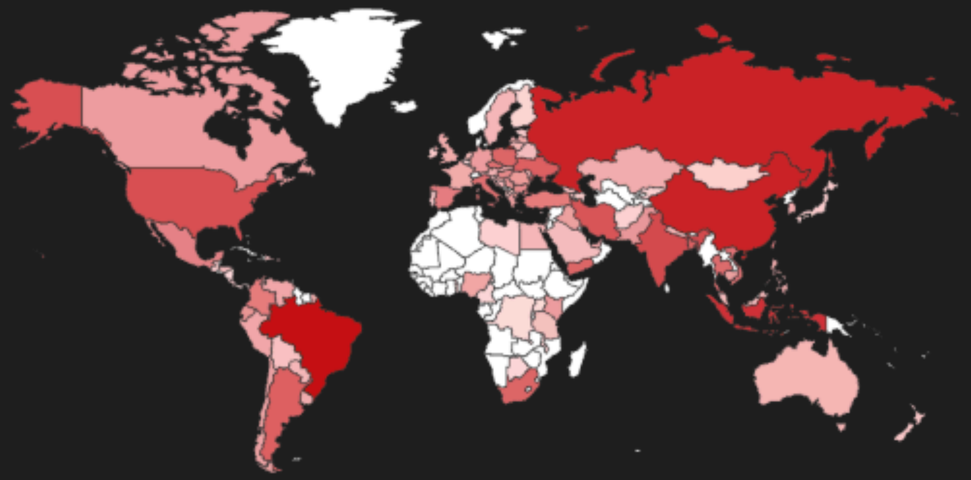
Target		Point
Initial Stage	Final Stage	
TP-Link AX1800 WiFi 6 Router	Meta Portal Go	
NETGEAR Nighthawk WiFi6 Router	Amazon Echo Show 15	
Synology RT6600ax	Google Nest Max	
Cisco Integrated Service Router C921-4P	Sonos One Speaker	
Mikrotik RouterBoard RB2011UiAS-IN	Apple HomePod mini	
Ubiquiti Networks EdgeRouter X SFP	Amazon Echo Studio	10
	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

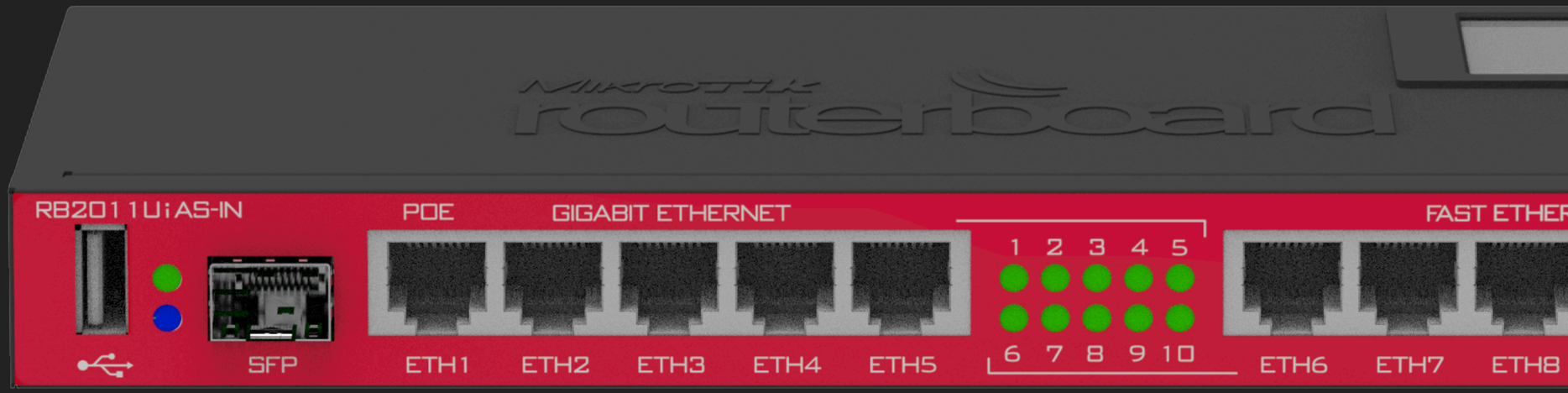


DEVCORE

2023/07/03 - https://www.shodan.io/search?query=MikroTik

RouterOS

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



DEVCORE

RouterOS

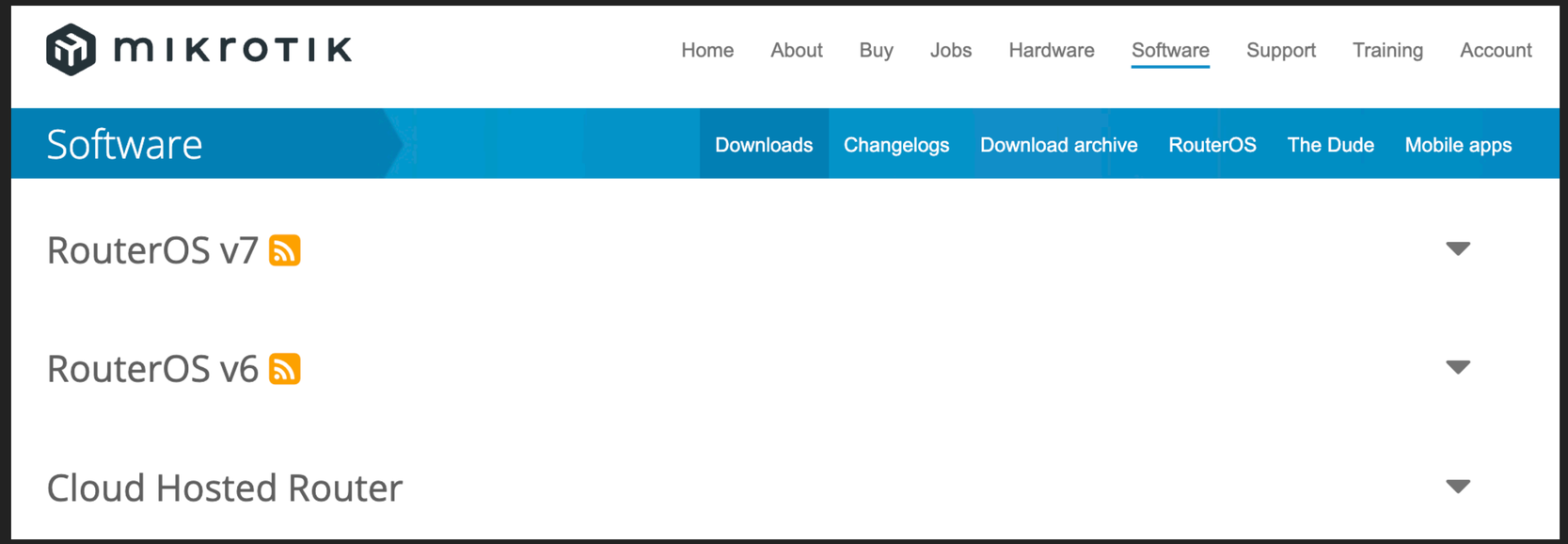
- A stand-alone operating system based on the Linux kernel
- Also available for virtual machines to turn a PC into a router
- Closed source and also a closed ecosystem.
  (It is said that you can get GPL sources used in RouterOS if you ask them)

> To get a CD with the corresponding source code for the GPL-covered programs in this distribution, wire transfer $45 to MikroTikls SIA, Ūnijas iela 2, Riga, LV-1039, Latvia. Please contact MikroTikls 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 MikroTikls proprietary SOFTWARE.

https://mikrotik.com/downloadterms.html

RouterOS

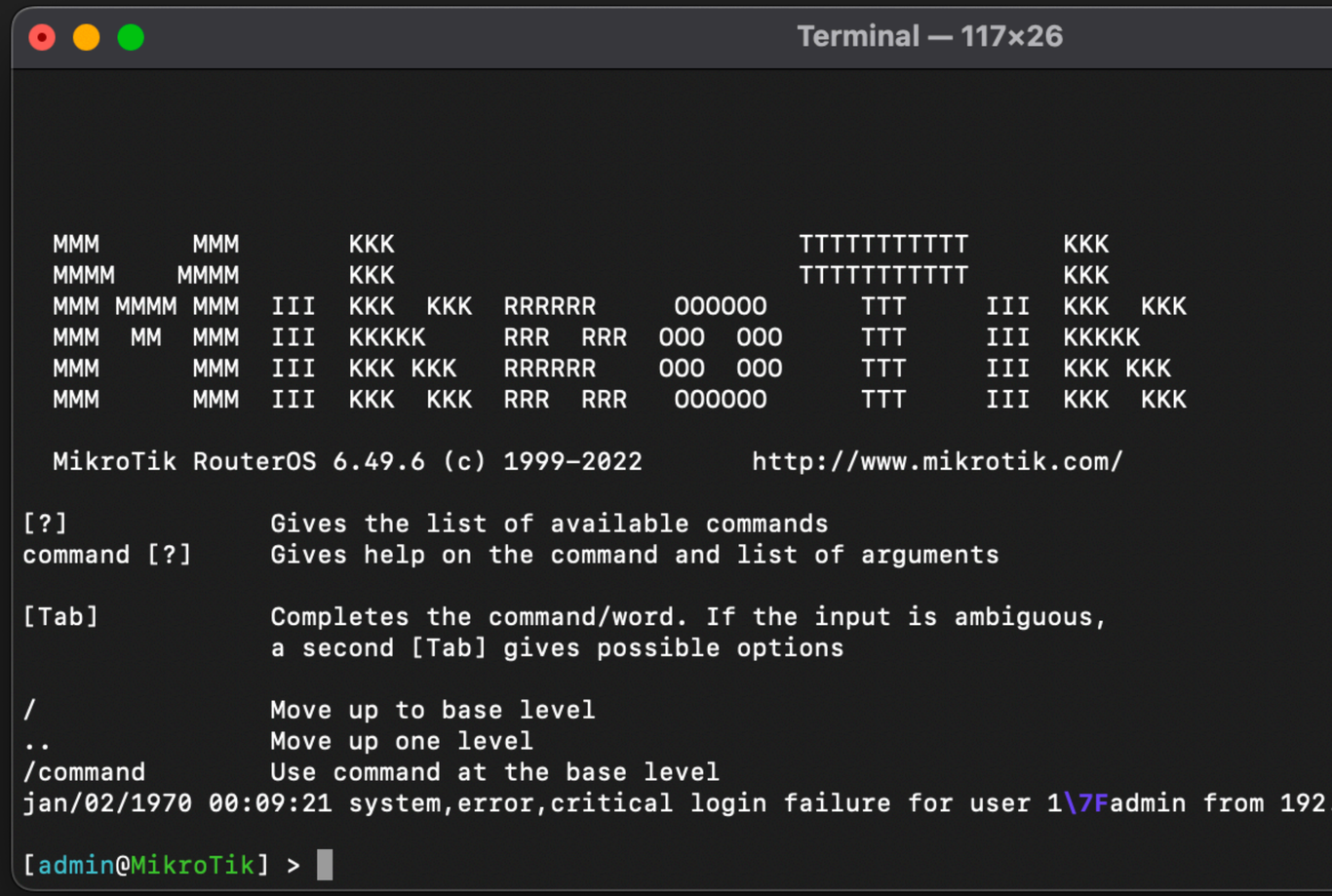
- RouterOS v6 and RouterOS v7 can be considered two different branches
- CHR is the RouterOS image for VM, x86_64 only



DEVCORE

RouterOS

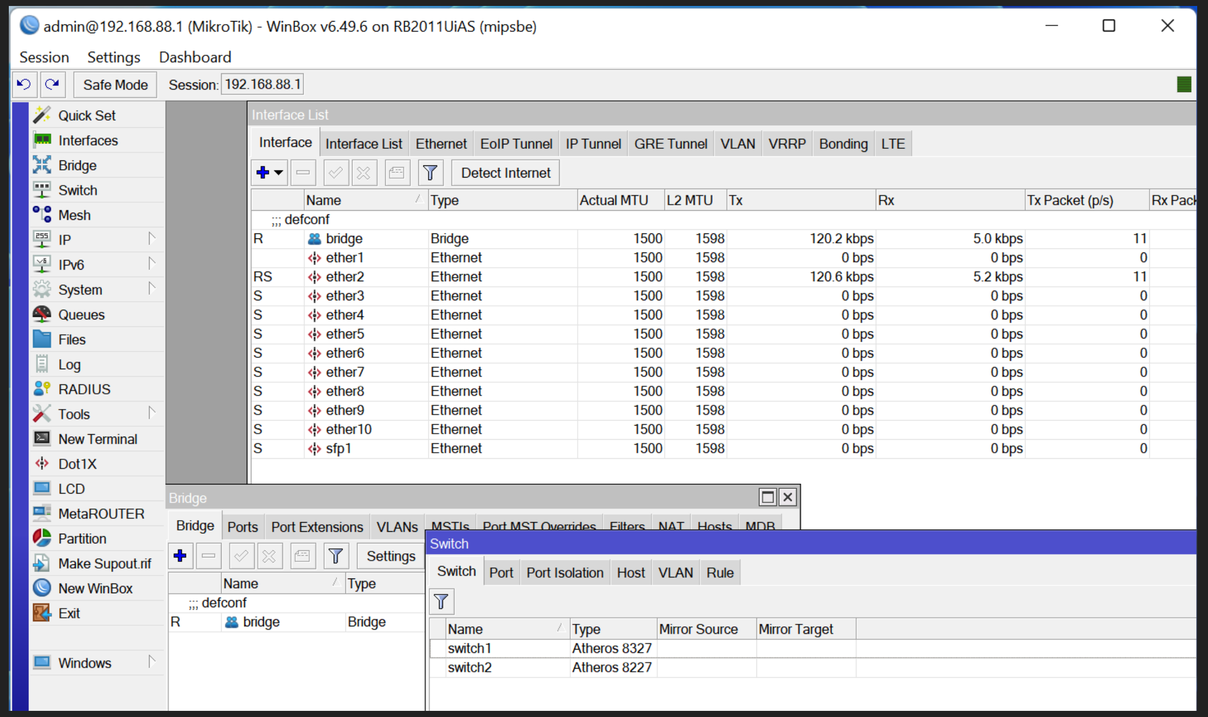
- Most binaries are “Nova binary”
- No official method to access the linux system.



DEVCORE

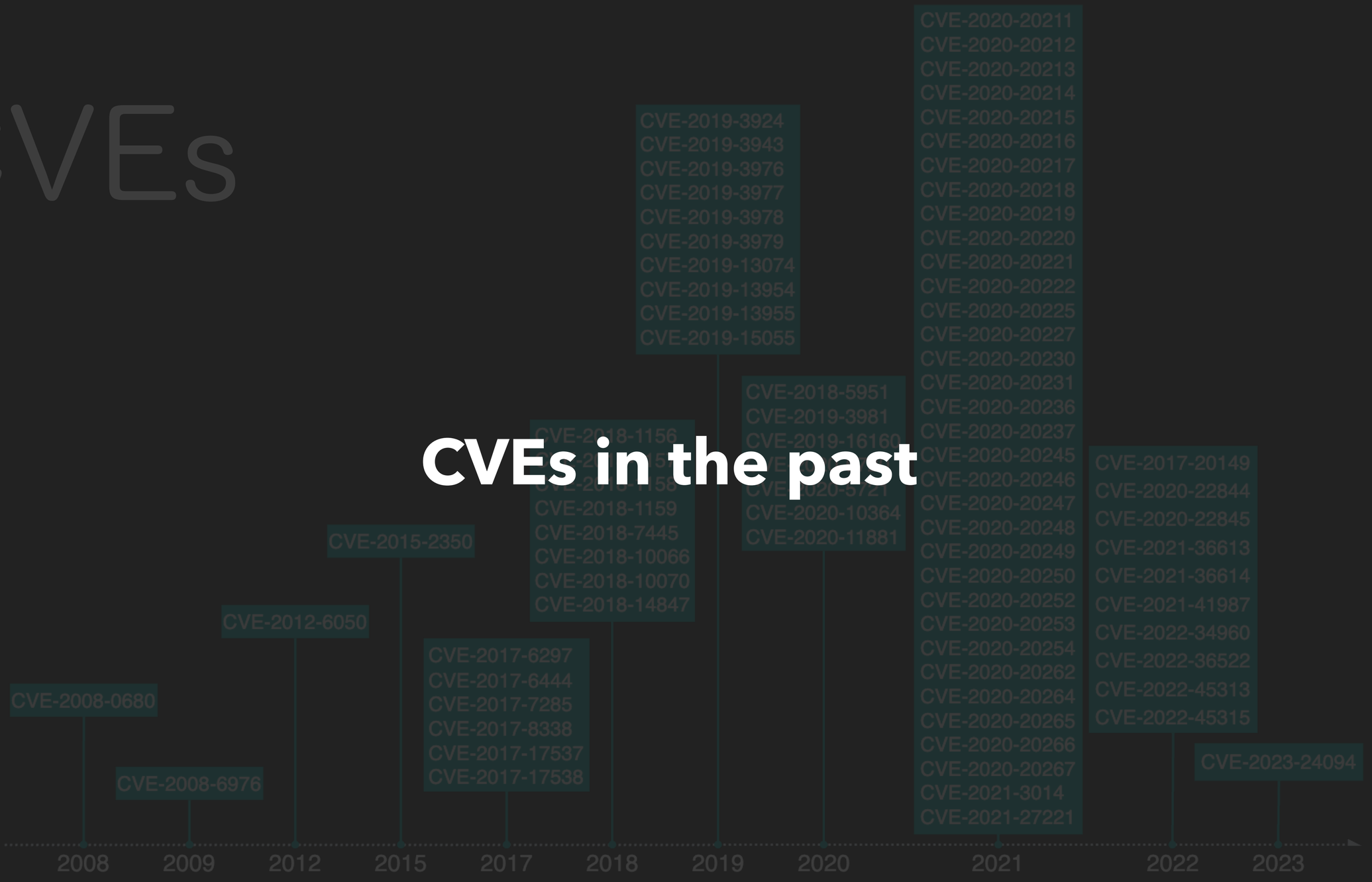
Winbox

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



DEVCORE

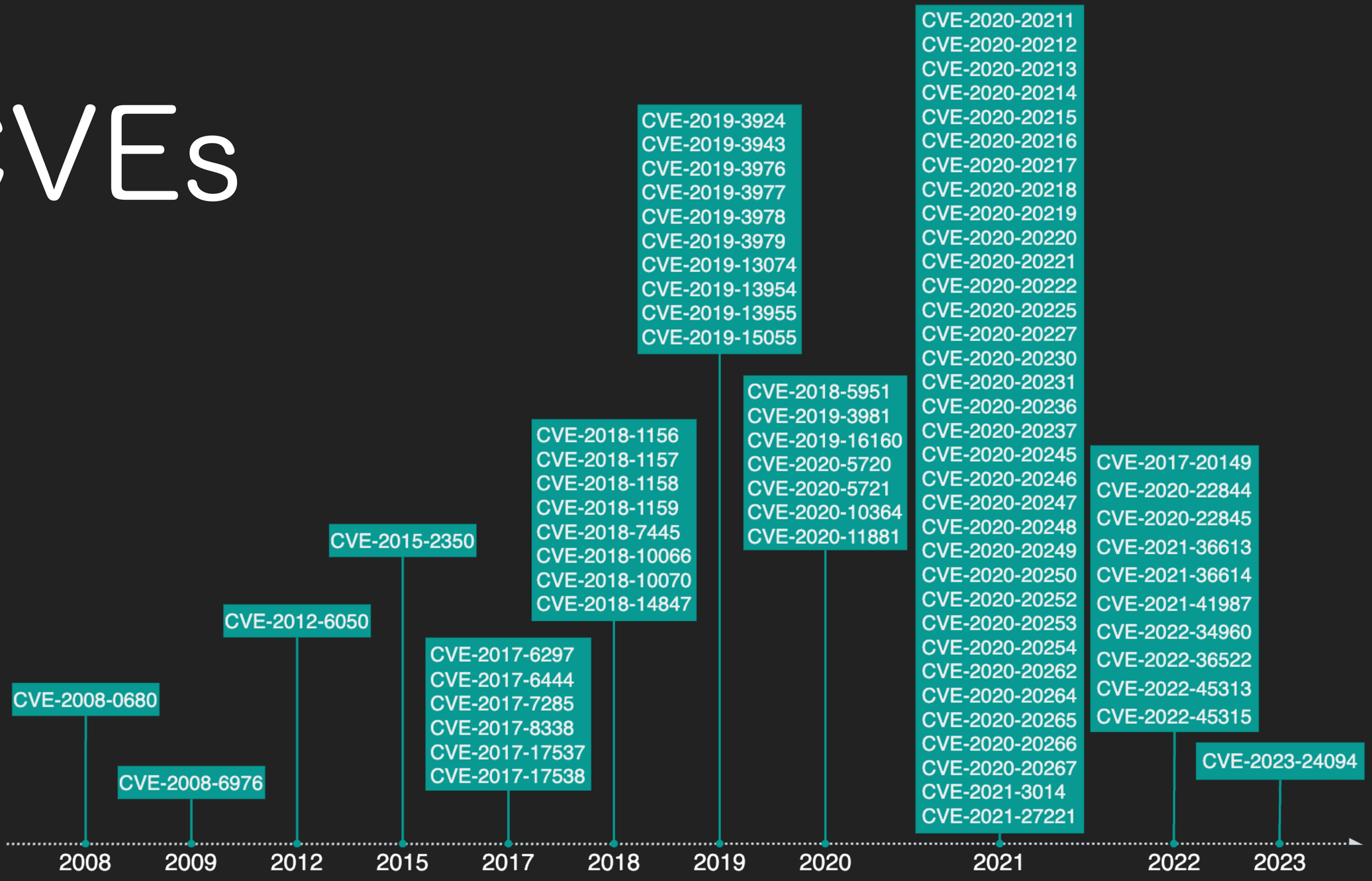
CVEs



CVEs in the past

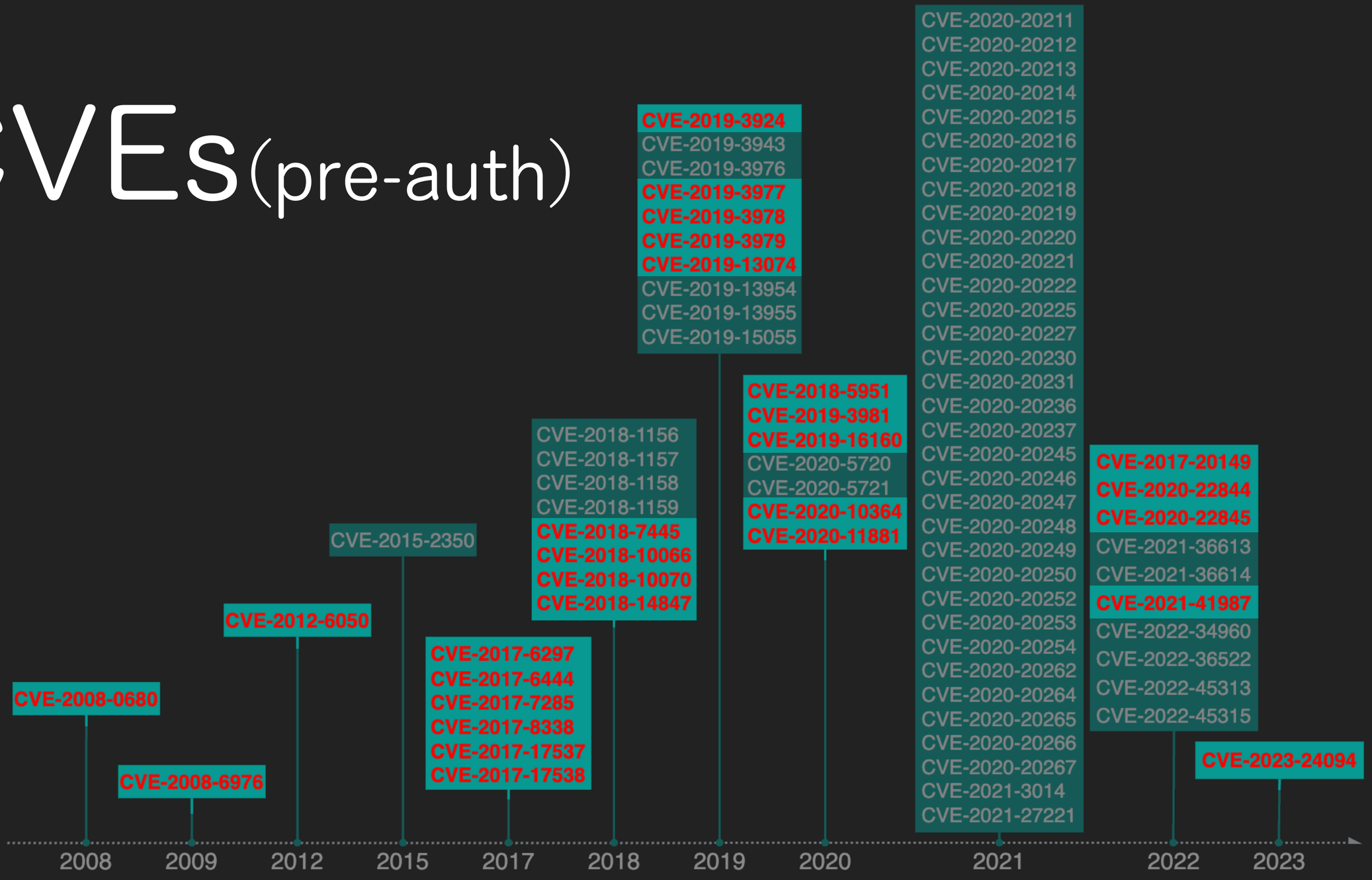
DEVCORE

CVEs



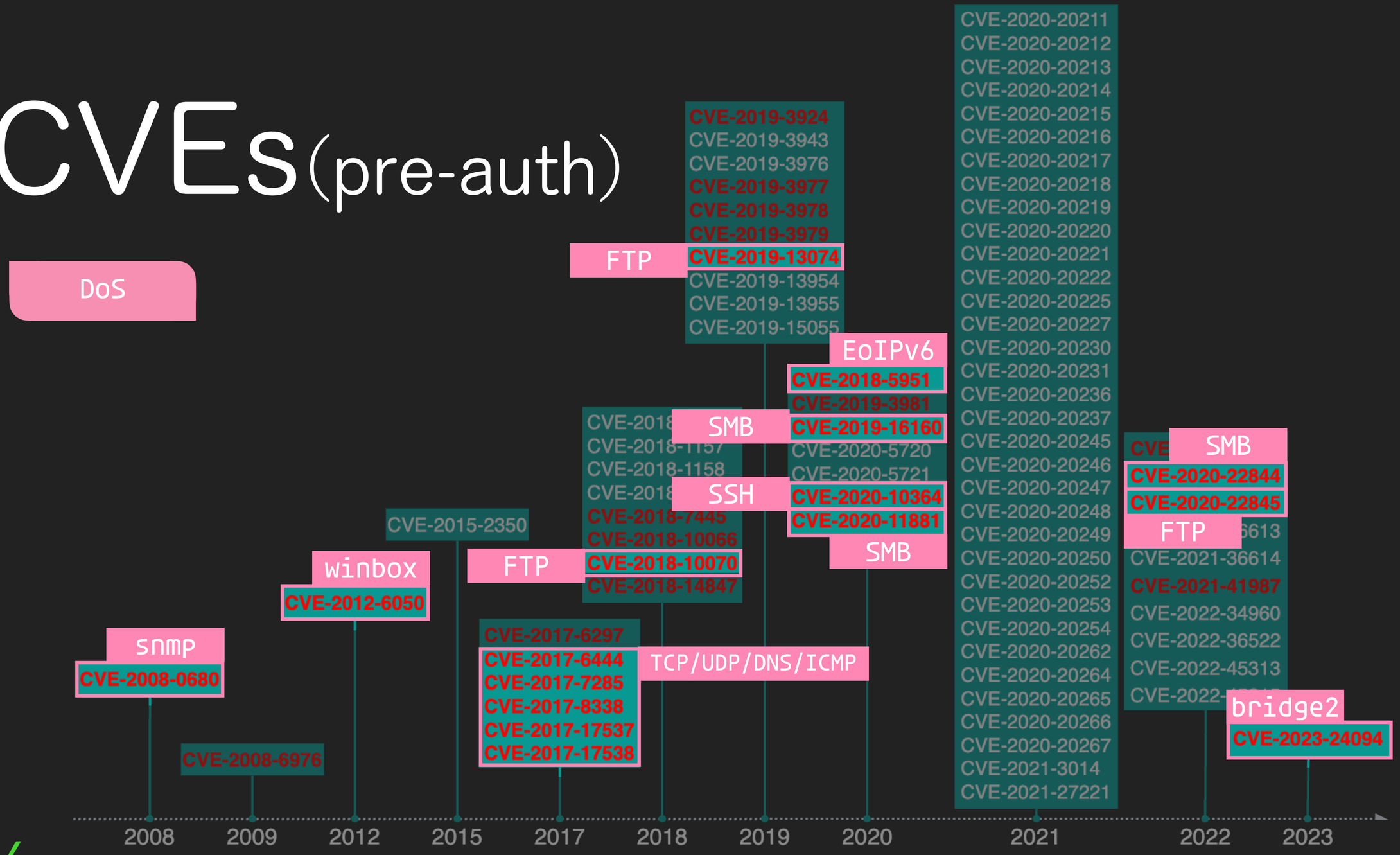
DEVCORE

CVEs (pre-auth)



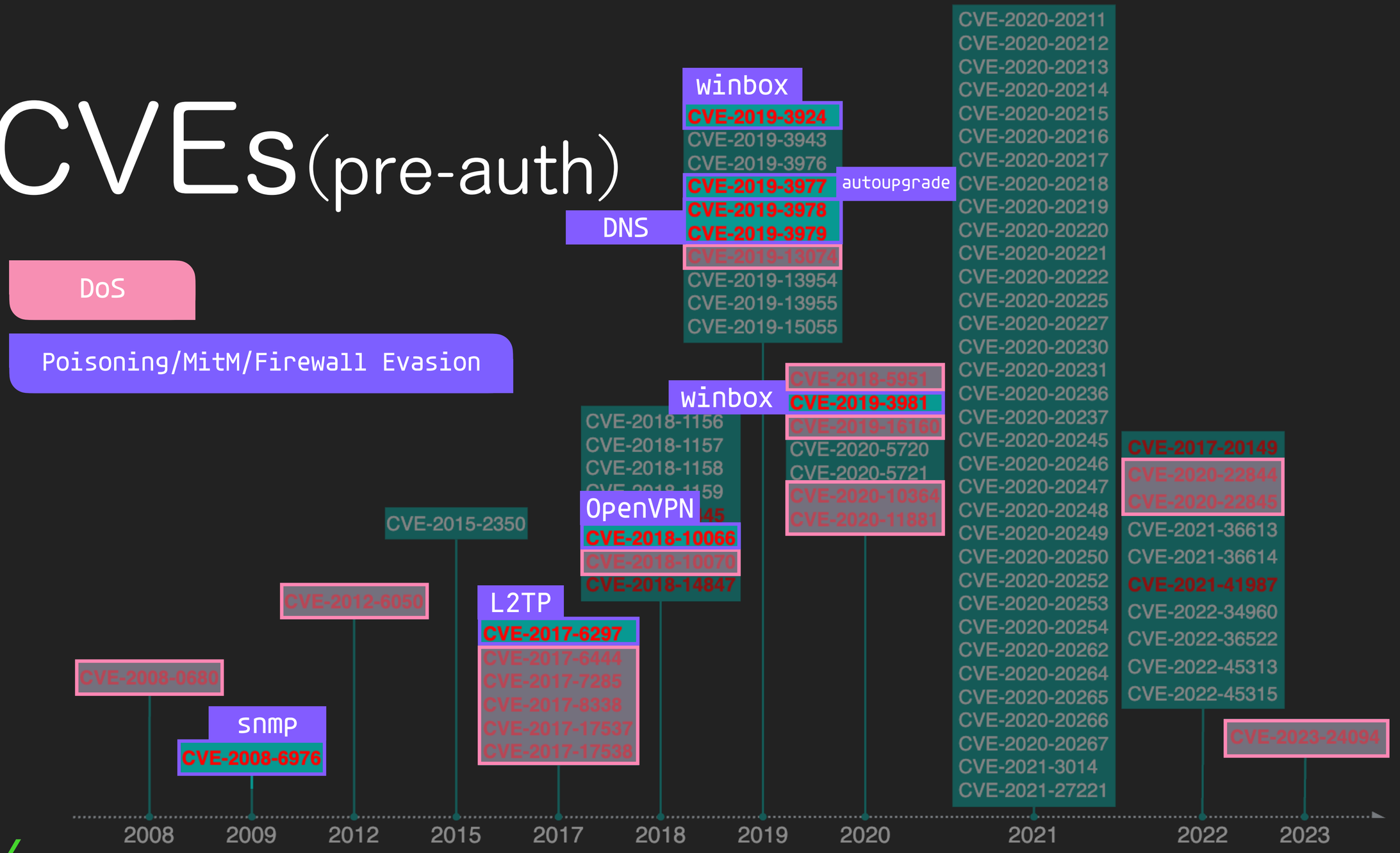
DEVCORE

CVEs(pre-auth)



DEVCORE

CVEs(pre-auth)

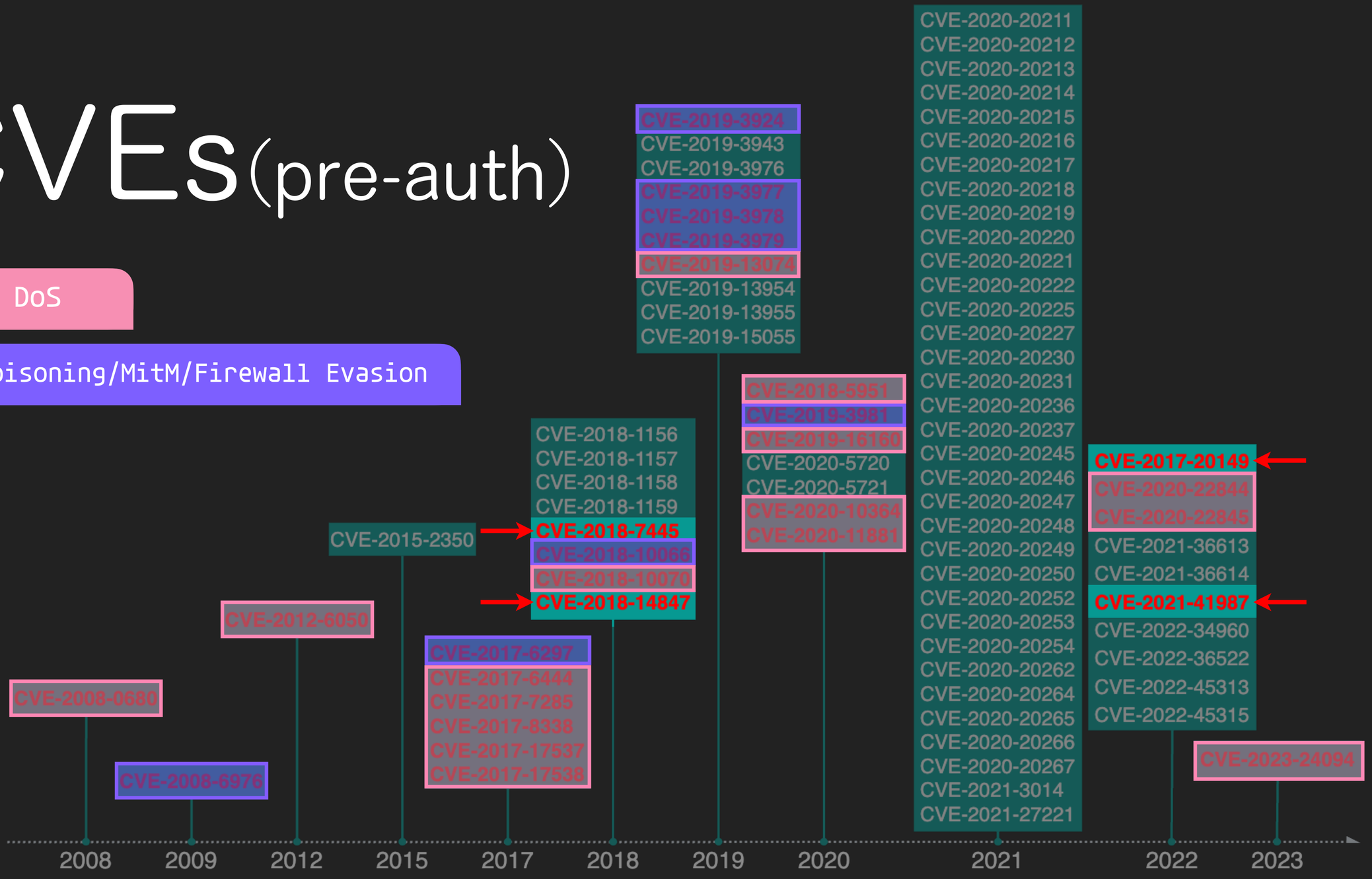


DEVCORE

CVEs(pre-auth)

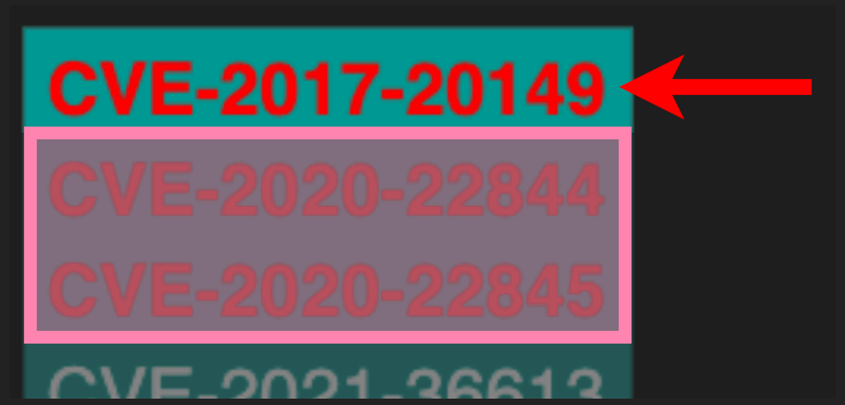
DoS

Poisoning/MitM/Firewall Evasion



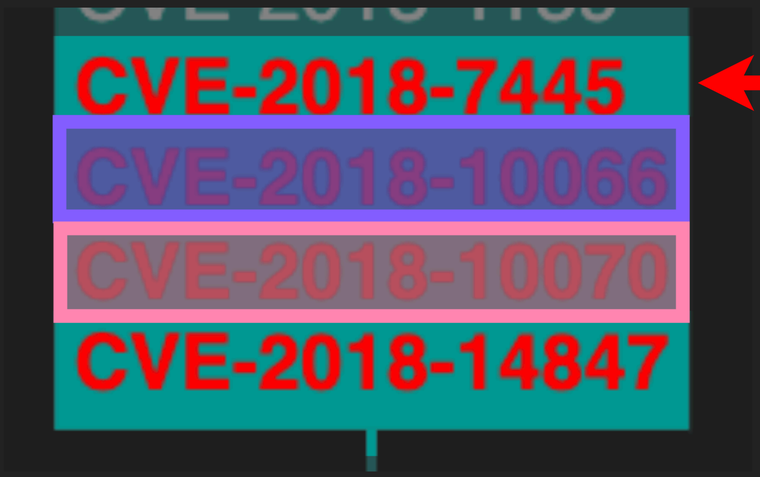
DEVCORE

CVEs(pre-auth)



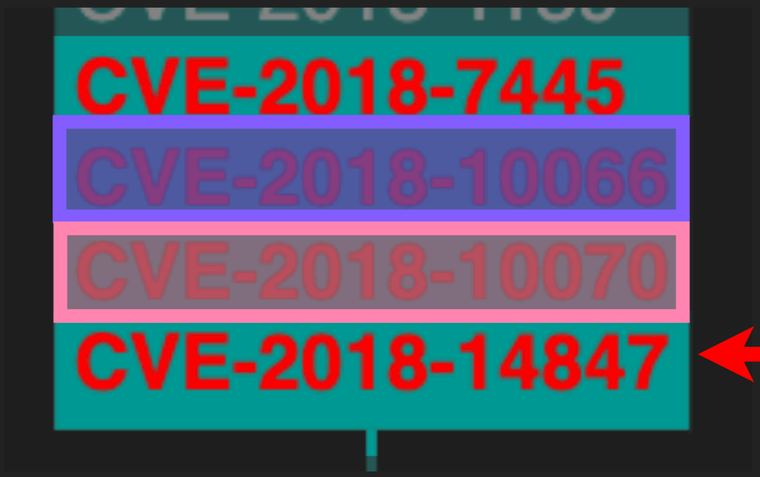
- aka Chimay-Red
- One of the exploits leaked from the CIA
- Lacking input validation on Content-Length in HTTP request cause the stack clash attack.

CVEs(pre-auth)



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

CVEs(pre-auth)



- 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

CVE-2018-14847

$password_{db} = password_{raw}$ xor md5(username + "283i4jfkai3389")

- Allows unauthenticated attackers to read arbitrary files on RouterOS

CVEs(pre-auth)

CVE-2021-36614
CVE-2021-41987
CVE-2022-34960

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

DEVCORE

2017

Kirils

Rooting the MikroTik routers

focus on jailbreak

Kirils

A deeper journey into MikroTik routers

focus on jailbreak

Kirils

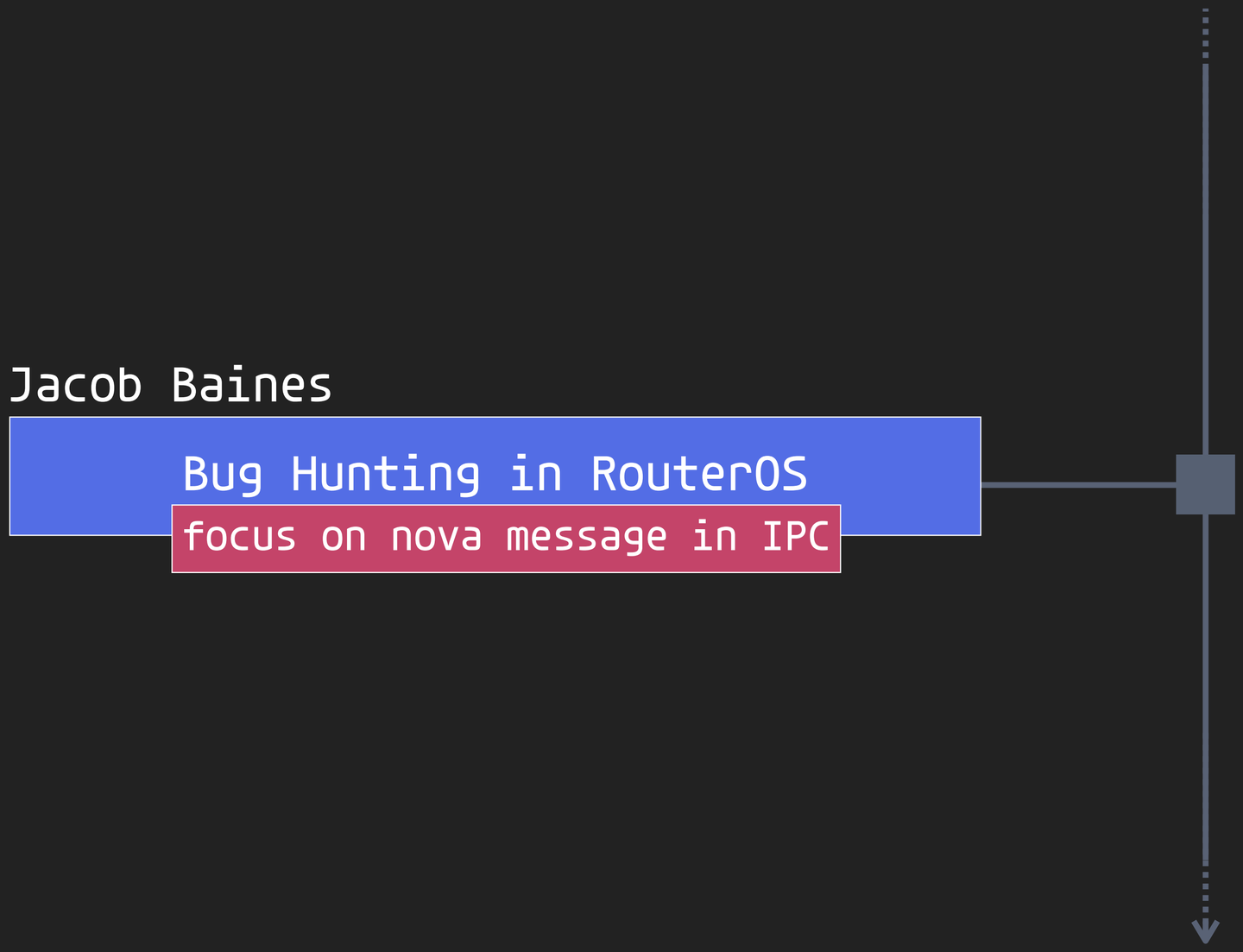
Tools for effortless reverse engineering of MikroTik router

focus on jailbreak



DEVCORE

2018



DEVCORE

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)

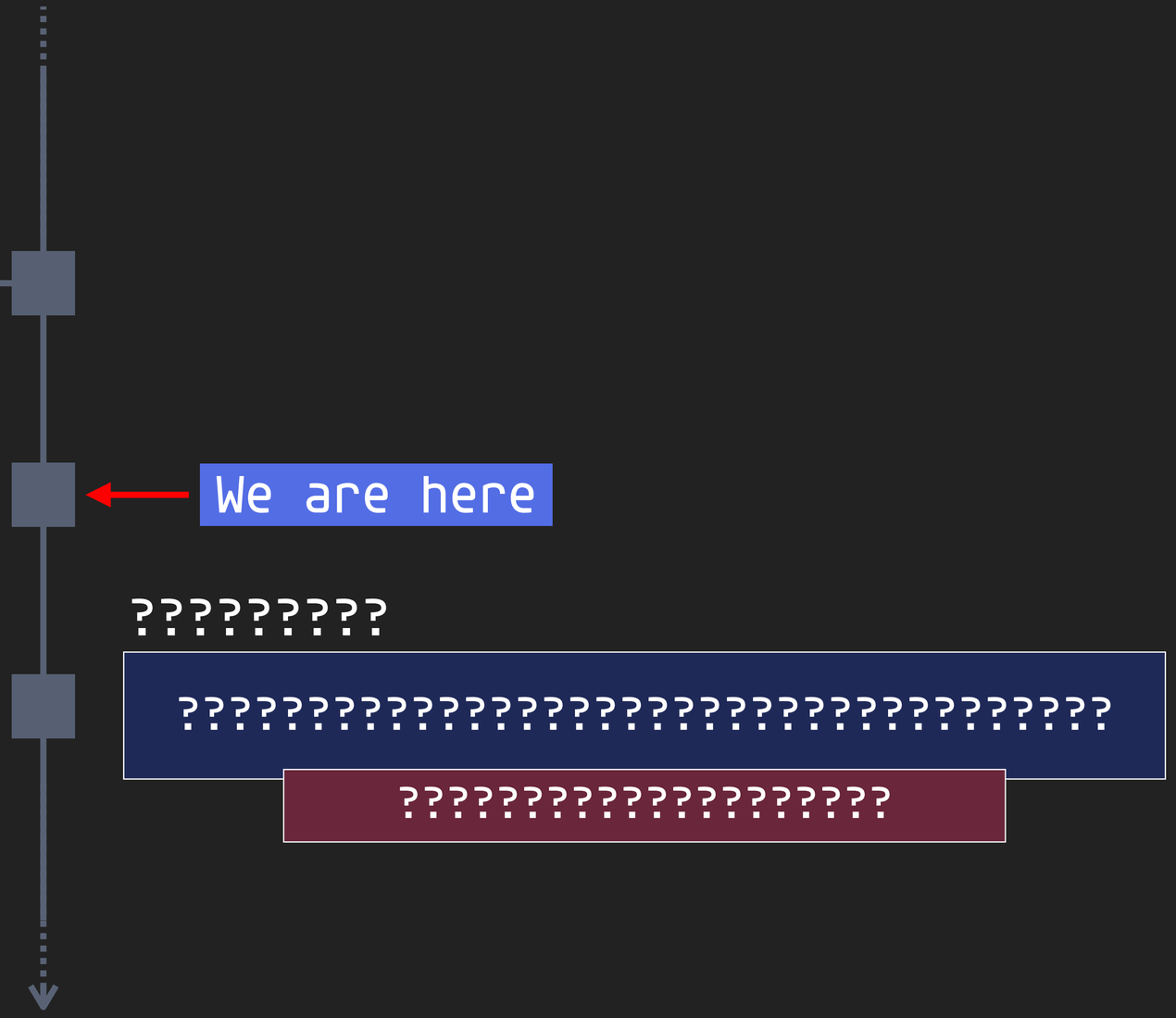
DEVCORE

2022

Ian Dupont, Harrison Green

Pulling MikroTik into the Limelight

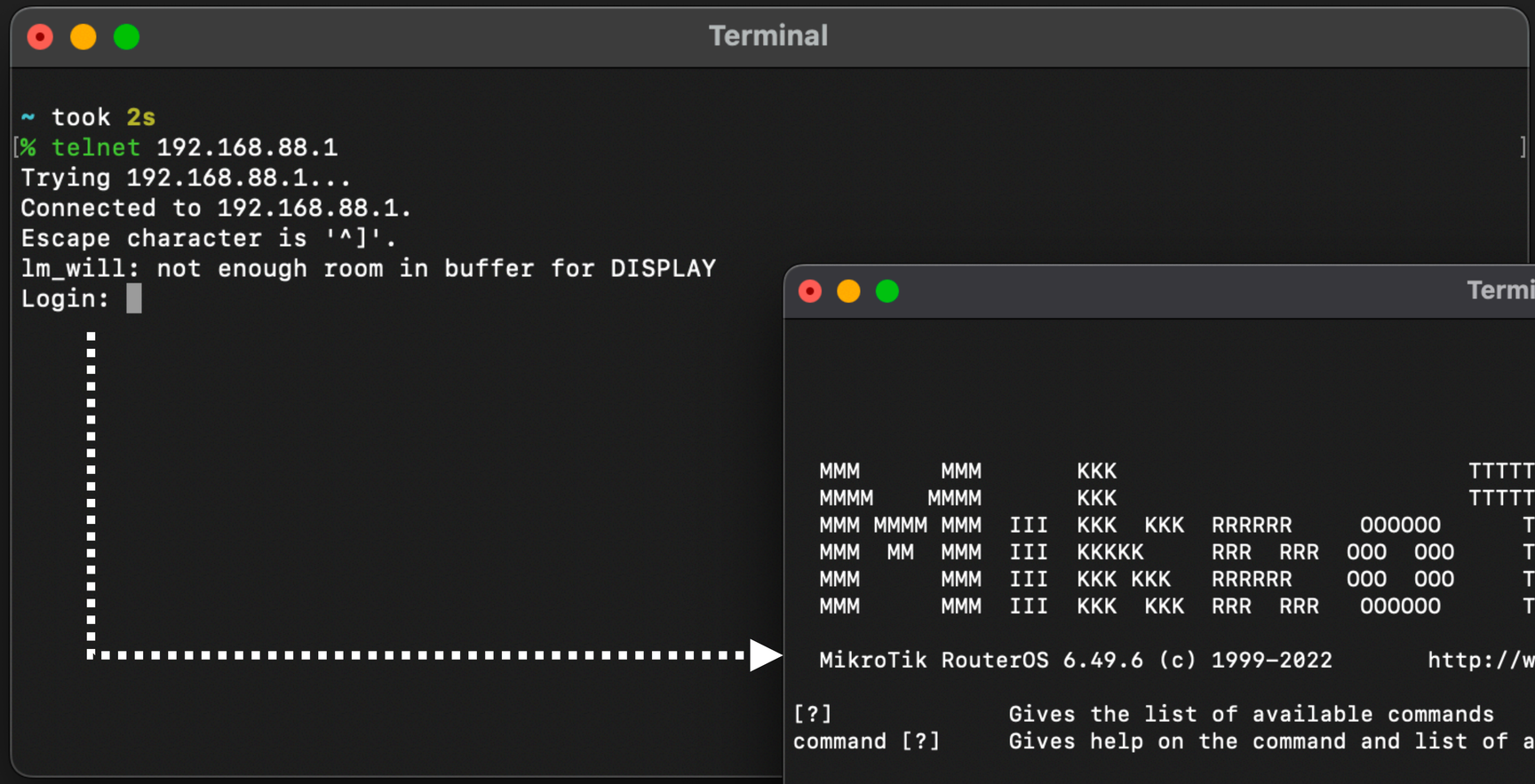
focus on nova message in IPC

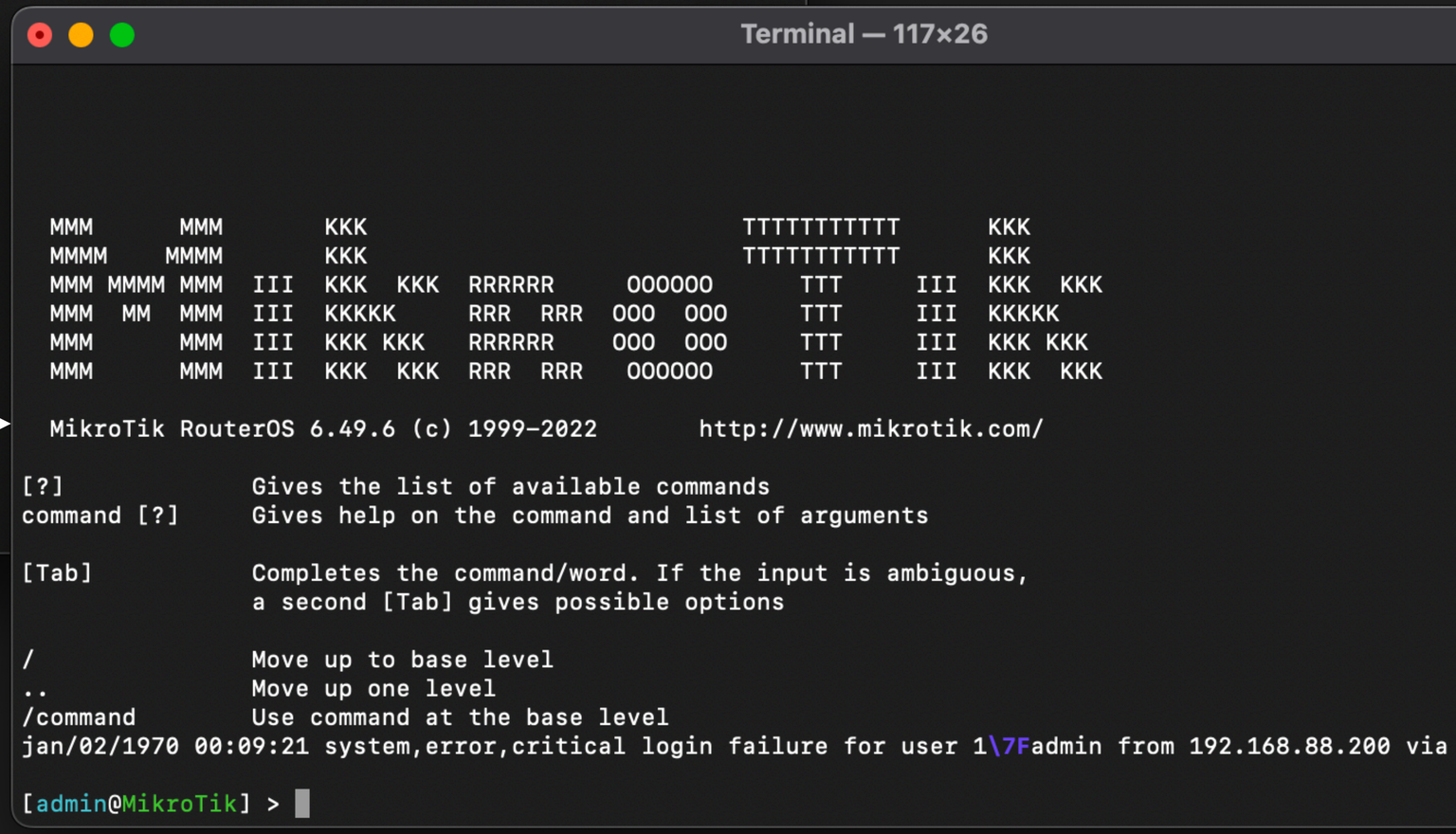


DEVCORE


Terminal
~ took 2s
[% telnet 192.168.88.1
Trying 192.168.88.1...
Connected to 192.168.88.1.
Escape character is '^]'.
lm_will: not enough room in buffer for DISPLAY
Login:
Terminal — 117×26
IPC
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 192.168.88.200 via
[admin@MikroTik] >

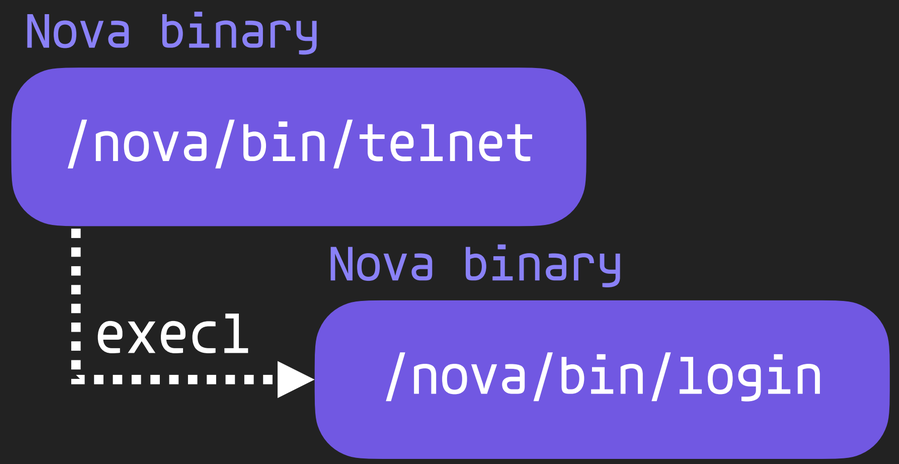
DEVCORE


Terminal
~ took 2s
[% telnet 192.168.88.1
Trying 192.168.88.1...
Connected to 192.168.88.1.
Escape character is '^]'.
lm_will: not enough room in buffer for DISPLAY
Login:


Terminal — 117×26
MMM MMM KKK TTTTTTTTTTT KKK
MMMM MMMM KKK TTTTTTTTTTT KKK
MMM MMMM MMM III KKK KKK RRRRRR OOOOOO TTT III KKK KKK
MMM MM MMM III KKKKK RRR RRR OOO OOO TTT III KKKKK
MMM MMM III KKK KKK RRRRRR OOO OOO TTT III KKK KKK
MMM MMM III KKK KKK RRR RRR OOOOOO 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 192.168.88.200 via
[admin@MikroTik] >

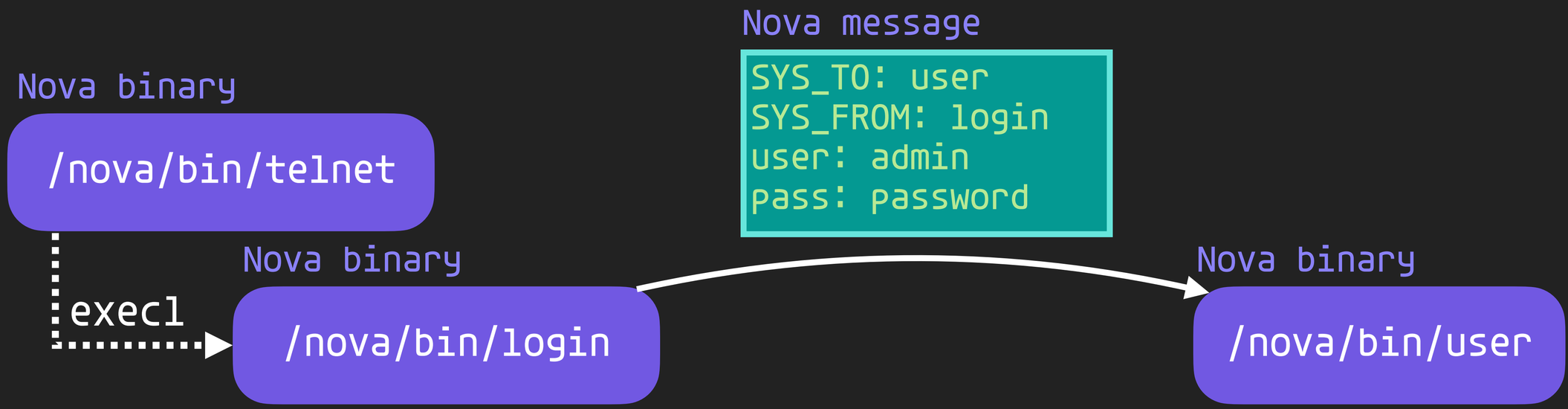
DEVCORE

IPC


Nova binary
/nova/bin/telnet
execl
Nova binary
/nova/bin/login

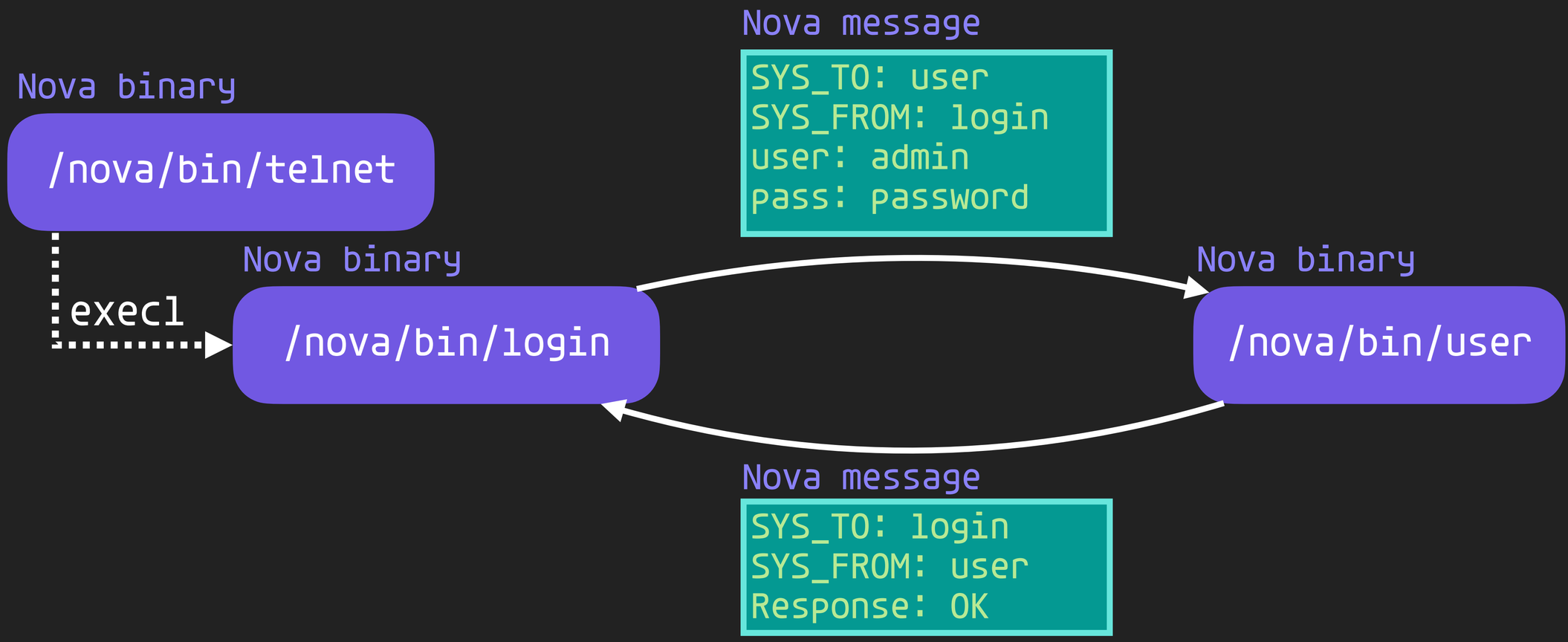
DEVCORE

IPC


Nova message
SYS_TO: user
SYS_FROM: login
user: admin
pass: password
Nova binary
/nova/bin/telnet
execl
Nova binary
/nova/bin/login
Nova binary
/nova/bin/user

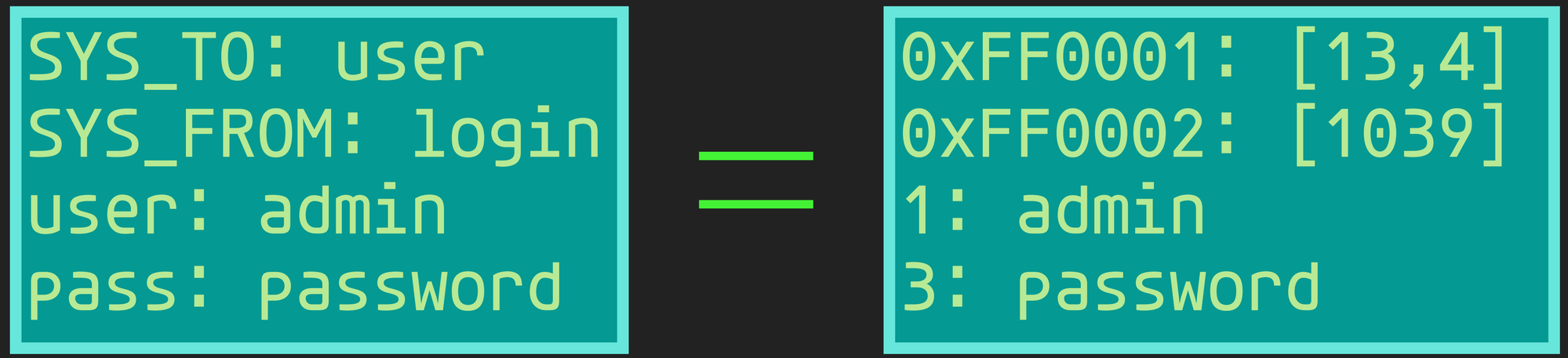
DEVCORE

IPC

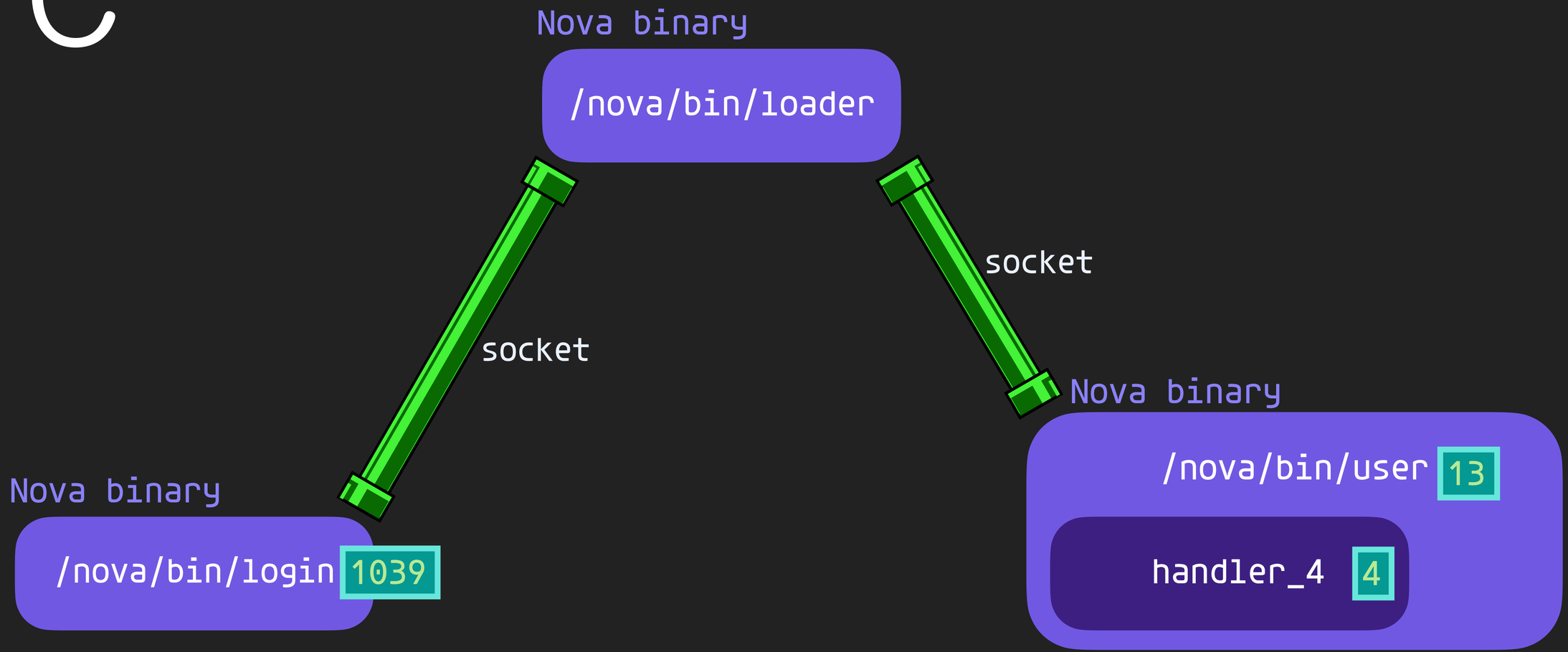

Nova message
SYS_TO: user
SYS_FROM: login
user: admin
pass: password
Nova binary
/nova/bin/telnet
execl
Nova binary
/nova/bin/login
Nova binary
/nova/bin/user
Nova message
SYS_TO: login
SYS_FROM: user
Response: OK

DEVCORE

Nova Message

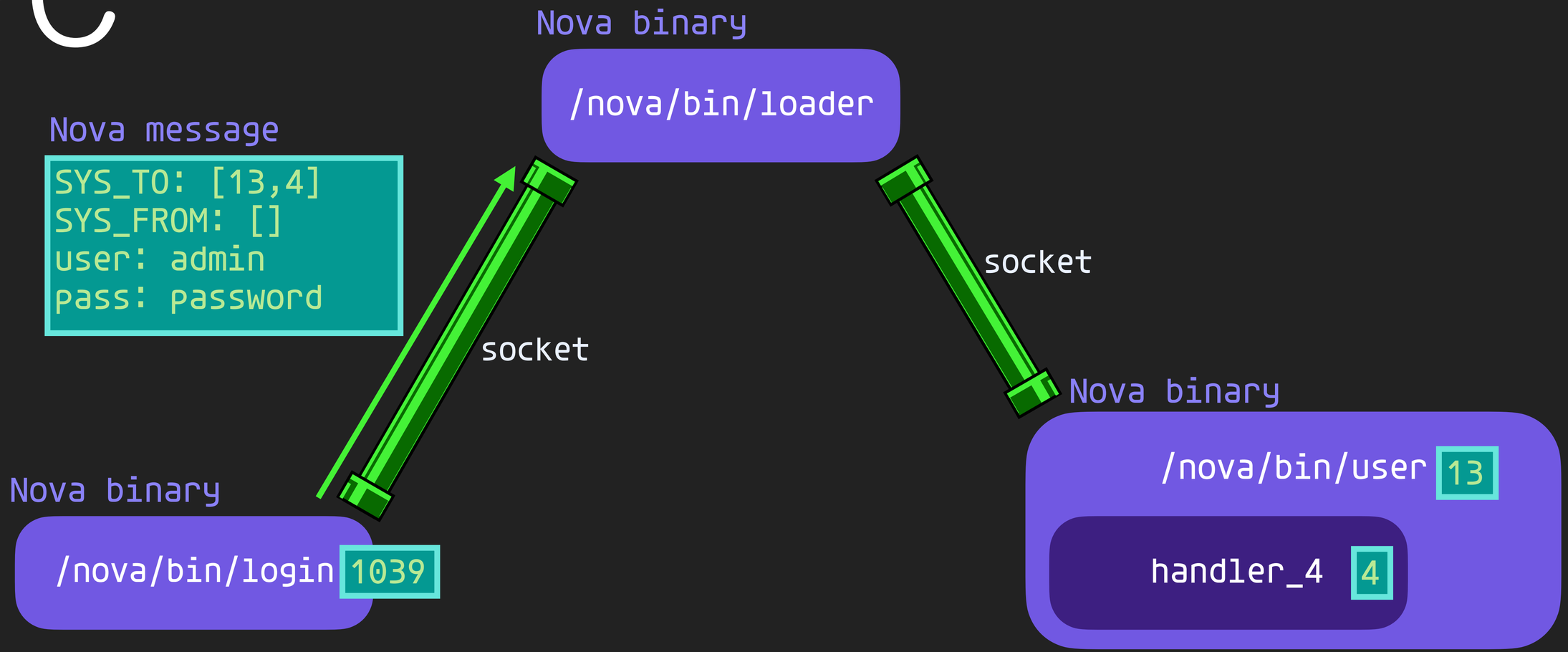


IPC


Nova binary
/nova/bin/loader
socket
socket
Nova binary
/nova/bin/login
1039
Nova binary
/nova/bin/user
13
handler_4
4

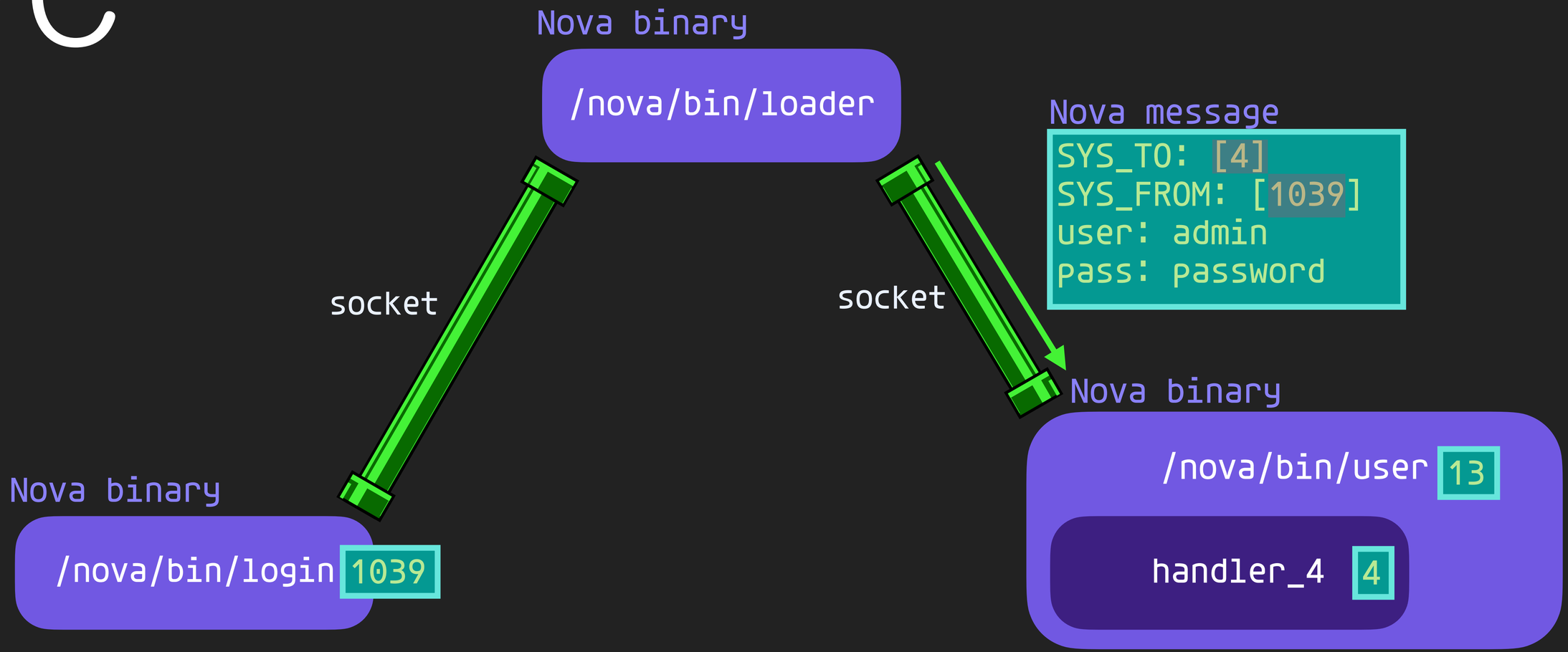
DEVCORE

IPC


Nova binary
/nova/bin/loader
Nova message
SYS_TO: [13,4]
SYS_FROM: []
user: admin
pass: password
socket
socket
Nova binary
Nova binary
/nova/bin/user
13
handler_4
4
/nova/bin/login
1039

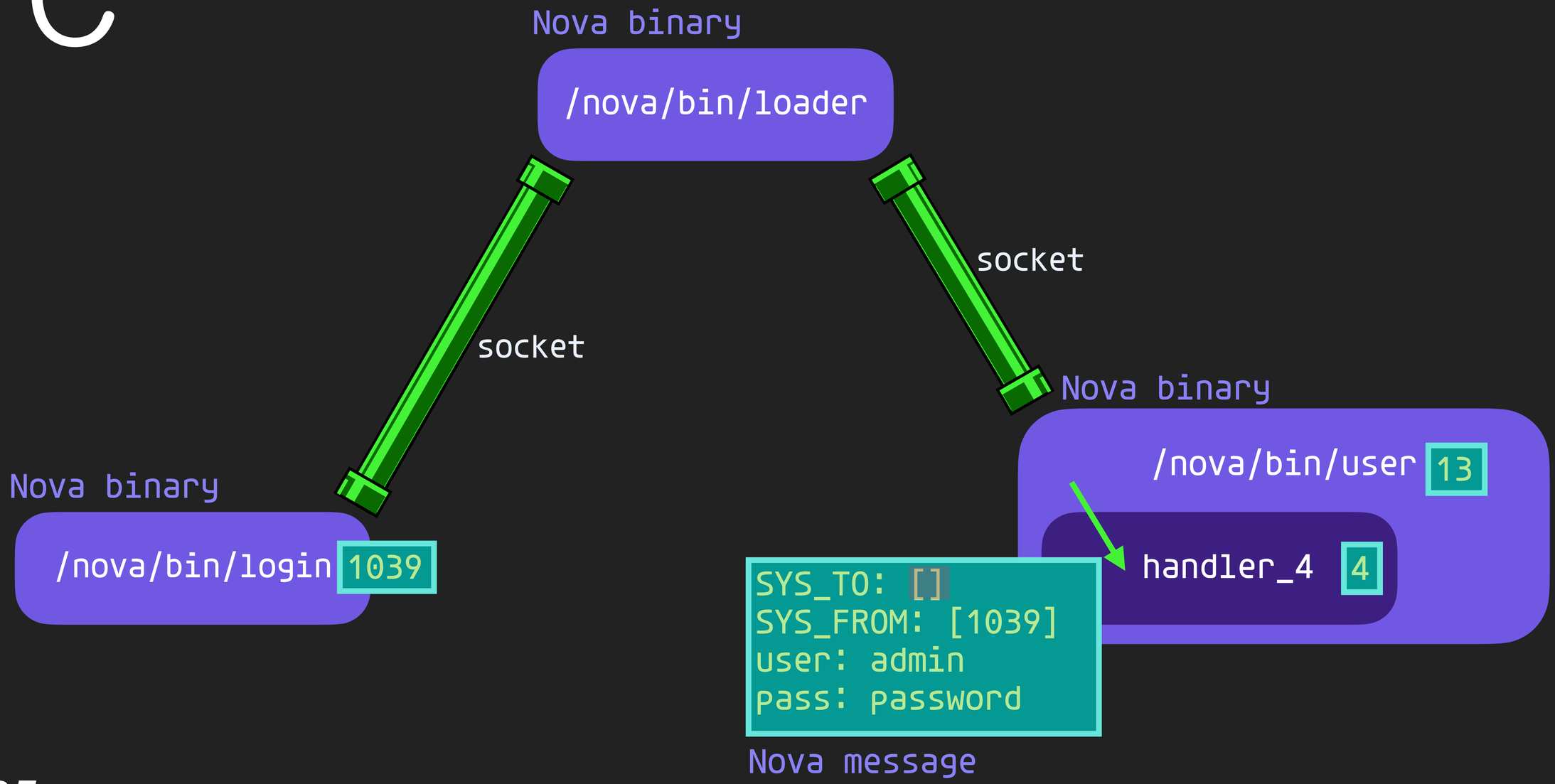
DEVCORE

IPC


Nova binary
/nova/bin/loader
Nova message
SYS_TO: [4]
SYS_FROM: [1039]
user: admin
pass: password
socket
socket
Nova binary
/nova/bin/user 13
handler_4 4
Nova binary
/nova/bin/login 1039

DEVCORE

IPC


Nova binary
/nova/bin/loader
socket
socket
Nova binary
/nova/bin/login
1039
Nova binary
/nova/bin/user
13
handler_4
4
SYS_TO: []
SYS_FROM: [1039]
user: admin
pass: password
Nova message

DEVCORE

IPC

Nova binary

/nova/bin/loader

Nova binary

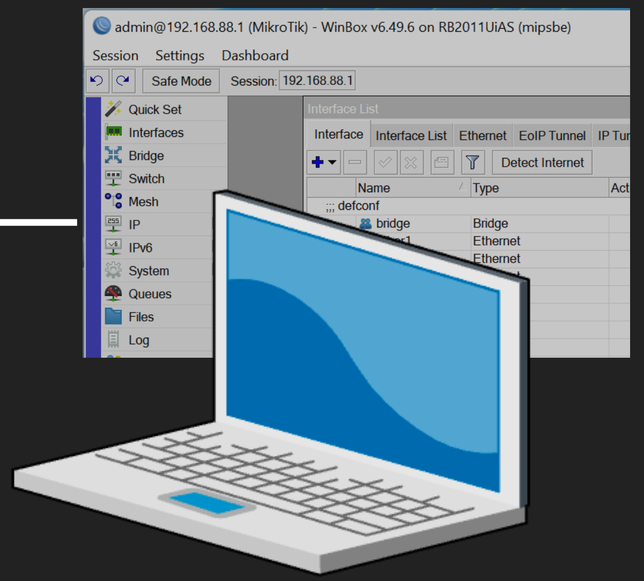
/nova/bin/sermgr

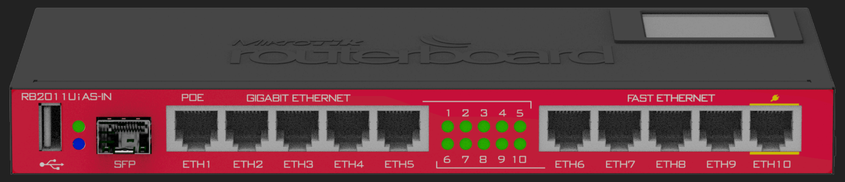
TCP

Nova message

SYS_TO: […]
SYS_FROM: […]
…

winbox





DEVCORE

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);
```

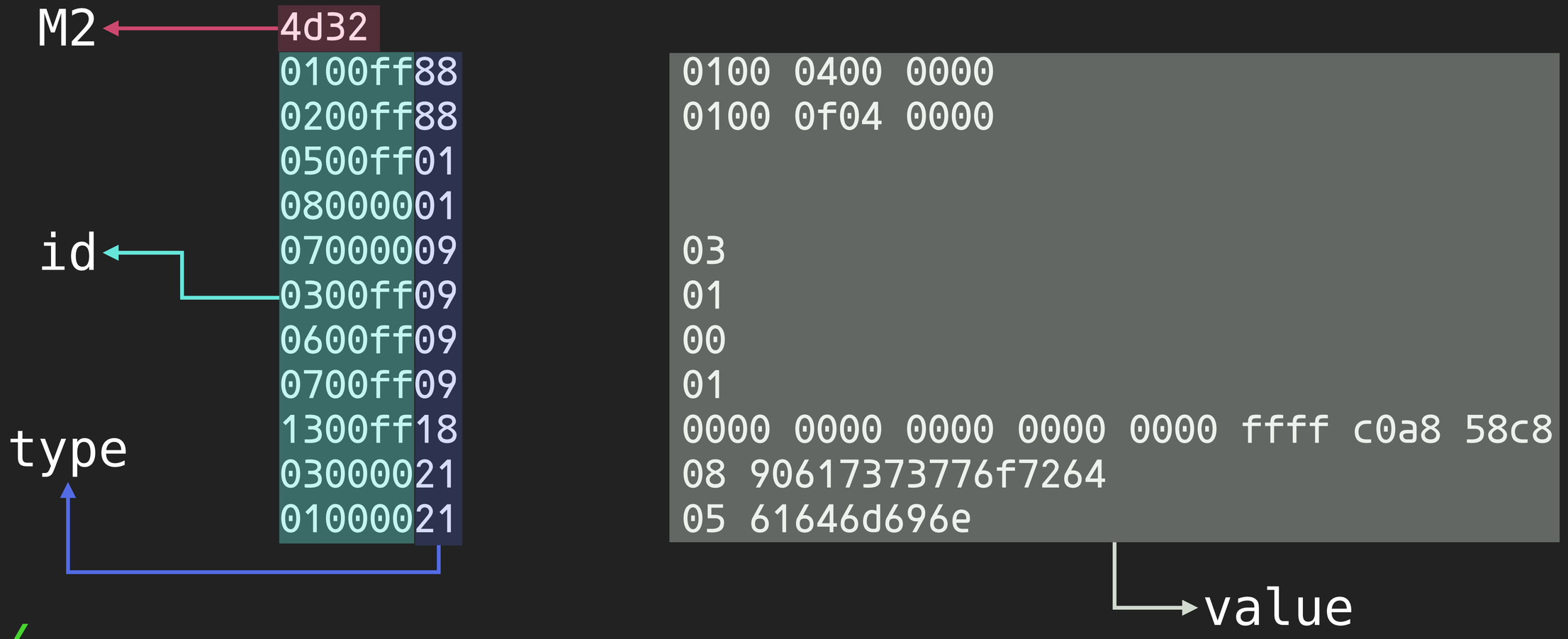
Nova Message

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

=

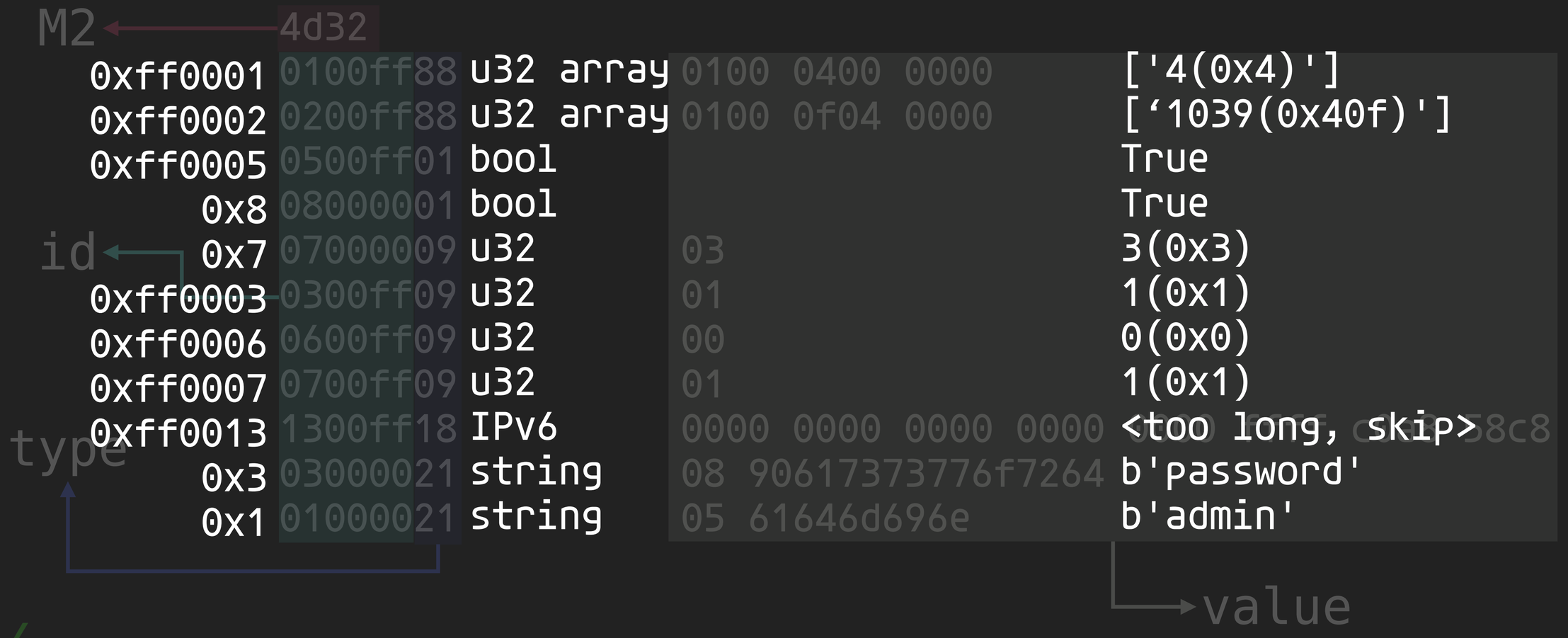
```
00000000: 5d00 0000 4d32 0100 ff88 0100 0400 0000  ]...M2..........
00000010: 0200 ff88 0100 0f04 0000 0500 ff01 0800  ................
00000020: 0001 0700 0009 0303 00ff 0901 0600 ff09  ................
00000030: 0007 00ff 0901 1300 ff18 0000 0000 0000  ................
00000040: 0000 0000 ffff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

Nova Message



DEVCORE

Nova Message



DEVCORE

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/>
```

DEVCORE

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>%
```

DEVCORE

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

```
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>
```

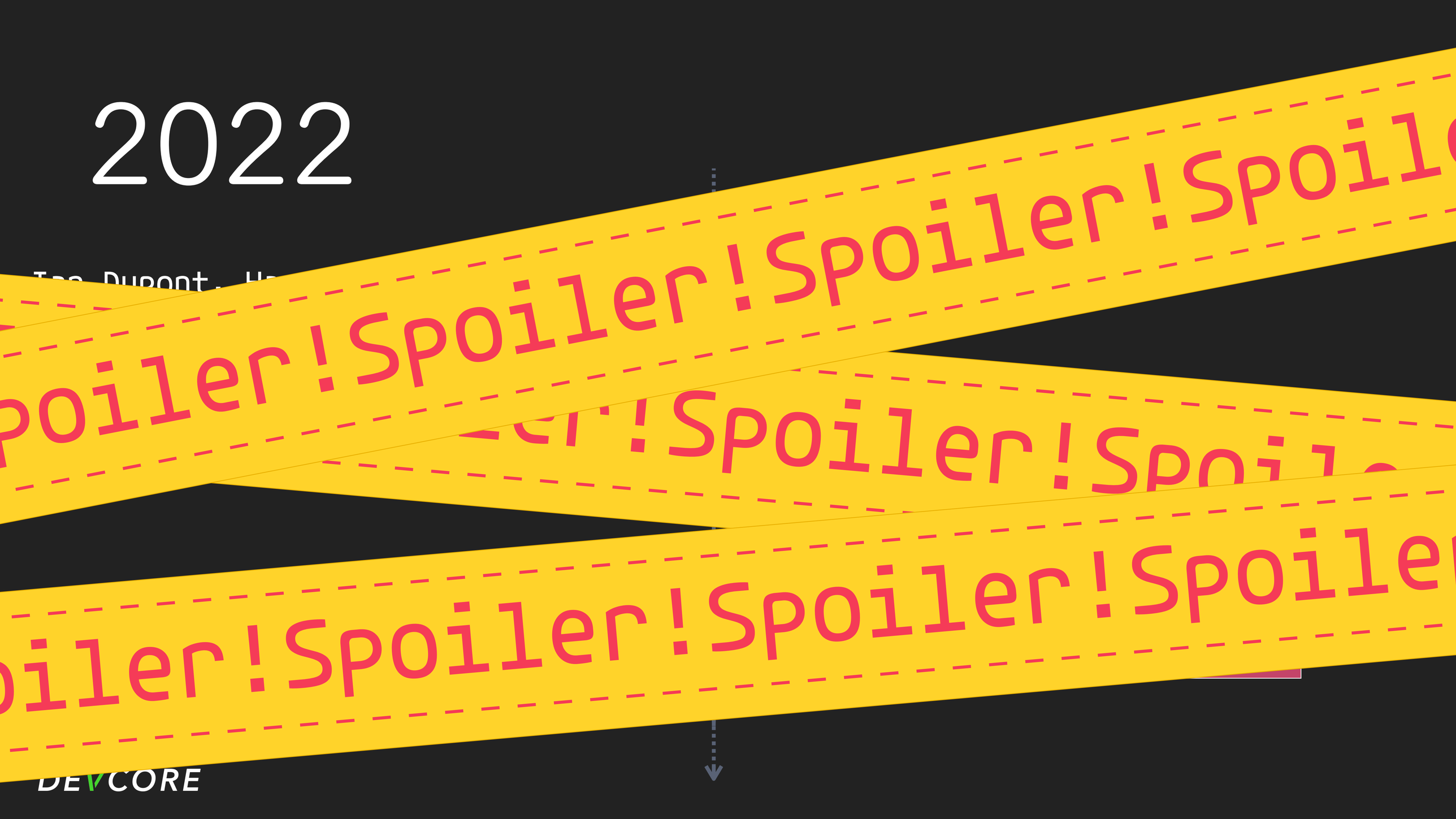
Nova Message

- 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
```

DEVCORE

2022


Spoiler!Spoiler!Spoiler!

2022

Ian Dupont, Harrison Green

Pulling MikroTik into the Limelight

focus on nova message in IPC

We are here

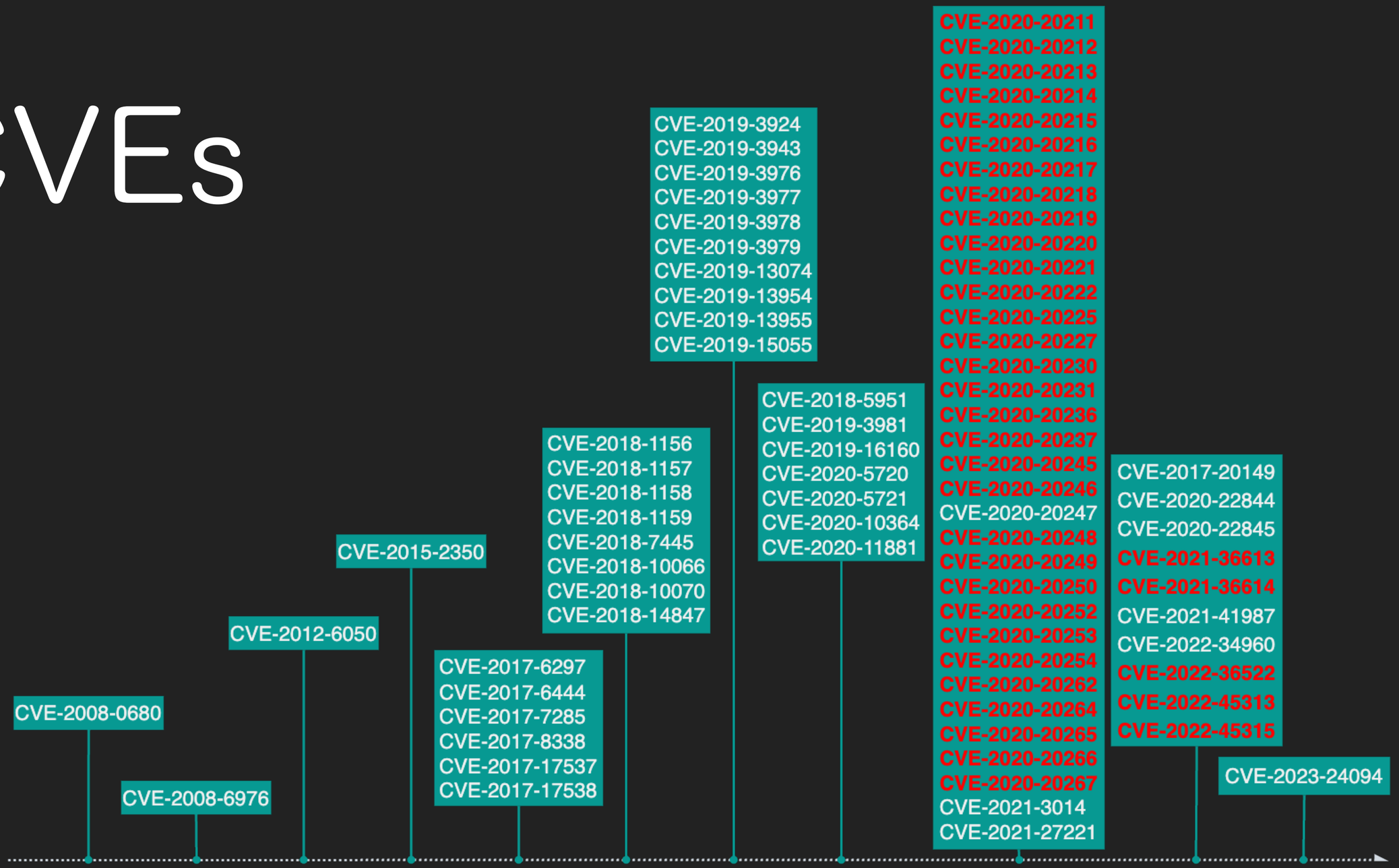
Qian Chen

MikroTik RouterOS Security:
The Forgotten IPC Message

fuzzing nova message in IPC

DEVCORE

CVEs



DEVCORE



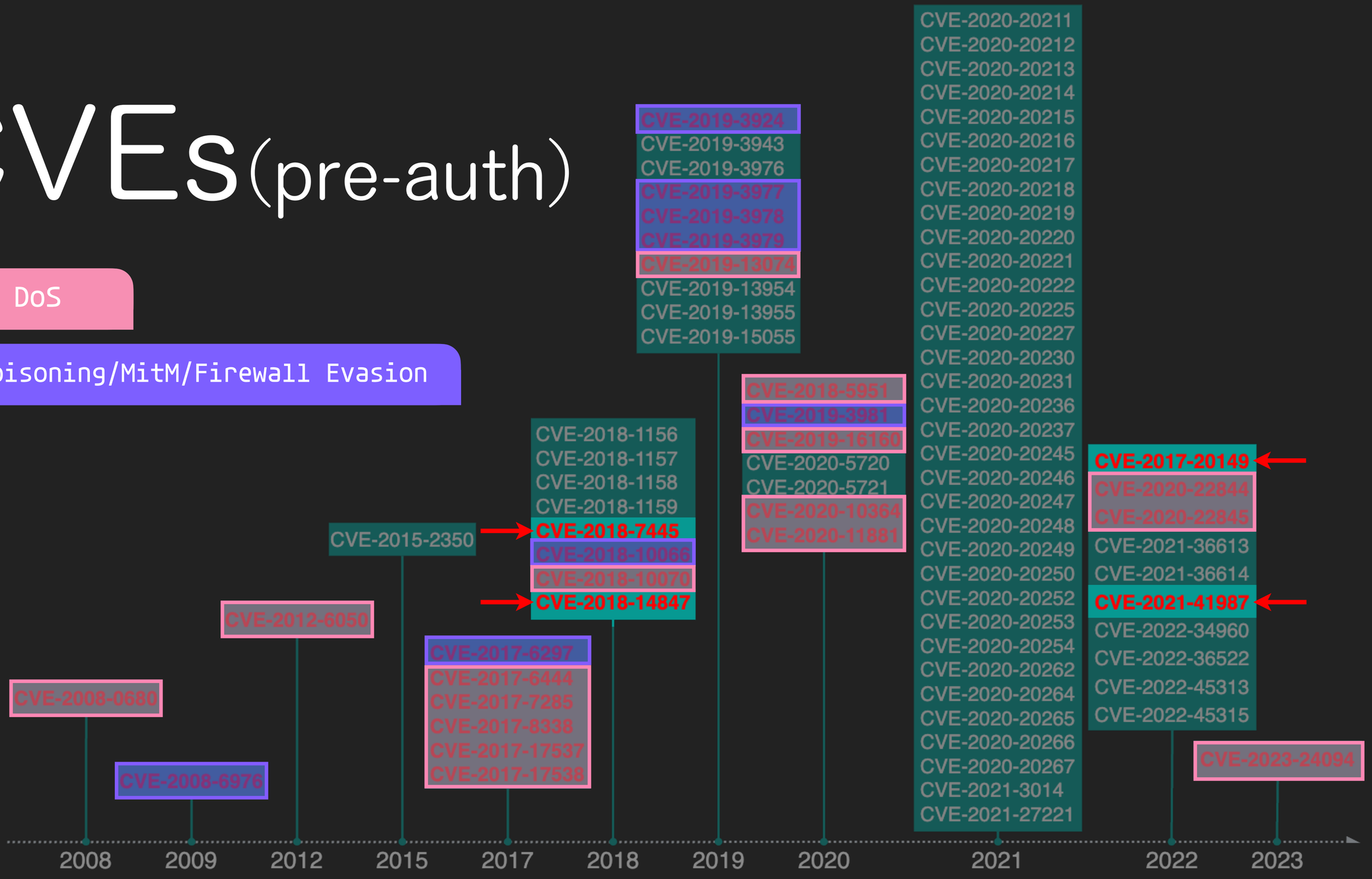
Is RouterOS now impervious to hacking?

DEVCORE

CVEs(pre-auth)

DoS

Poisoning/MitM/Firewall Evasion



DEVCORE

Observation

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

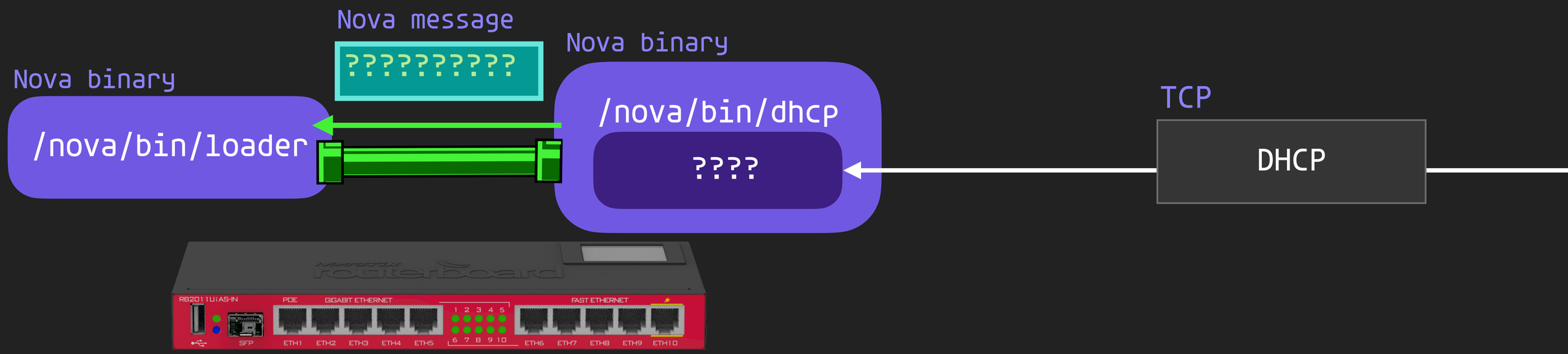
Brief Summary

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 ?

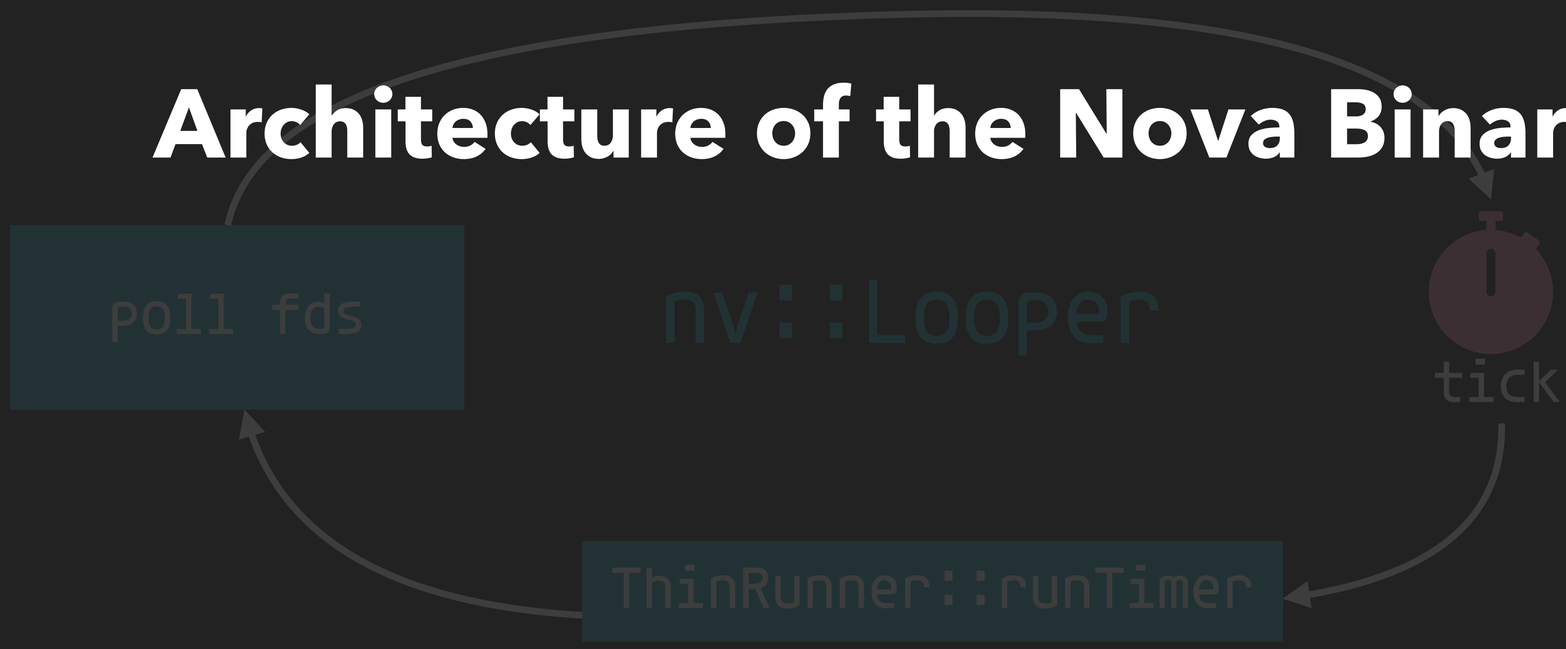


DEVCORE

Nova Binary

- Every Nova binary has a Looper or MultifiberLooper.

Architecture of the Nova Binary



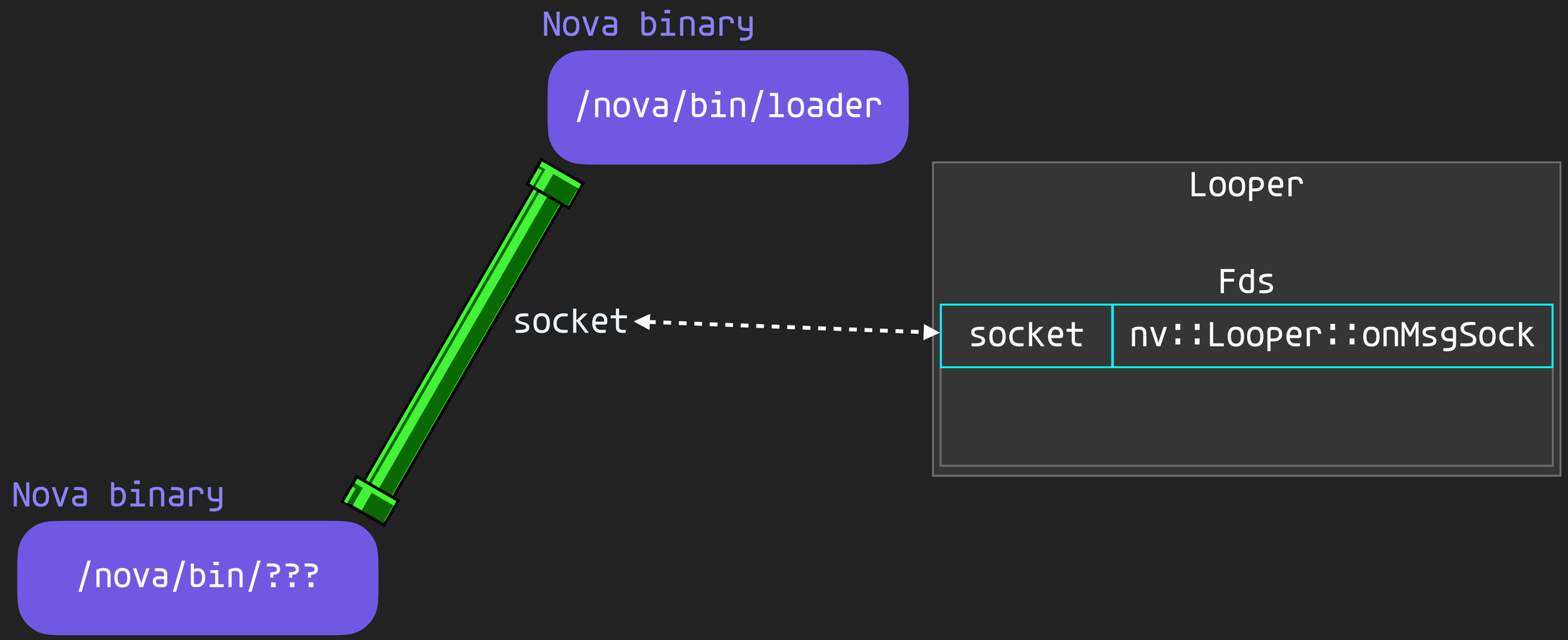
Nova Binary

- Every Nova binary has a Looper or a MultifiberLooper.

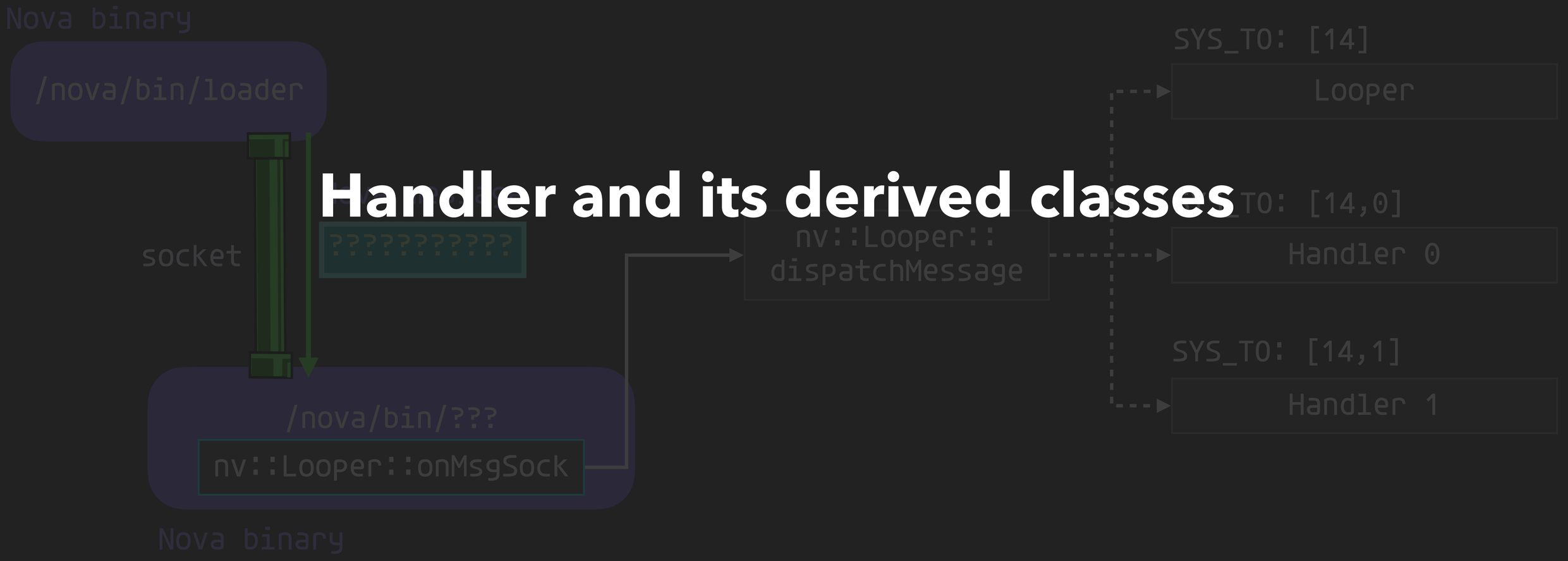


DEVCORE

Nova Binary

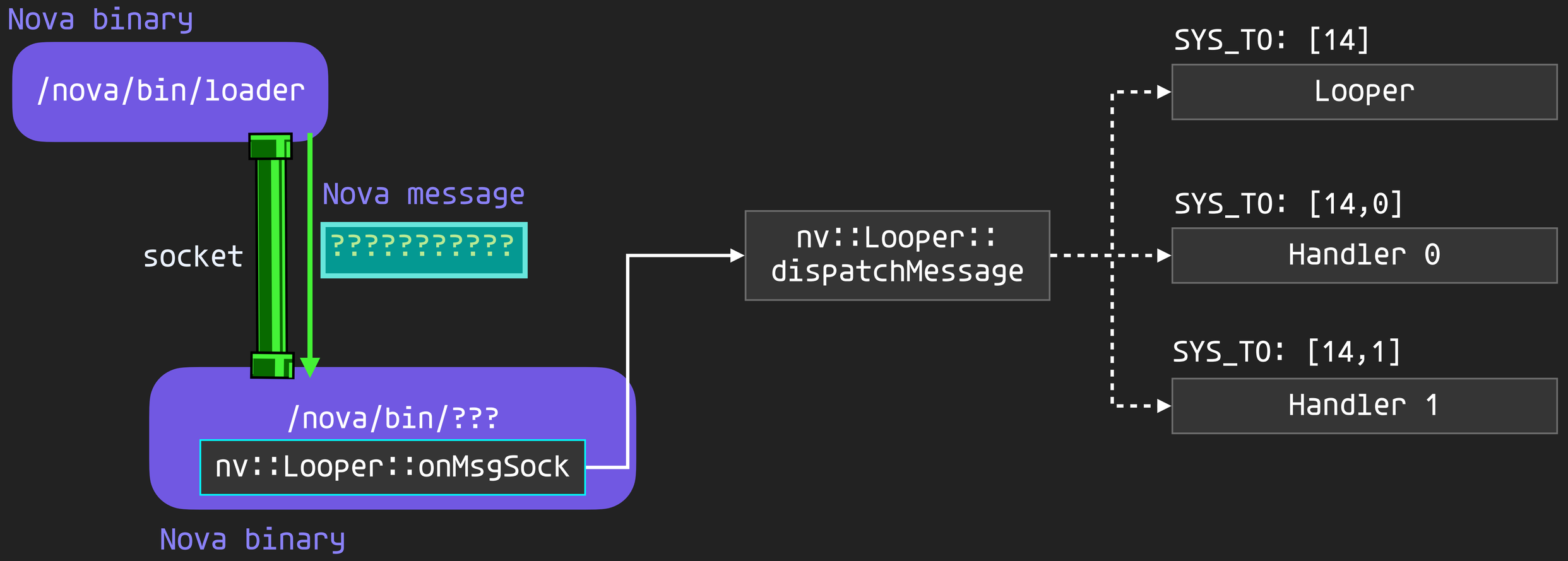

Nova binary
/nova/bin/loader
socket
Looper
Fds
socket
nv::Looper::onMsgSock
Nova binary
/nova/bin/???

Nova Binary



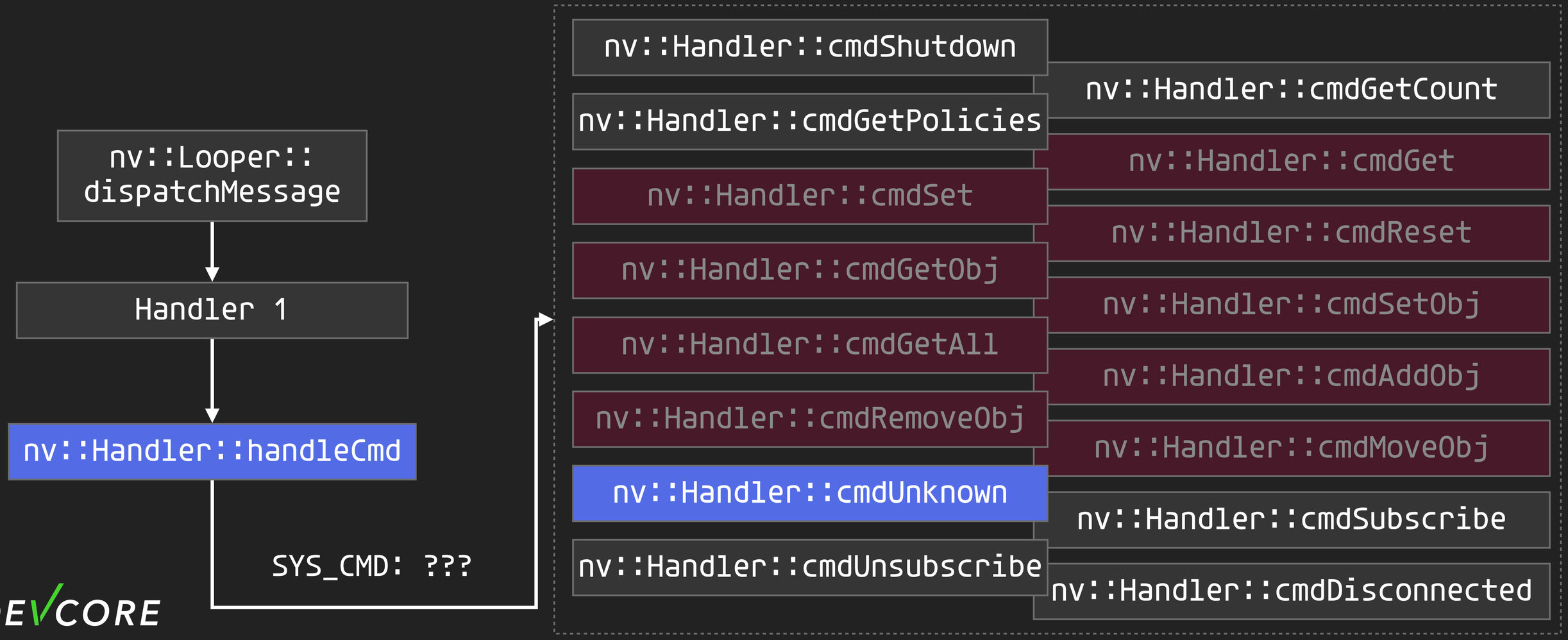
Handler and its derived classes

Nova Binary



DEVCORE

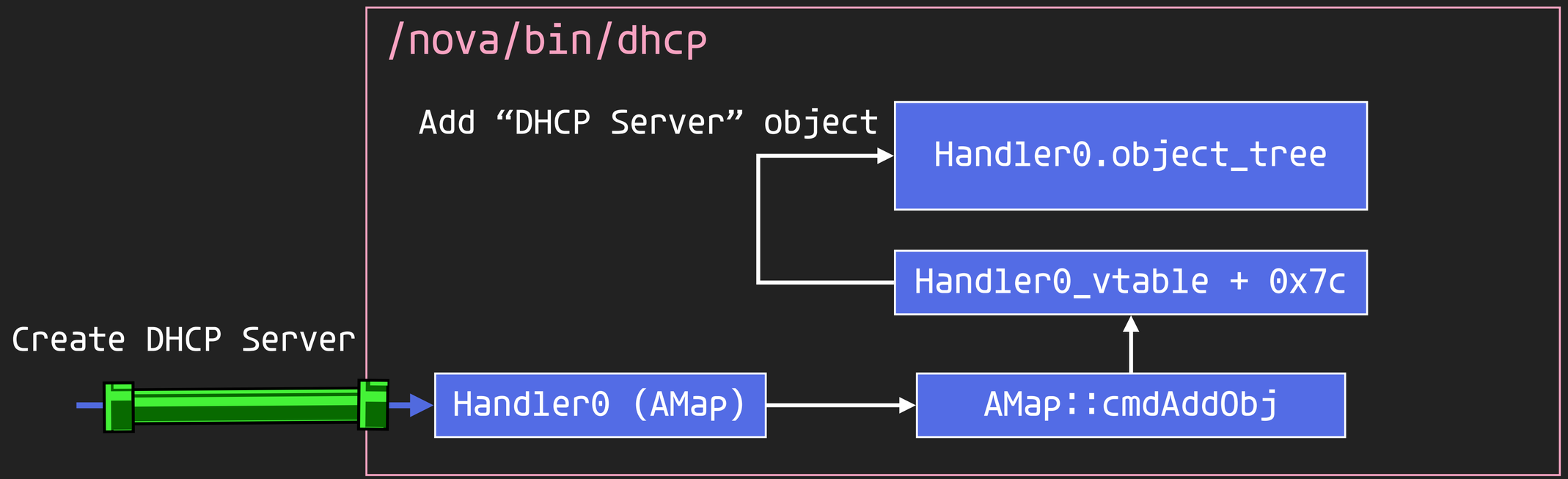
Nova Binary



DEVCORE

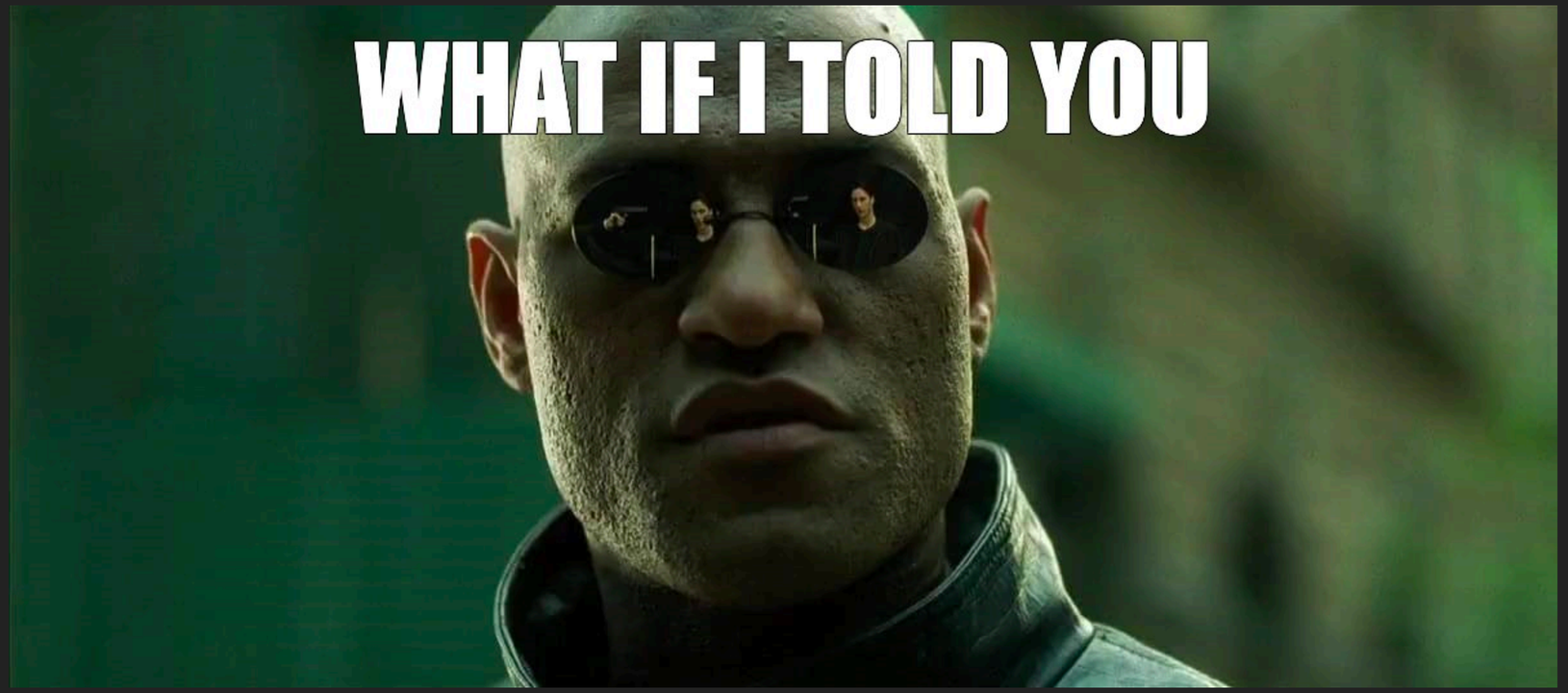
Nova Binary

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

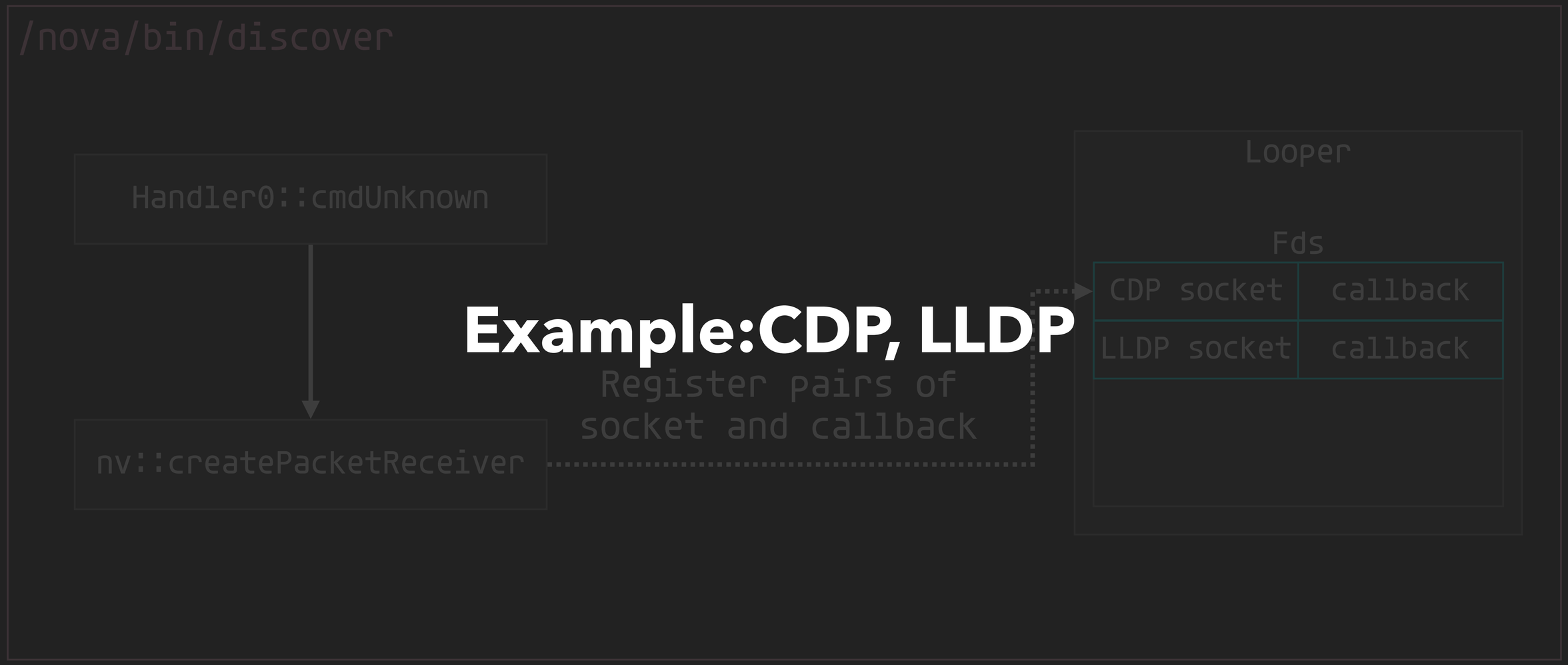


Nova Message

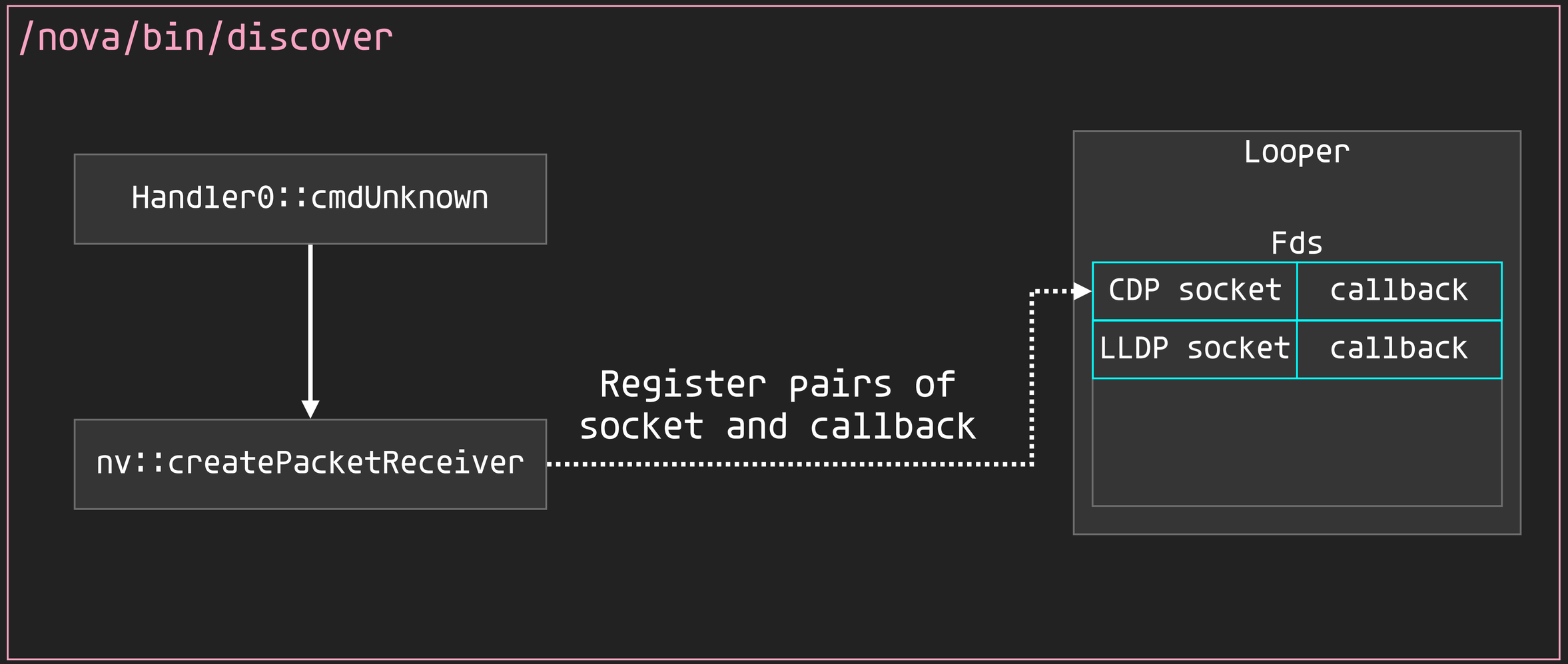
- Some functionalities don’t even use the Nova message.

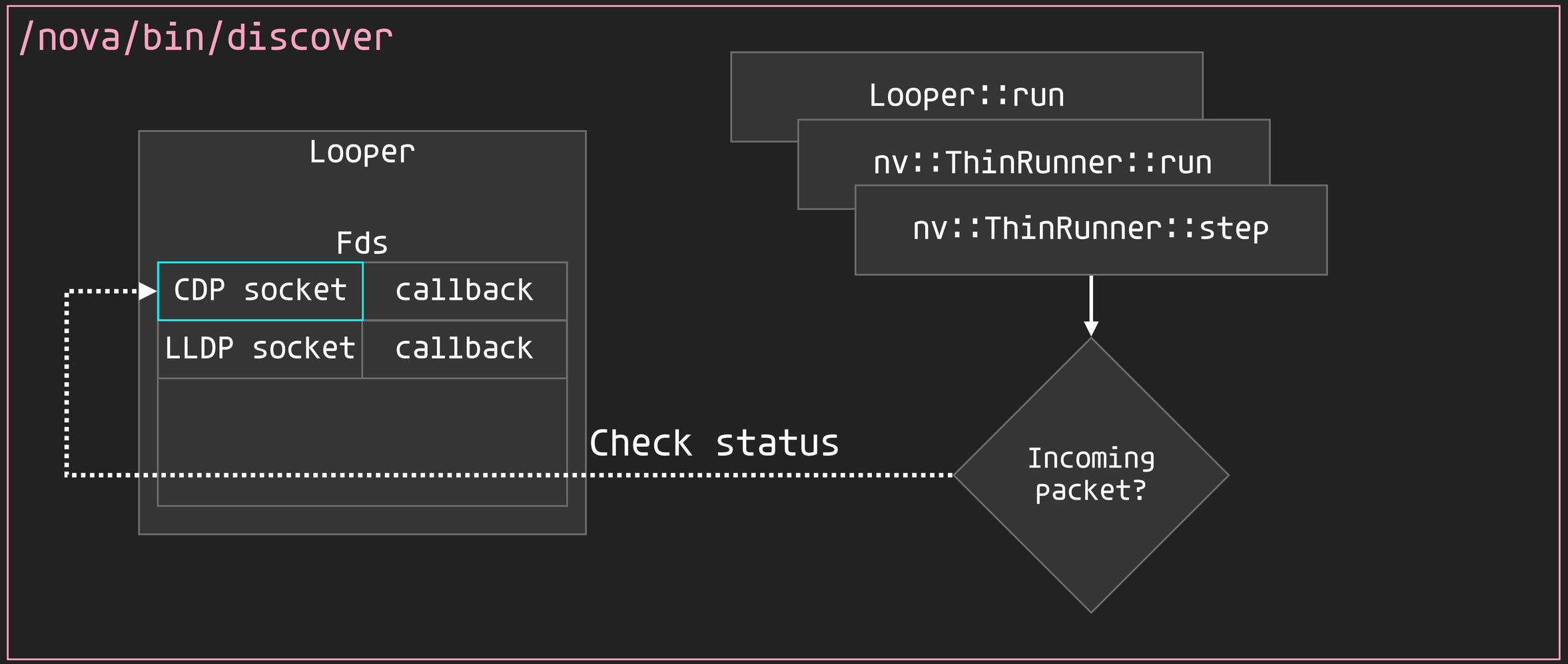


DEVCORE

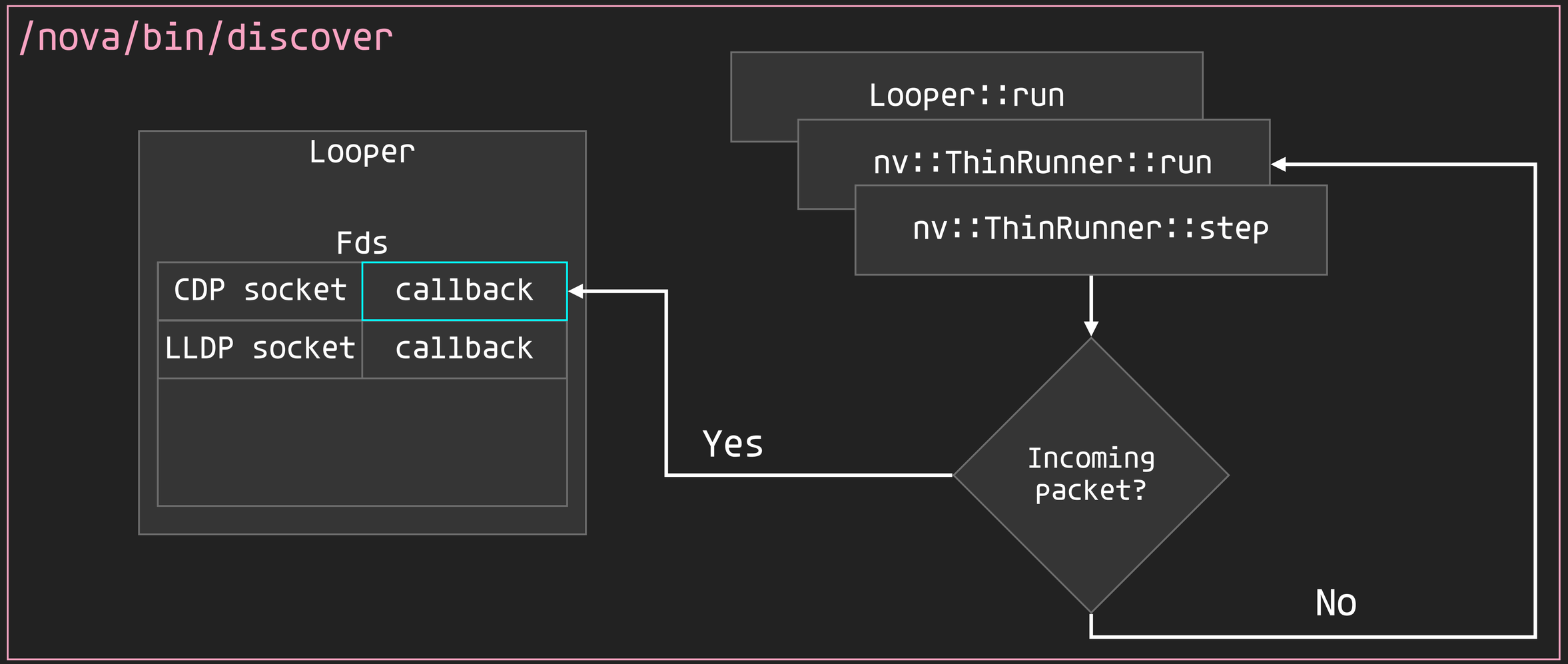


Example:CDP, LLDP

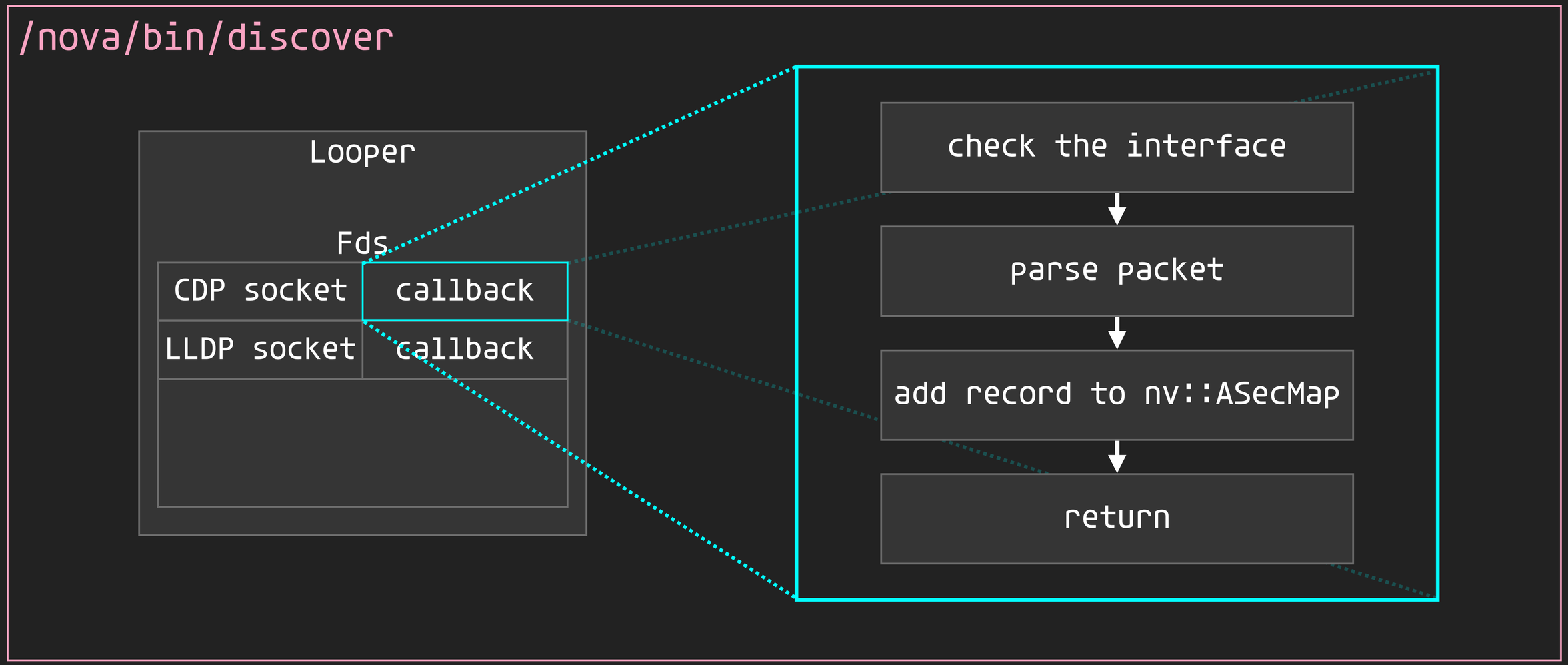

/nova/bin/discover
Handler0::cmdUnknown
nv::createPacketReceiver
Register pairs of
socket and callback
Looper
Fds
CDP socket
callback
LLDP socket
callback


/nova/bin/discover
Looper::run
nv::ThinRunner::run
nv::ThinRunner::step
Looper
Fds
CDP socket
callback
LLDP socket
callback
Check status
Incoming packet?

DEVCORE


/nova/bin/discover
Looper::run
nv::ThinRunner::run
nv::ThinRunner::step
Looper
Fds
CDP socket
callback
LLDP socket
callback
Yes
Incoming packet?
No

DEVCORE


/nova/bin/discover
Looper
Fds
CDP socket
callback
LLDP socket
callback
check the interface
parse packet
add record to nv::ASecMap
return

DEVCORE

The pre-auth RCE

The pre-auth RCE

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

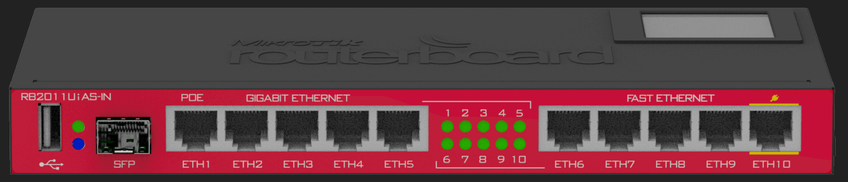


IPv6 SLAAC



ICMPv6

Router Solicitation (RS)



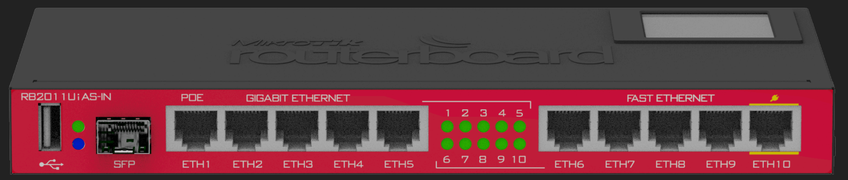
DEVCORE

IPv6 SLAAC



ICMPv6

Router Solicitation (RS)



ICMPv6

Router Advertisement (RA)

prefix: 2001:db8::/64

DEVCORE

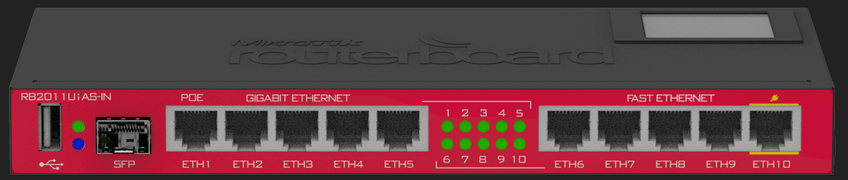
IPv6 SLAAC



IPv6 = prefix + EUI-64

ICMPv6

Router Solicitation (RS)



ICMPv6

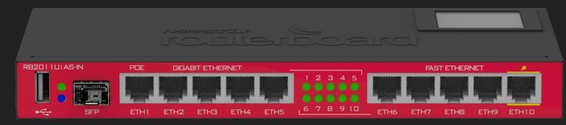
Router Advertisement (RA)

prefix: 2001:db8::/64

DEVCORE

IPv6 SLAAC

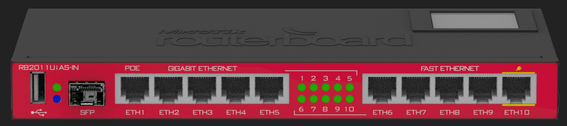
Router A



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

link prefix

Router B



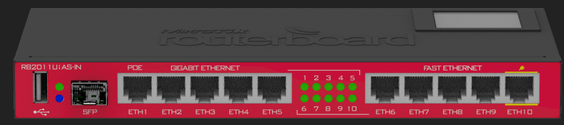
routed prefix



DEVCORE

IPv6 SLAAC

Router A



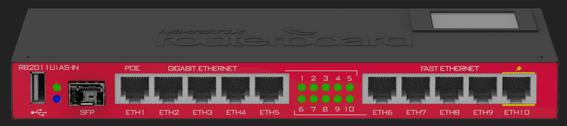
Route 2001:db8:1::/48
To ?????



RA

prefix: 2001:db8::/64
(link prefix)

Router B

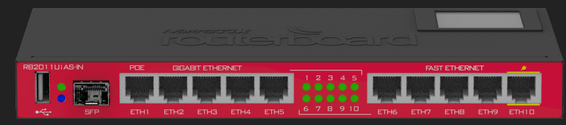




DEVCORE

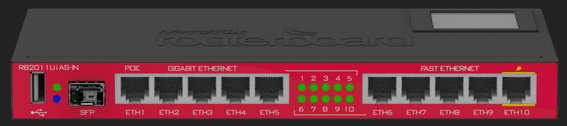
IPv6 SLAAC

Router A



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

Router B



2001:db8::1

RA

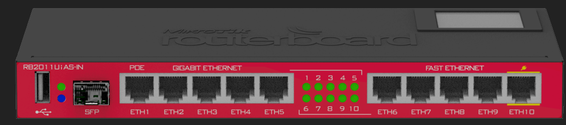
prefix: 2001:db8::/64
(link prefix)



DEVCORE

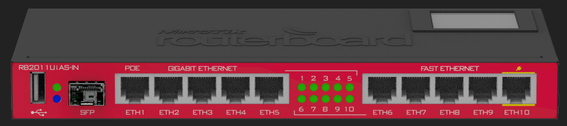
IPv6 SLAAC

Router A



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

Router B



2001:db8::1



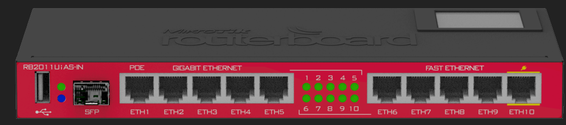
RA

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

DEVCORE

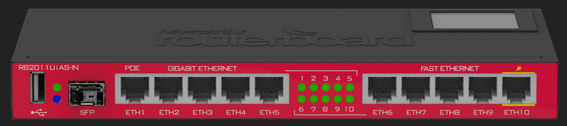
IPv6 SLAAC

Router A



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

Router B



2001:db8::1

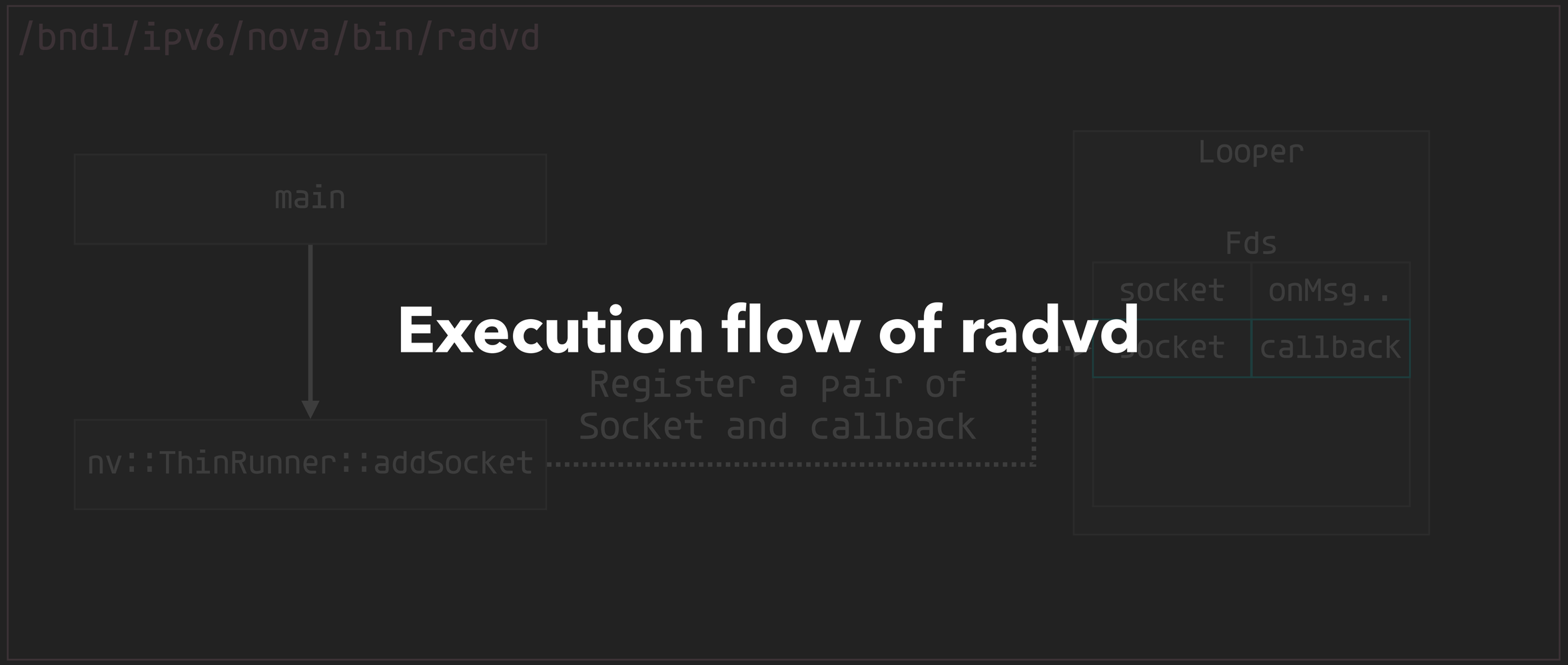


RA

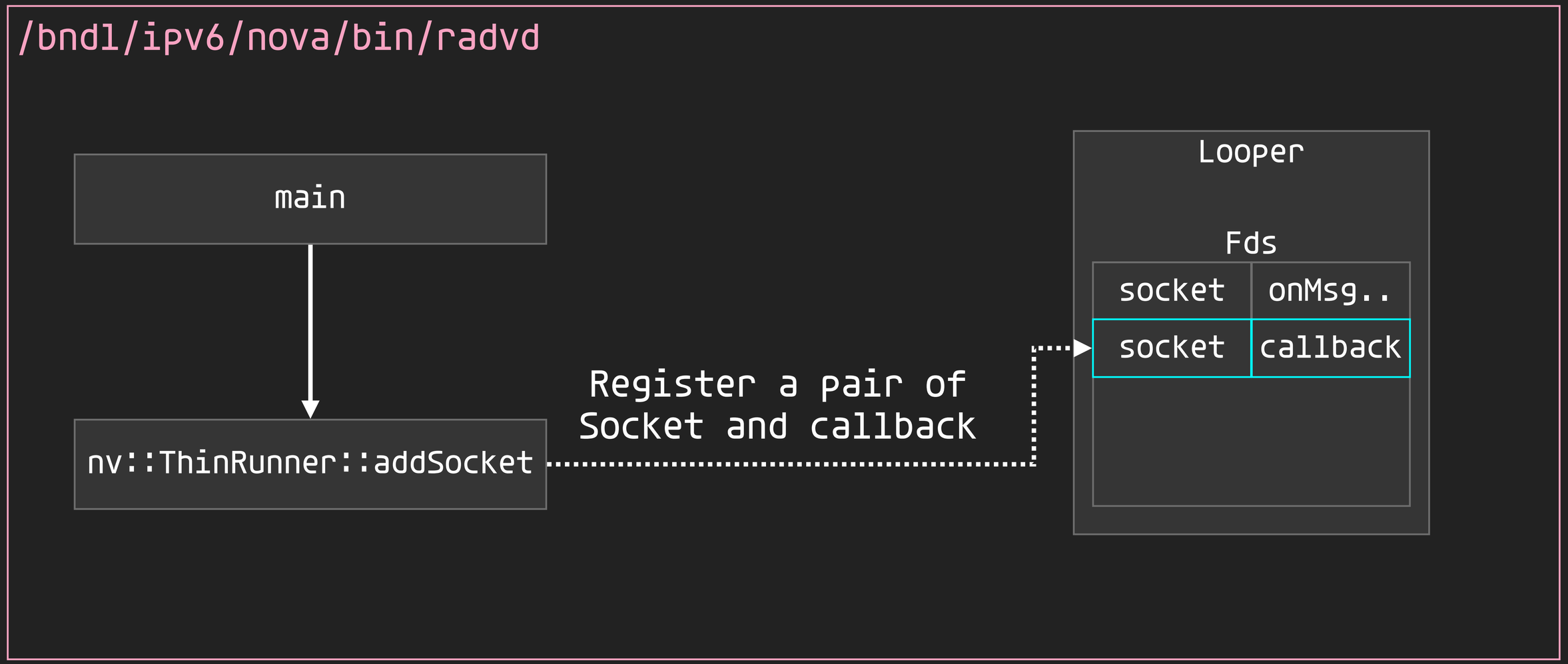
prefix: 2001:db8:1::/48
(routed prefix)

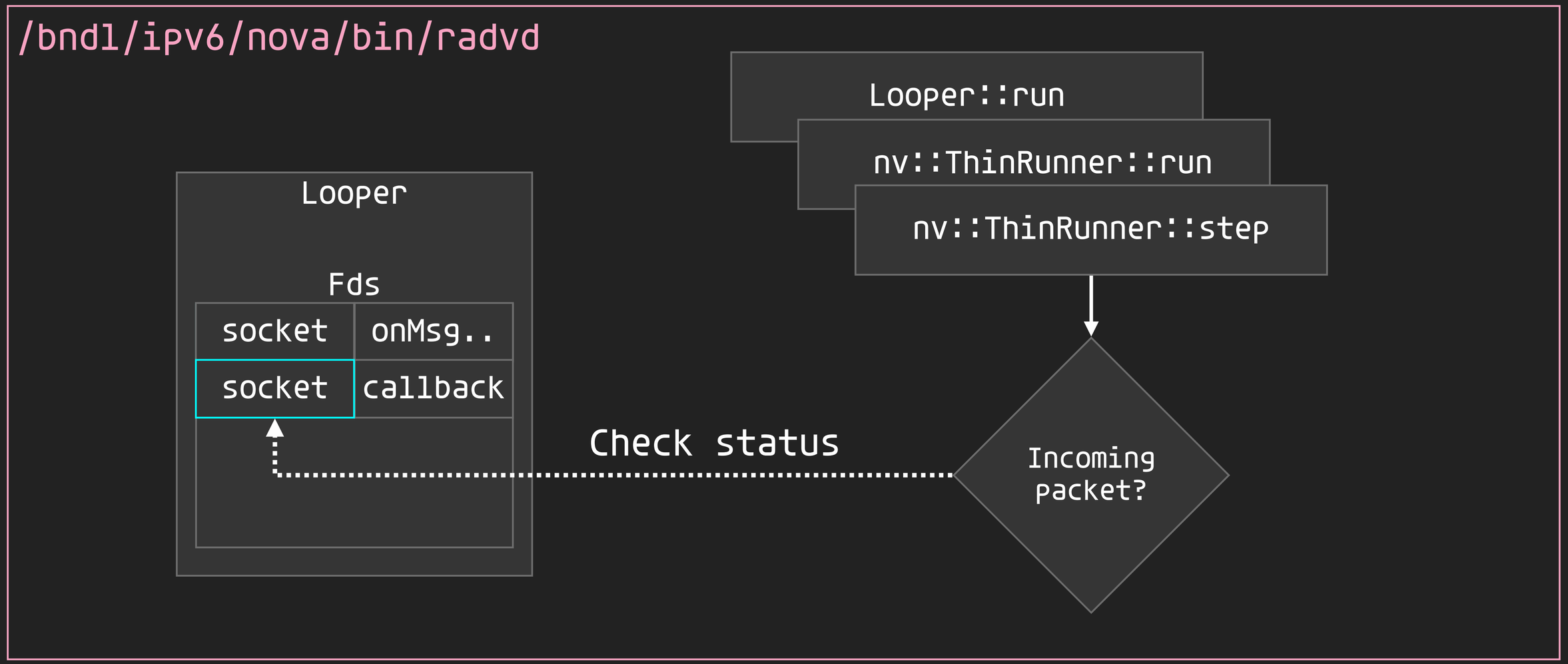
2001:db8:1::1

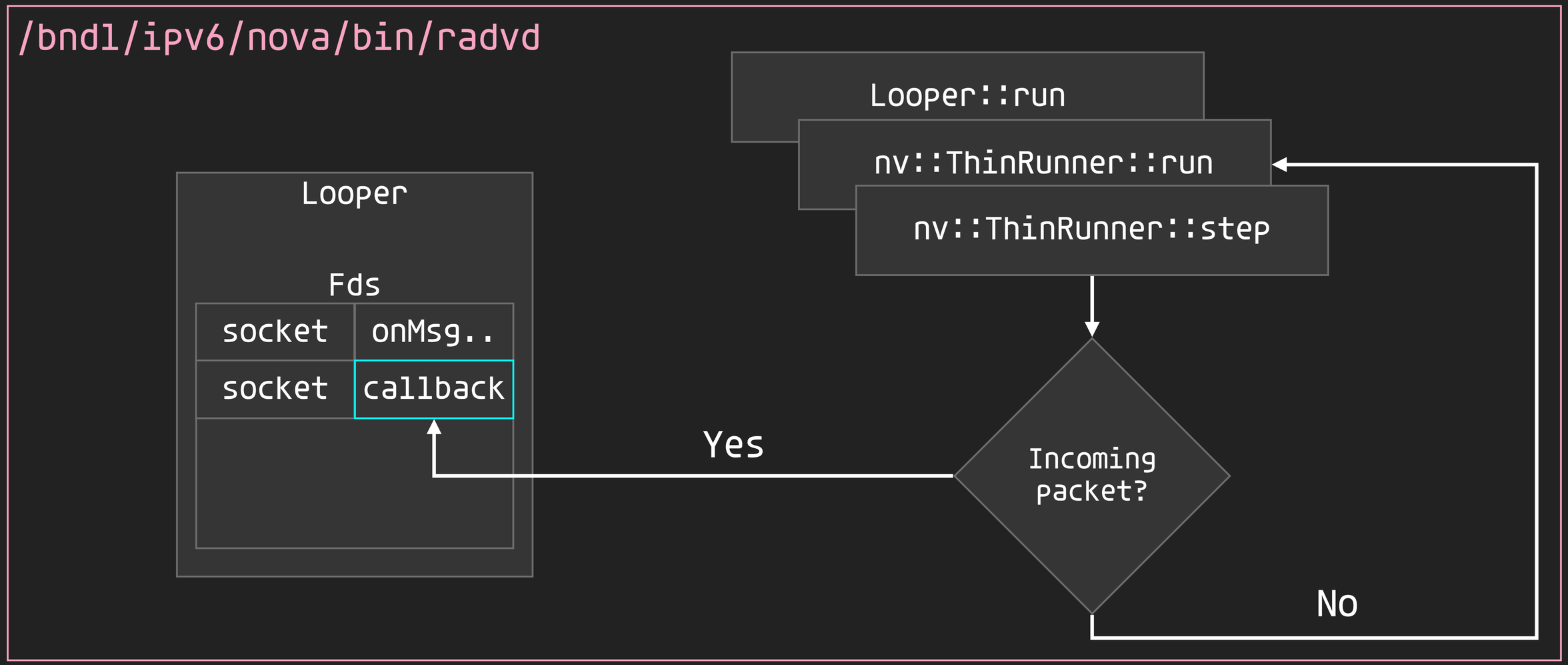
DEVCORE



Execution flow of radvd


/bndl/ipv6/nova/bin/radvd
main
nv::ThinRunner::addSocket
Register a pair of
Socket and callback
Looper
Fds
socket
onMsg..
socket
callback


/bndl/ipv6/nova/bin/radvd
Looper::run
nv::ThinRunner::run
nv::ThinRunner::step
Looper
Fds
socket
onMsg..
socket
callback
Check status
Incoming packet?

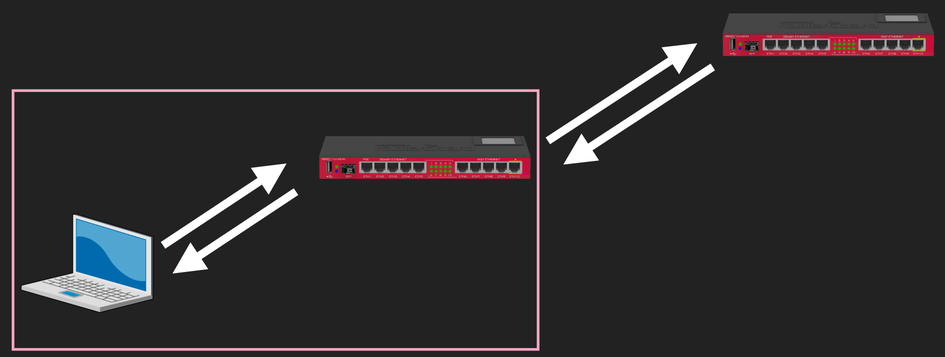

/bndl/ipv6/nova/bin/radvd
Looper::run
nv::ThinRunner::run
nv::ThinRunner::step
Looper
Fds
socket
onMsg..
socket
callback
Yes
Incoming packet?
No

DEVCORE

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

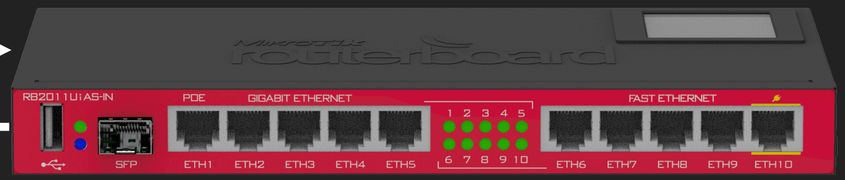
IPv6 SLAAC





ICMPv6

Router Solicitation



ICMPv6

Router Advertisement

ICMPv6

Router Advertisement

DEVCORE

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);
  addNextTimer_407308(a1);
  return 0;
}
```

What we really care

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

DEVCORE

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

DEVCORE

```
int __fastcall addDNS(int a1, tree_base *a2, int lifetime)
{
  // [COLLAPSED LOCAL DECLARATIONS. PRESS KEYPAD CTRL-"+" TO EXPAND]

    PYTE *)a1 = 25;
          a2->length;
                + 2) = 0;
  *(_b
  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

DEVCORE

```
int __fastcall addDNS(int a1, tree_base *a2, int lifetime)
{
  // [COLLAPSED LOCAL DECLARATIONS. PRESS KEYPAD CTRL-"+" TO EXPAND]

    PYTE *)a1 = 25;
          a2->length;
                + 2) = 0;
  *(_b
  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

DEVCORE

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.

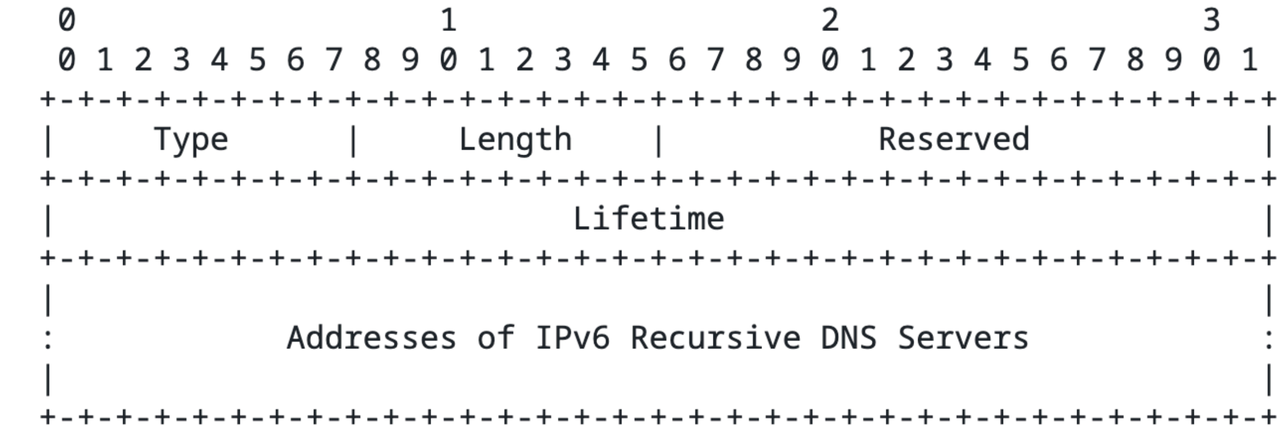


Figure 1: RDNSS Option Format

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.

DEVCORE

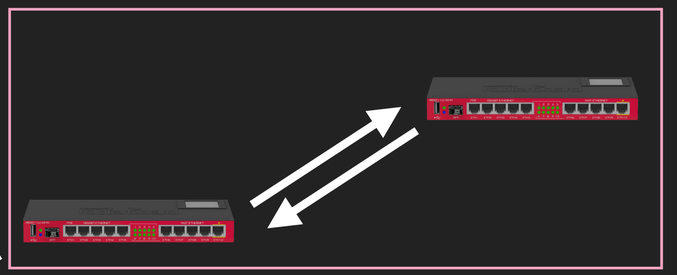
RADVD

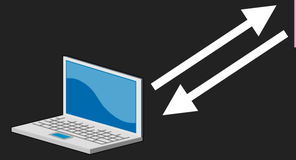
```
if ( v23->enable_advisory )
{
  lifetime = v23->lifetime;
  length = handler_1->DNS_tree.length;
  if ( length )
    length = addDNS((int)&RA_raw[pos], &handler_1->DNS_tree, (lifetime >> 1) + lifetime);
  expire_pos = length + pos;
  v32 = handler_1->expired_DNS_tree.length;
  if ( v32 )
    v32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

DEVCORE

IPv6 SLAAC

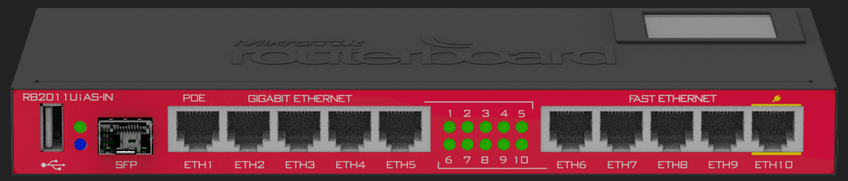


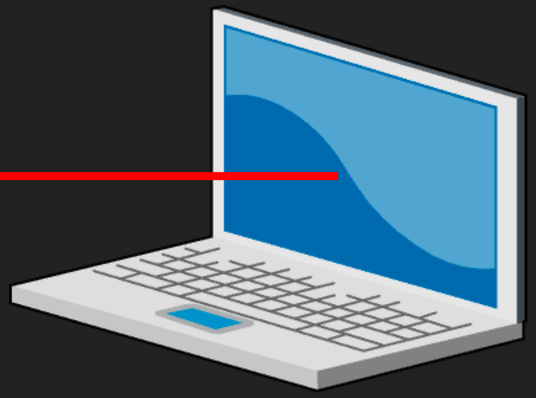


ICMPv6

```
Router Advertisement (RA)
```

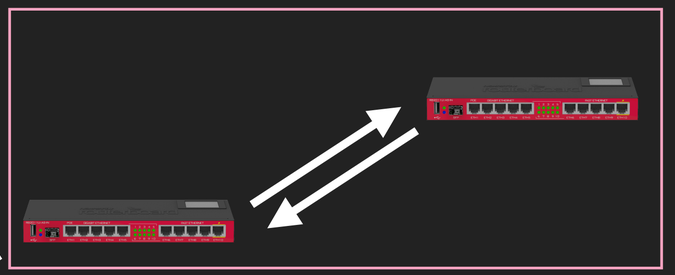
RDNSS: <a big list>





DEVCORE

IPv6 SLAAC

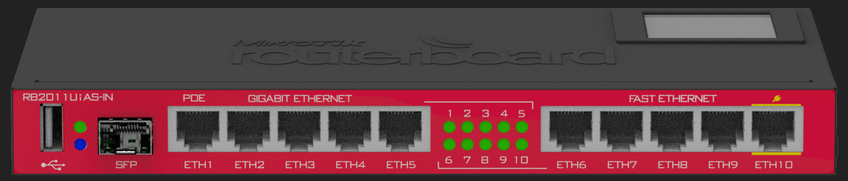


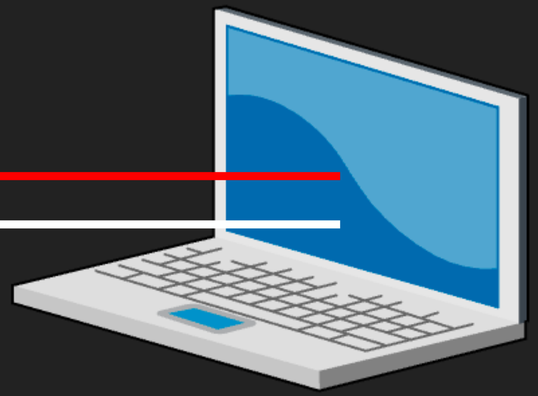


ICMPv6

```
Router Advertisement (RA)
```

RDNSS: <a big list>



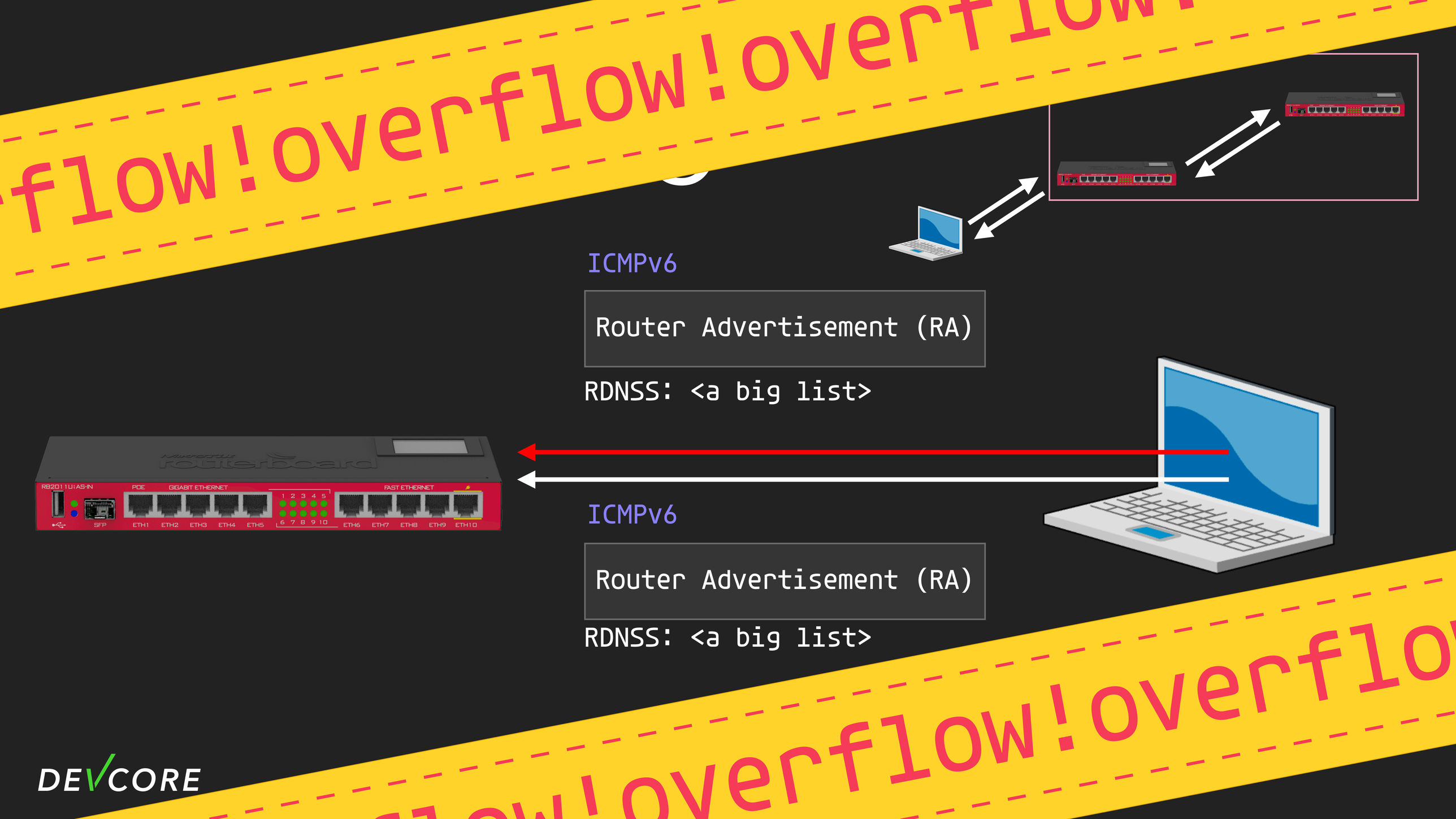


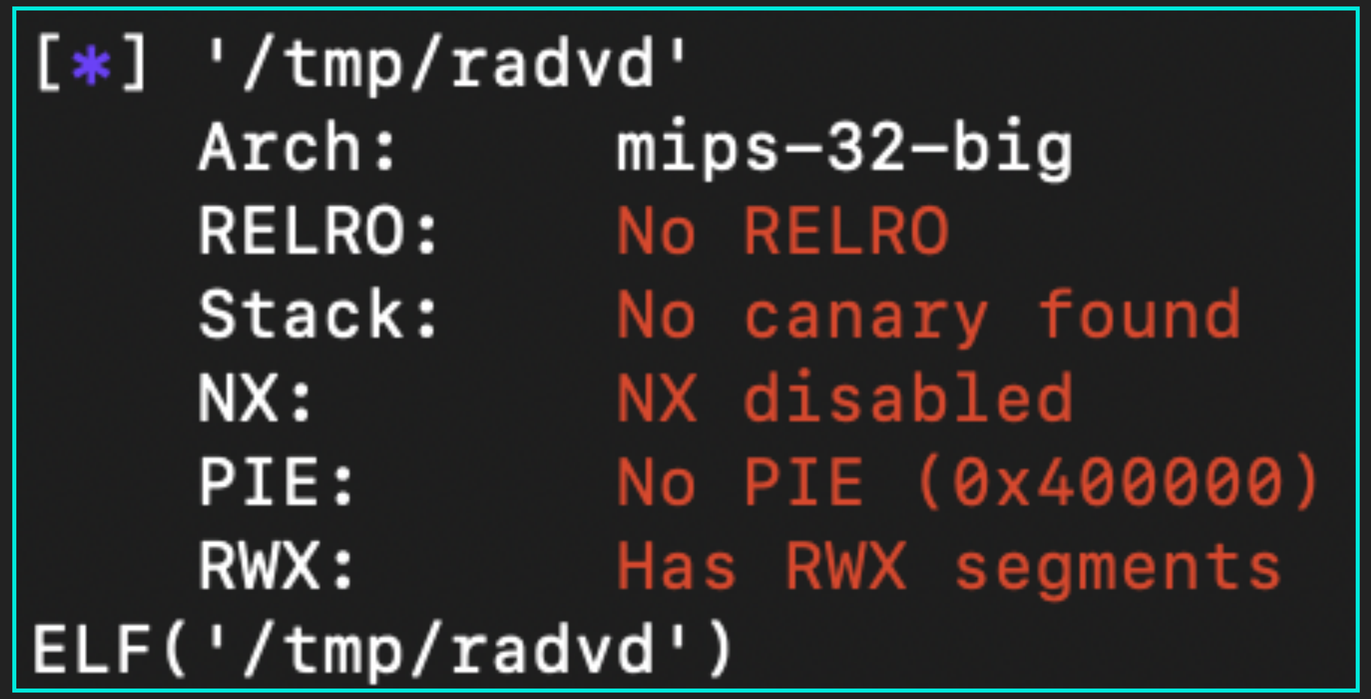
ICMPv6

```
Router Advertisement (RA)
```

RDNSS: <a big list>

DEVCORE


flow!overflow!overflow
ICMPv6
Router Advertisement (RA)
RDNSS: <a big list>
ICMPv6
Router Advertisement (RA)
RDNSS: <a big list>
DEVCORE
overflow!overflo

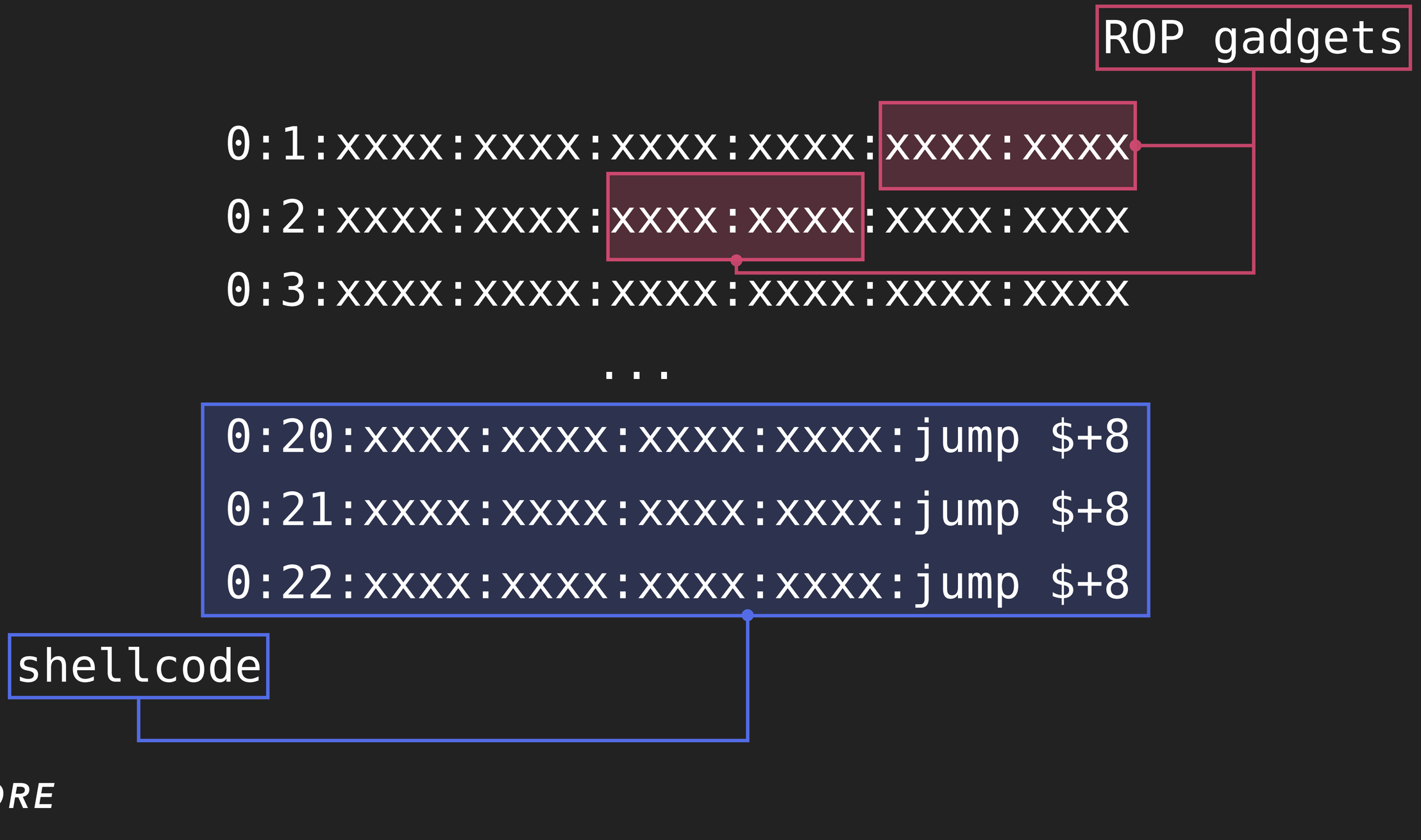

[*] '/tmp/radvd'
Arch: mips-32-big
RELRO: No RELRO
Stack: No canary found
NX: NX disabled
PIE: No PIE (0x400000)
RWX: Has RWX segments
ELF('/tmp/radvd')

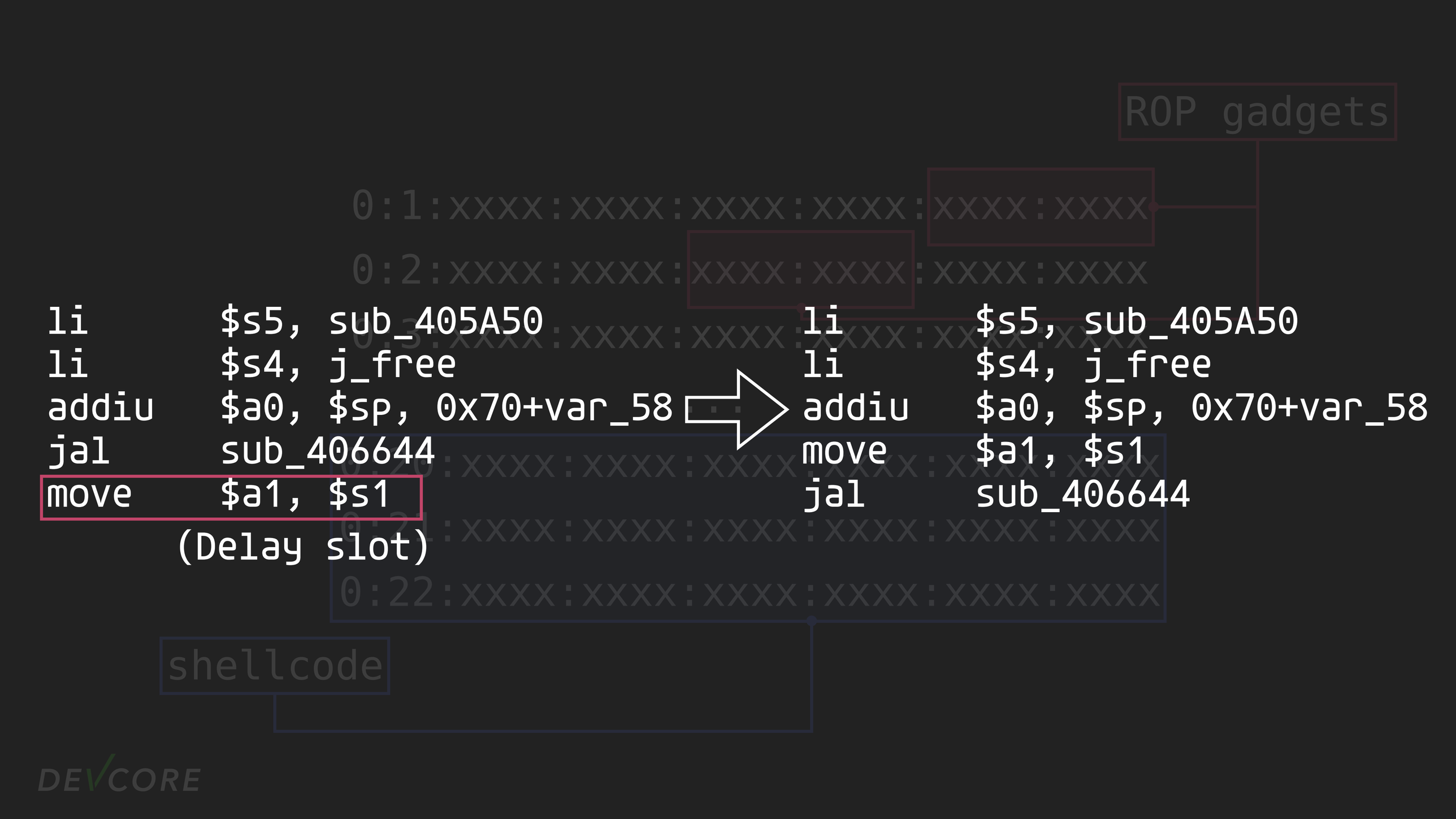
DEVCORE



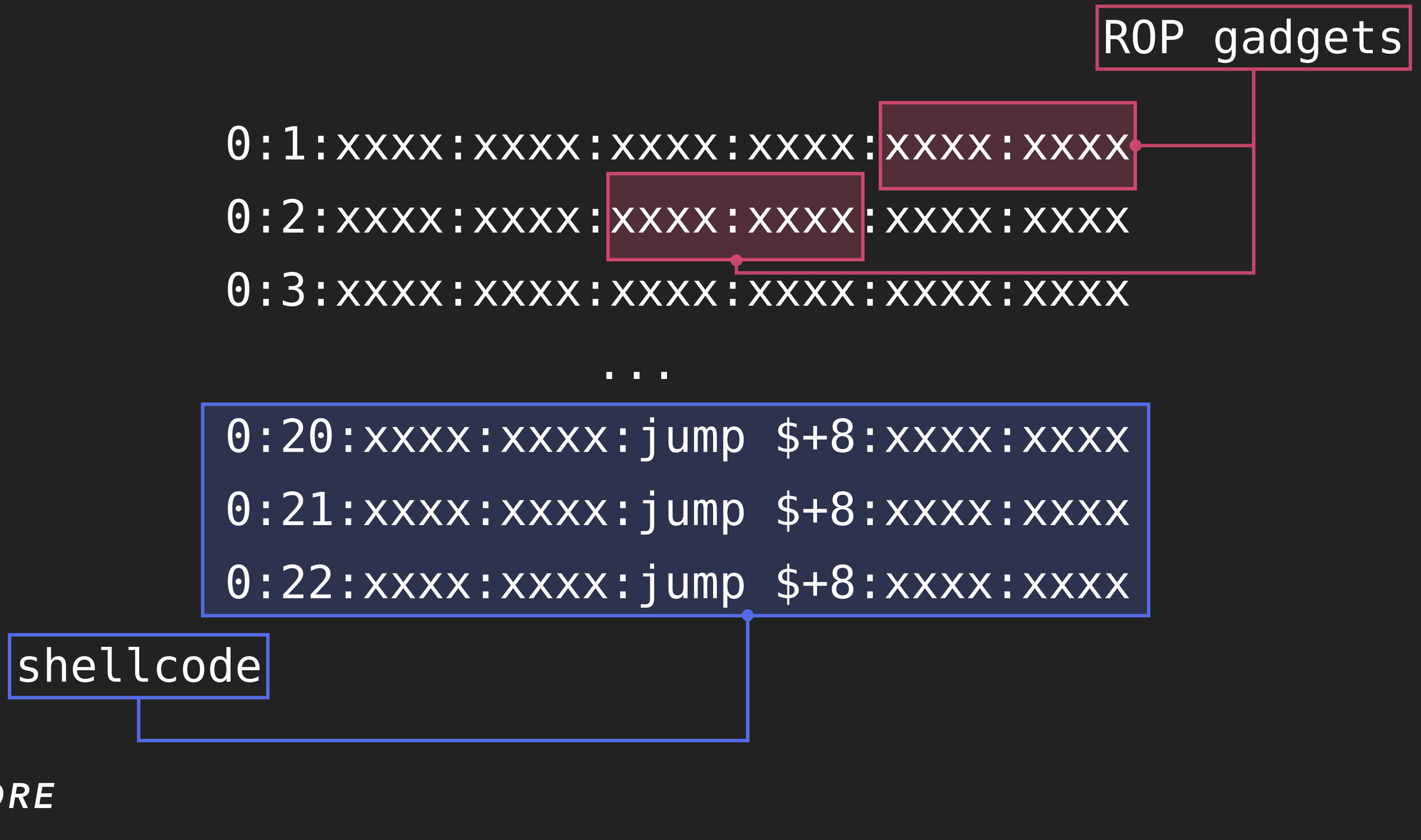
Credit: @__ammar2__

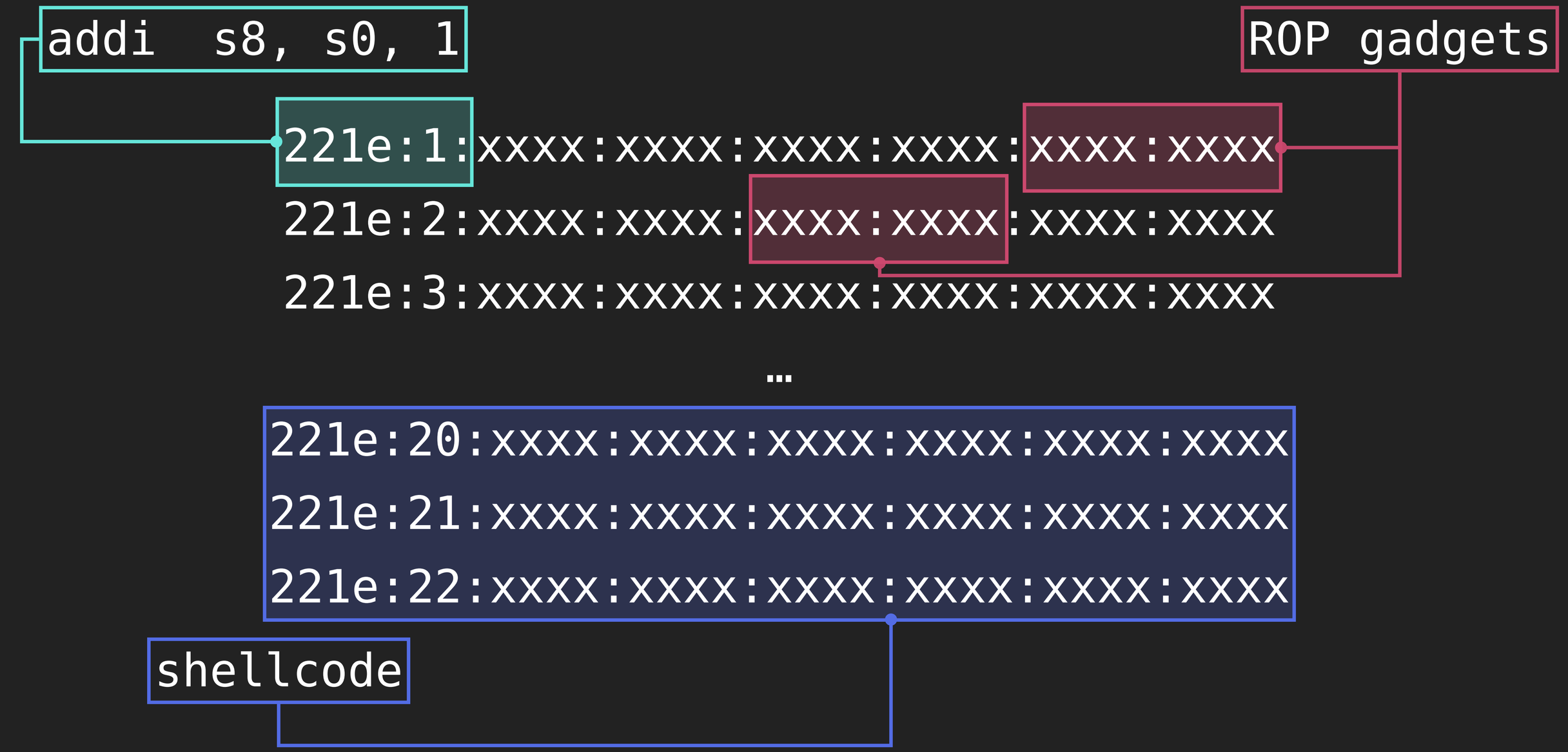
DEVCORE

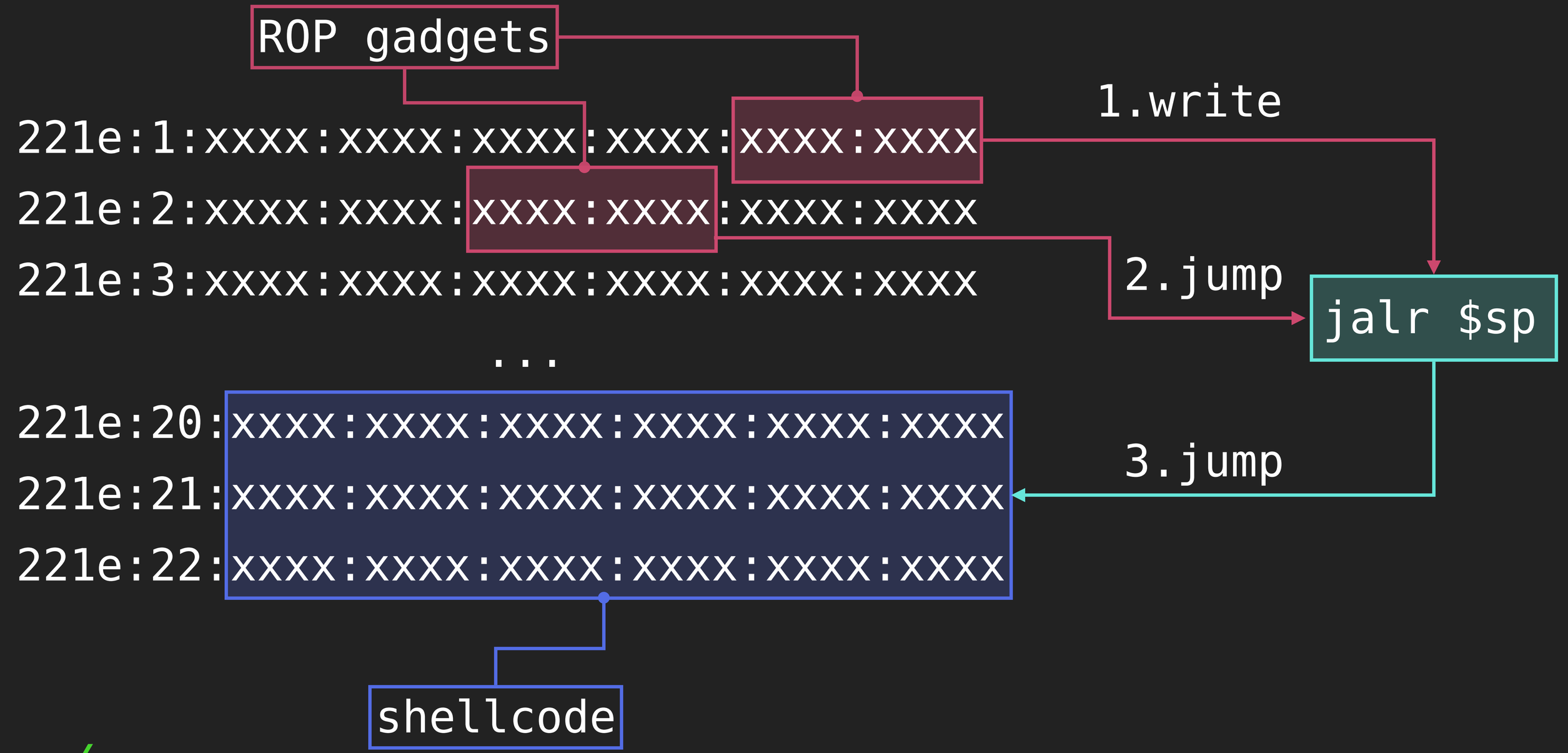

ROP gadgets
0:1:xxxx:xxxx:xxxx:xxxx:xxxx:xxxx
0:2:xxxx:xxxx:xxxx:xxxx:xxxx:xxxx
0:3:xxxx:xxxx:xxxx:xxxx:xxxx:xxxx
...
0:20:xxxx:xxxx:xxxx:xxxx:jump $+8
0:21:xxxx:xxxx:xxxx:xxxx:jump $+8
0:22:xxxx:xxxx:xxxx:xxxx:jump $+8
shellcode


ROP 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:xxxx:xxxx
0:21:xxxx:xxxx:xxxx:xxxx:xxxx:xxxx
0:22:xxxx:xxxx:xxxx:xxxx:xxxx:xxxx
shellcode
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

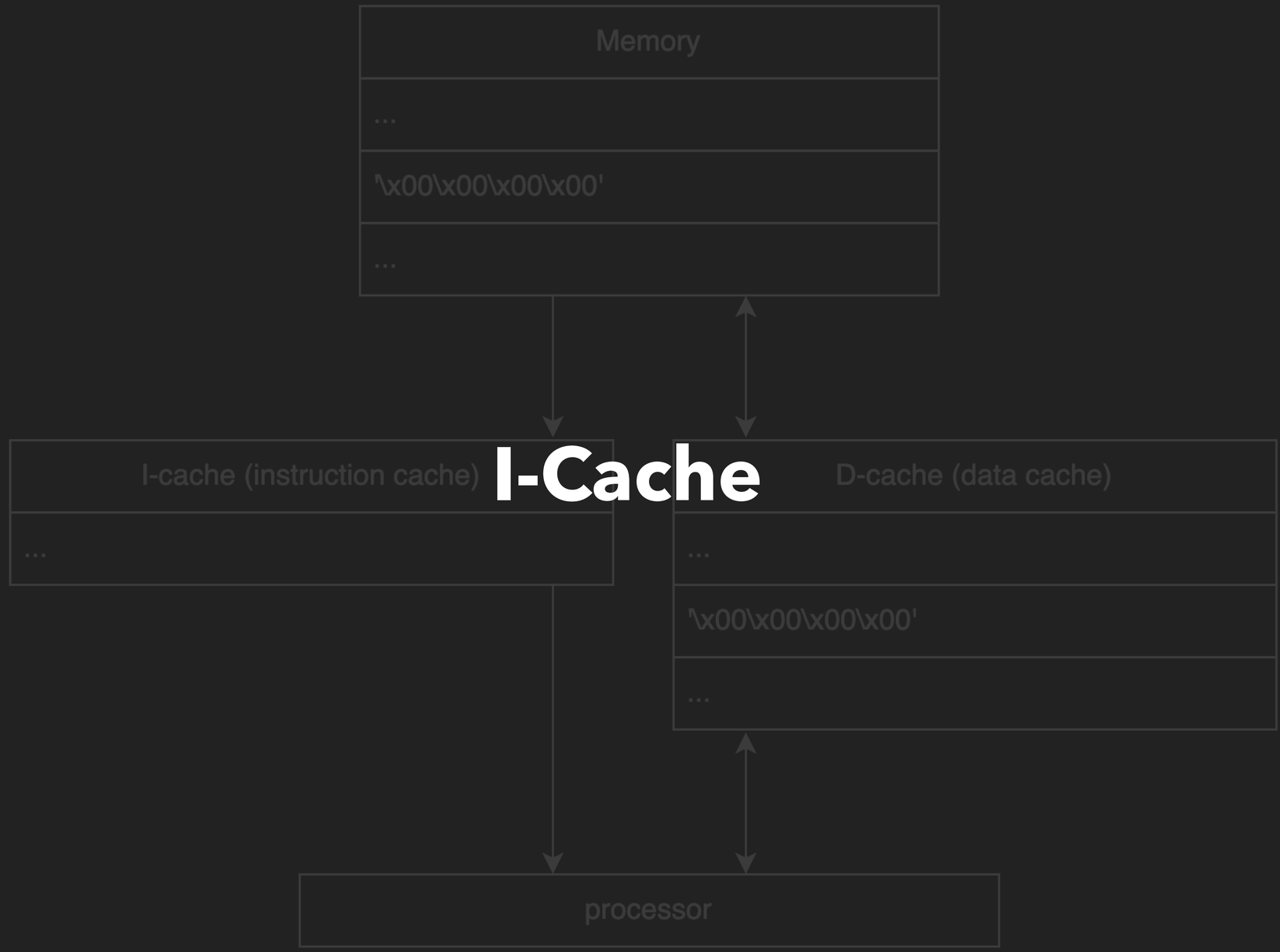
DEVCORE


ROP 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

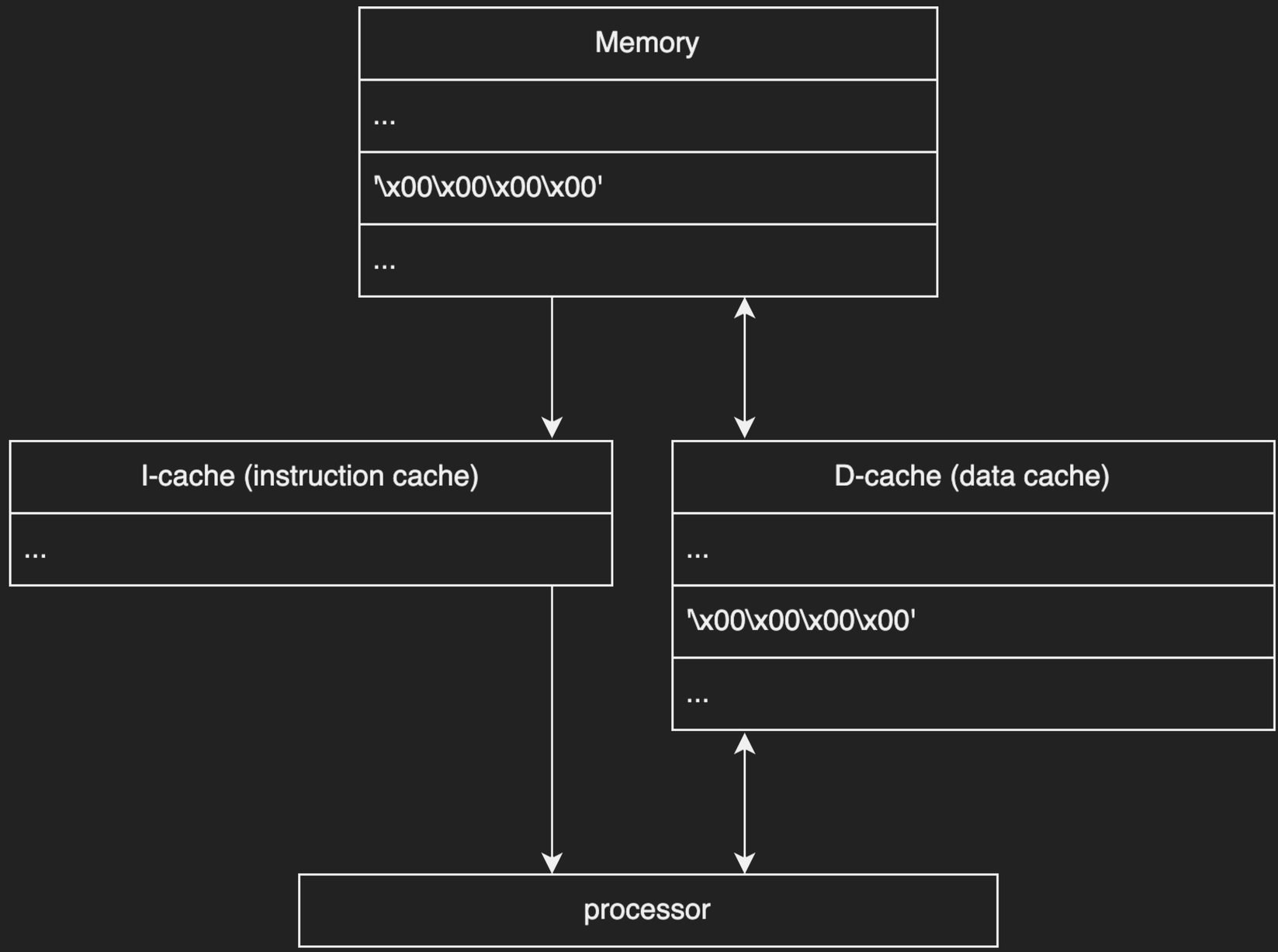

addi s8, s0, 1
ROP gadgets
221e:1:xxxx:xxxx:xxxx:xxxx:xxxx:xxxx
221e:2:xxxx:xxxx:xxxx:xxxx:xxxx:xxxx
221e:3:xxxx:xxxx:xxxx:xxxx:xxxx:xxxx
...
221e:20:xxxx:xxxx:xxxx:xxxx:xxxx:xxxx
221e:21:xxxx:xxxx:xxxx:xxxx:xxxx:xxxx
221e:22:xxxx:xxxx:xxxx:xxxx:xxxx:xxxx
shellcode


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
shellcode
1.write
2.jump
jalr $sp
3.jump

DEVCORE

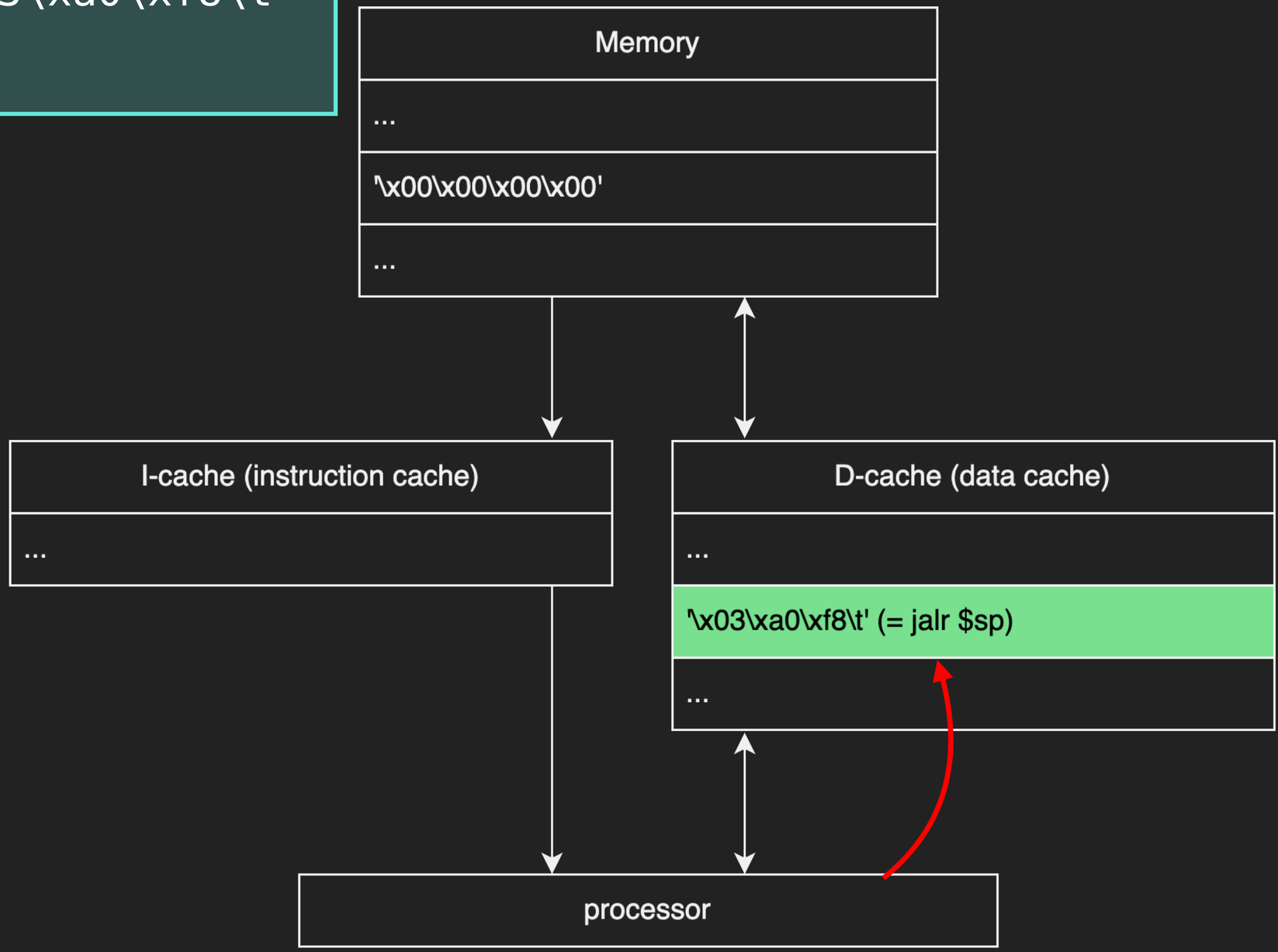

Memory
...
'\x00\x00\x00\x00'
...
I-cache (instruction cache)
...
D-cache (data cache)
...
'\x00\x00\x00\x00'
...
processor

I-Cache


Memory
...
'\x00\x00\x00\x00'
...
I-cache (instruction cache)
...
D-cache (data cache)
...
'\x00\x00\x00\x00'
...
processor

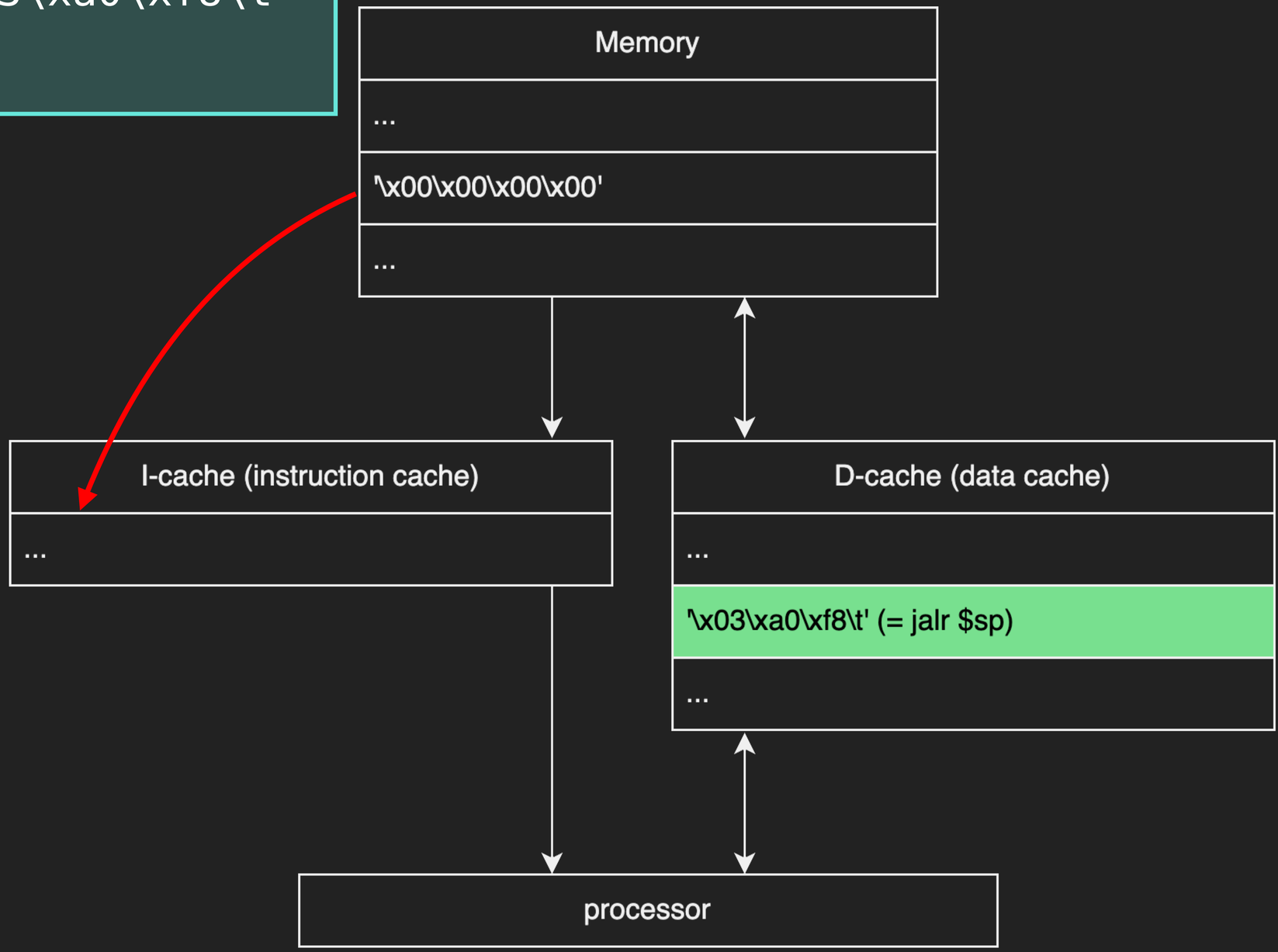
DEVCORE

```
write adr,’\x03\xa0\xf8\t’
```



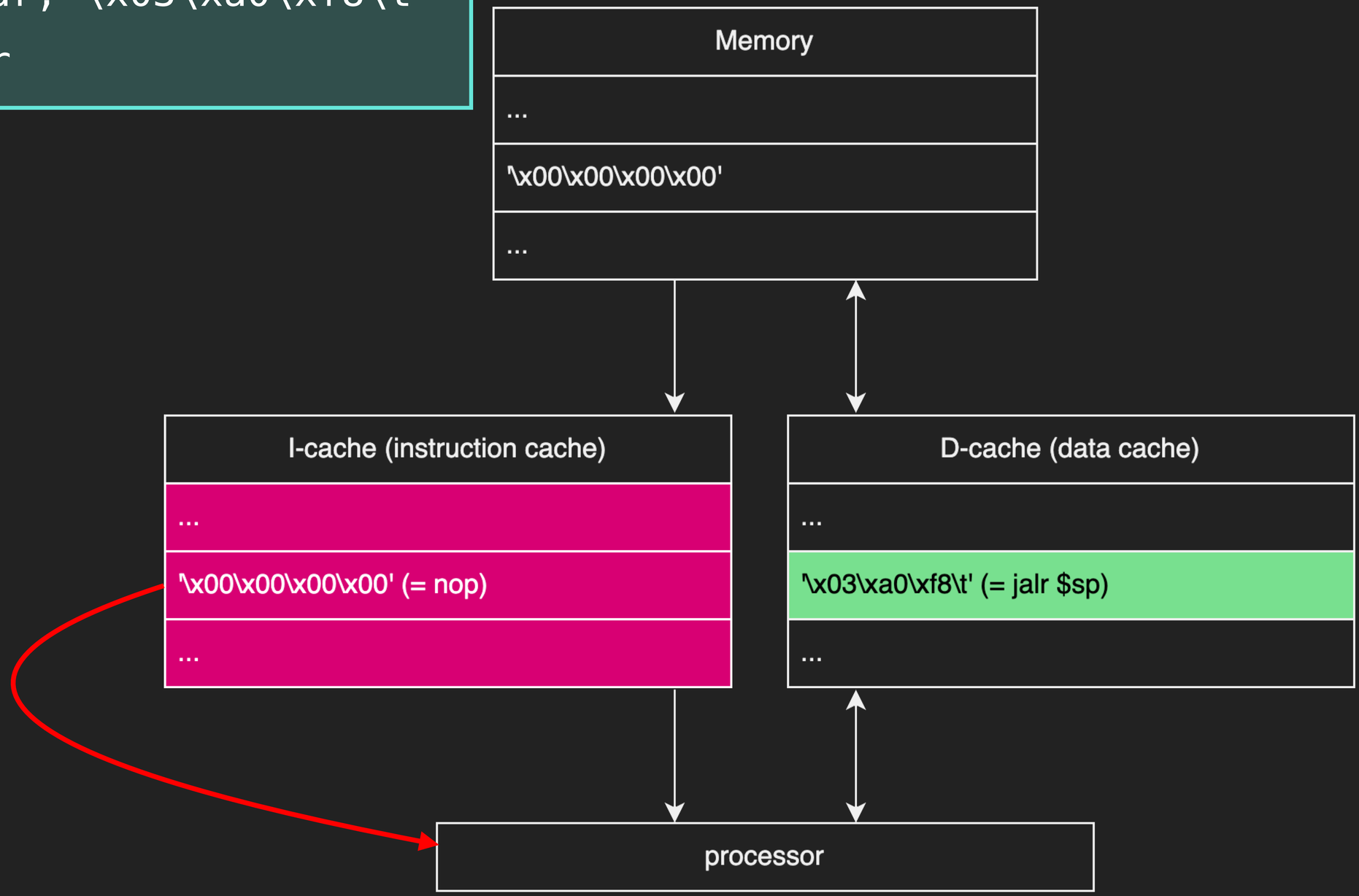
DEVCORE

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



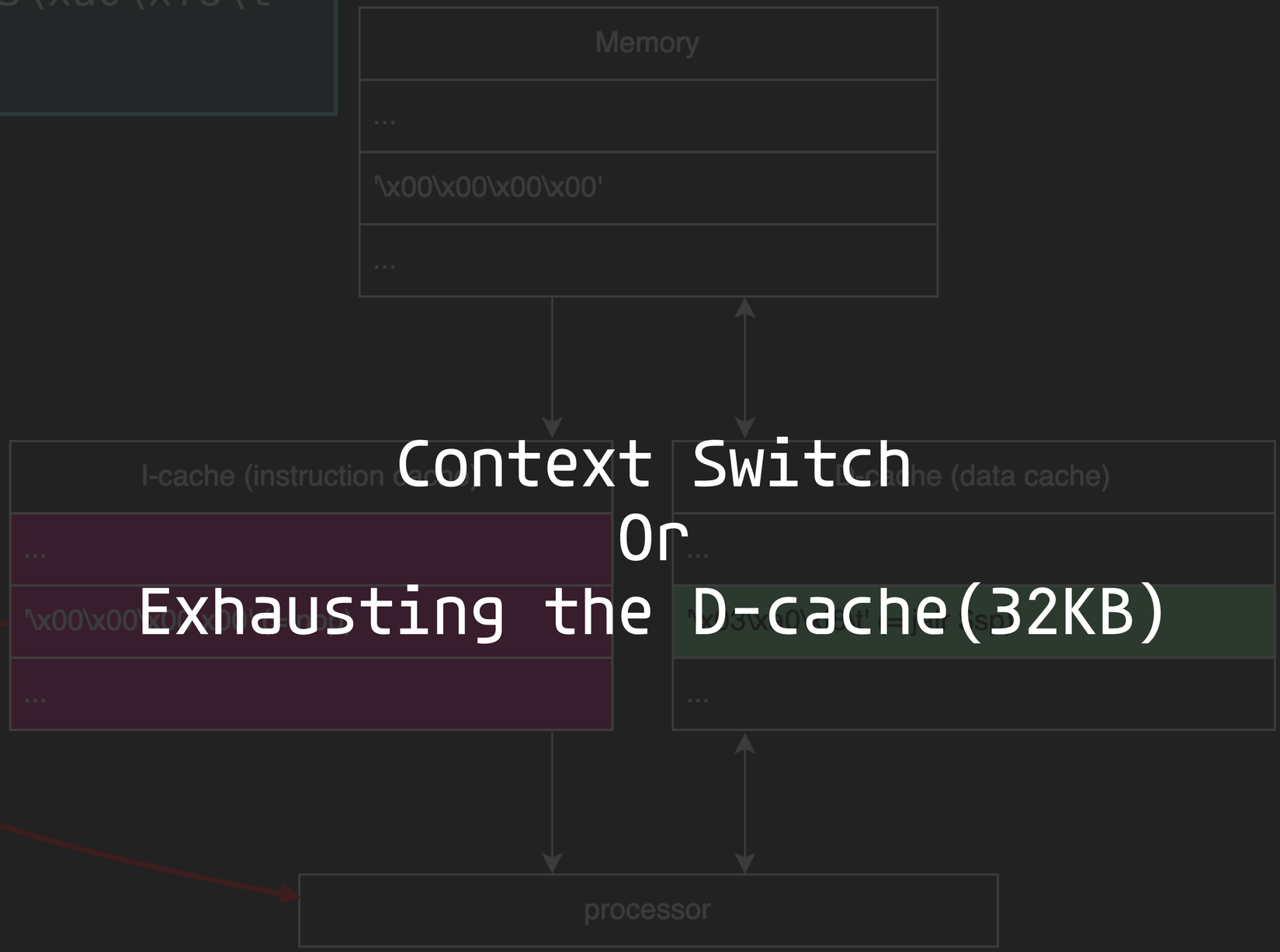
DEVCORE

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



DEVCORE

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



Context Switch
Or
Exhausting the D-cache(32KB)

DEVCORE

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

```
[pwndbg> vmmap
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
```

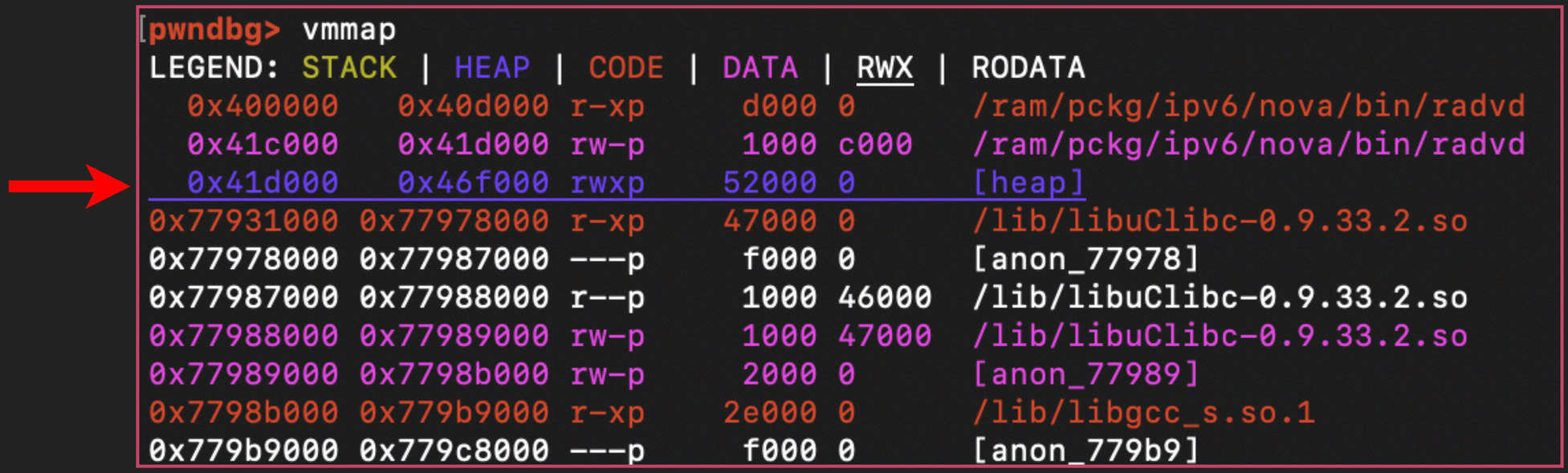
DEVCORE

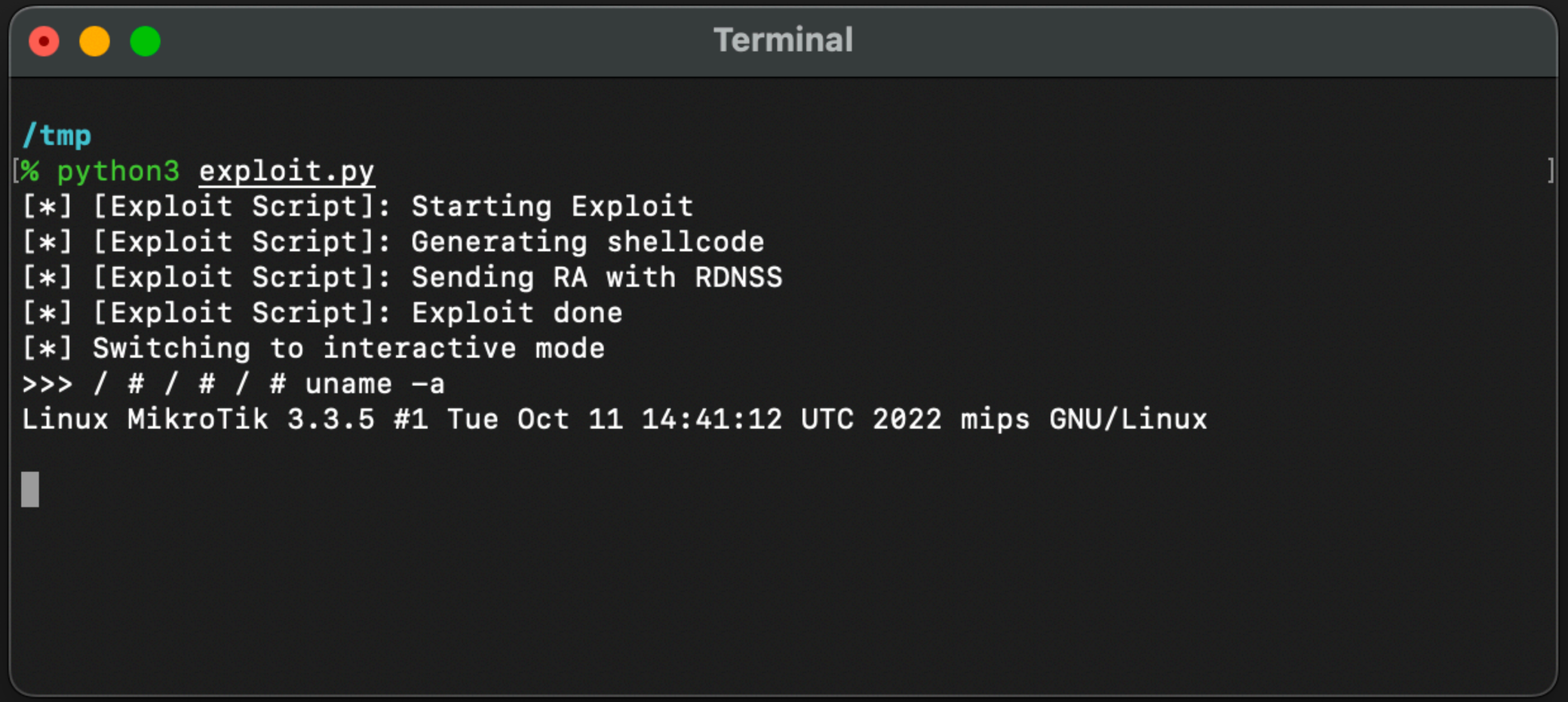
RADVD

```
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
```



DEVCORE


[pwndbg> vmmap
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

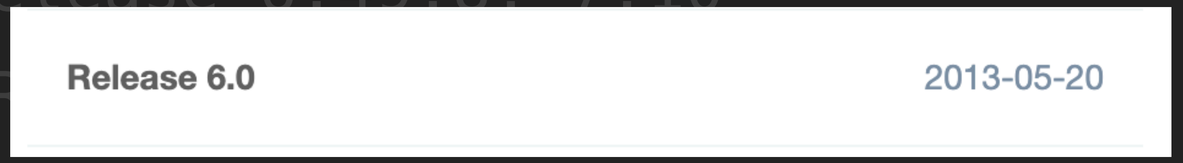
DEVCORE

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



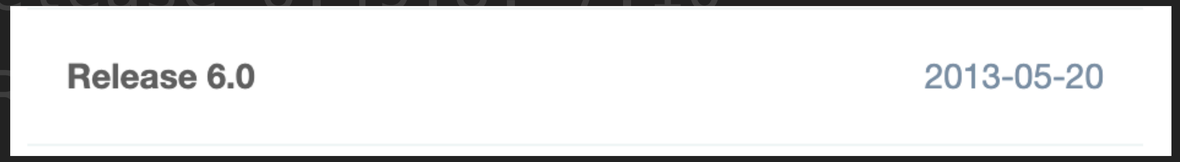
DEVCORE

CVE-2023-32|54

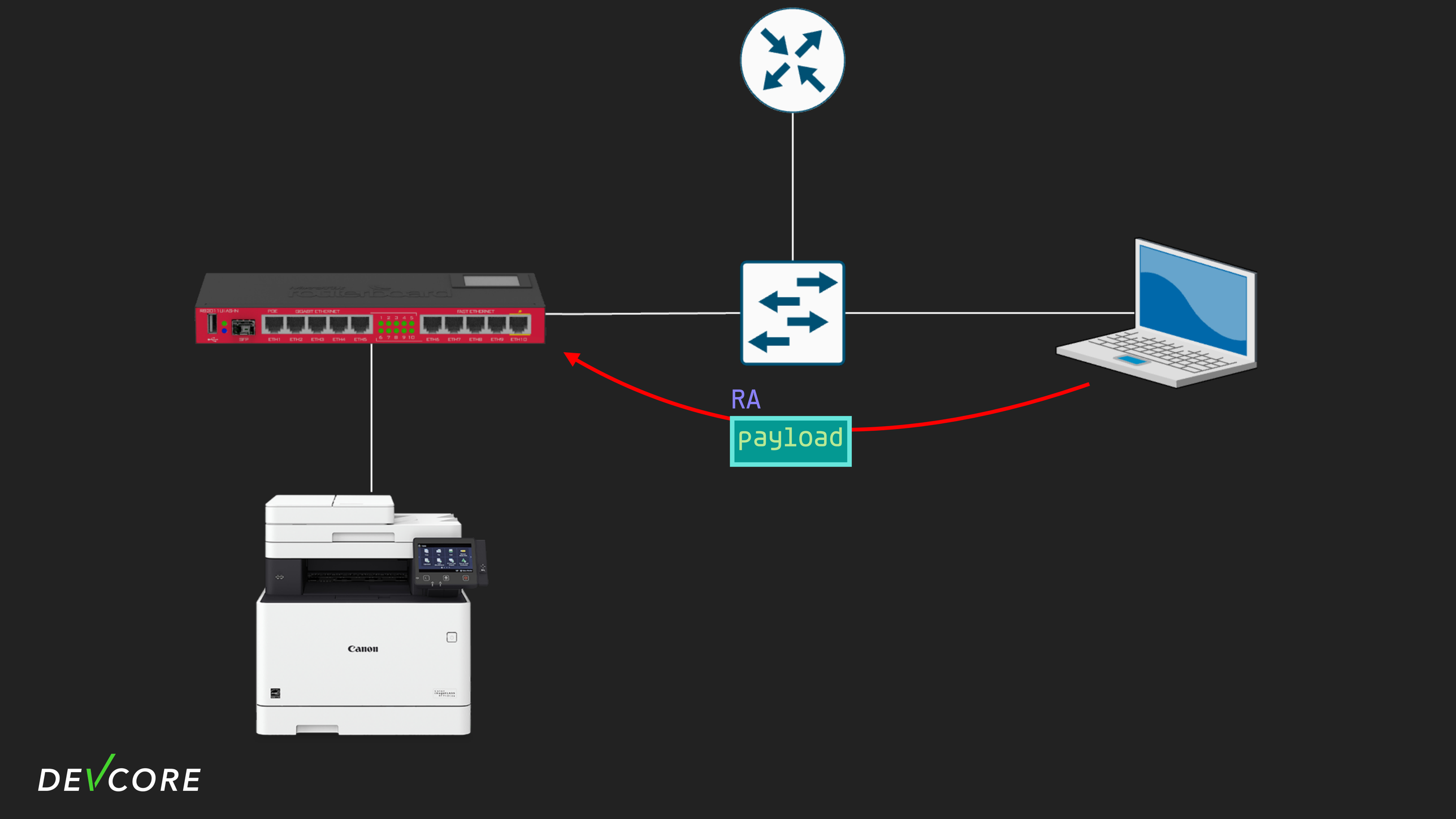
- Fixed at:
  - Long-term Release 6.48.7
  - Stable Release 6.49.8, 7.10
  - Testing R
- The vulnerable code has existed at least since v6.0

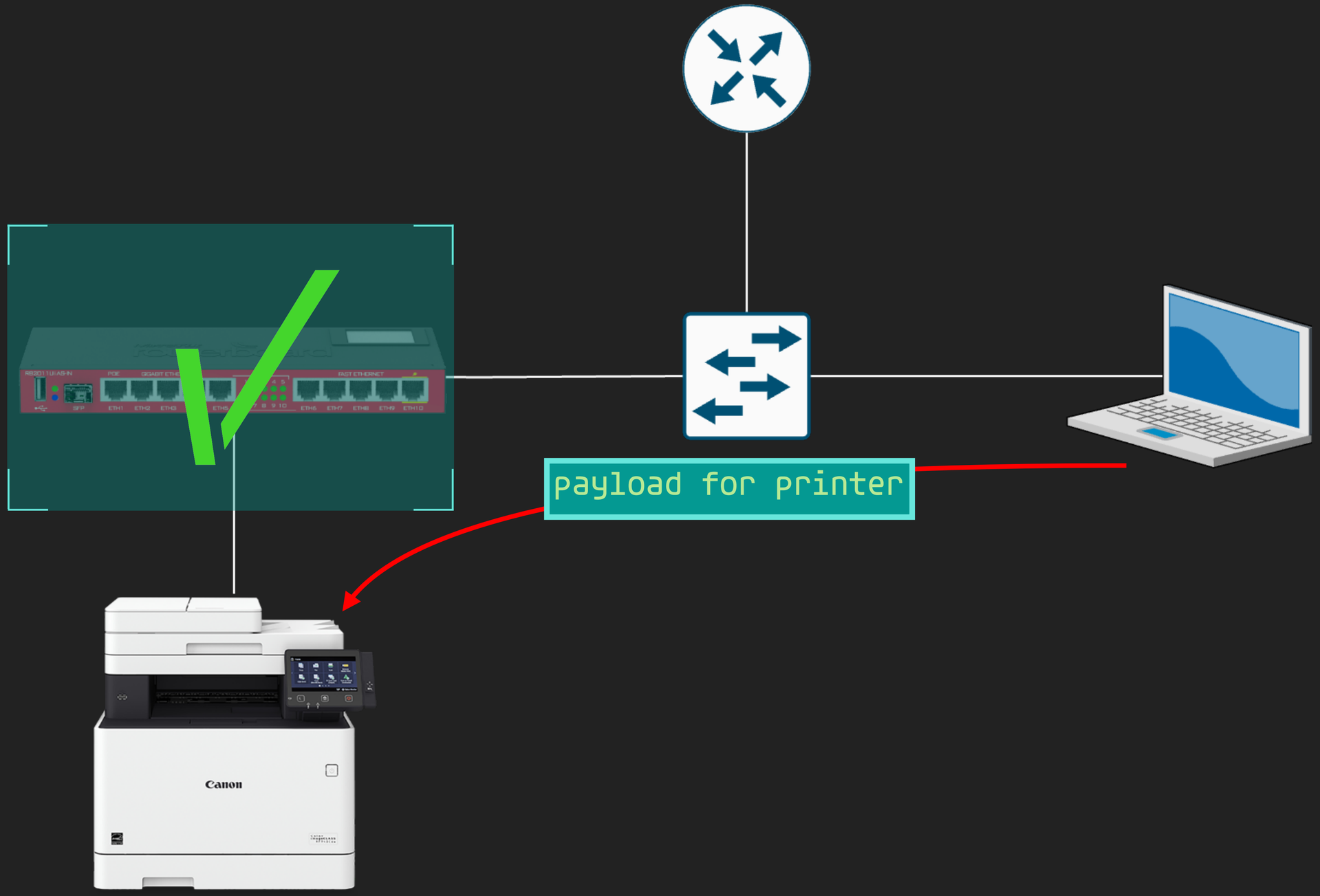
No one with sanity

would like to dive into the details of Nova Binary

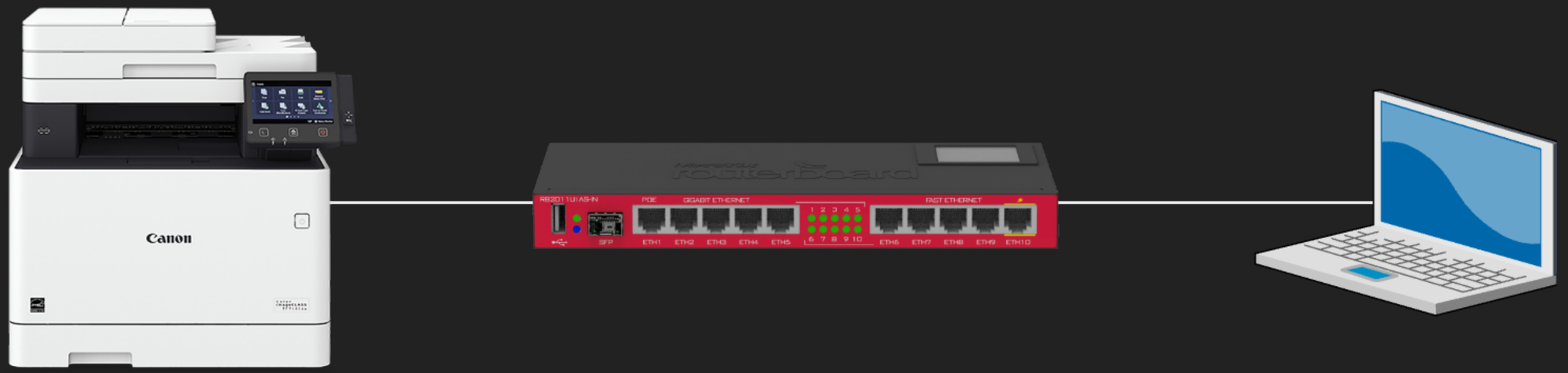


Q.E.D.

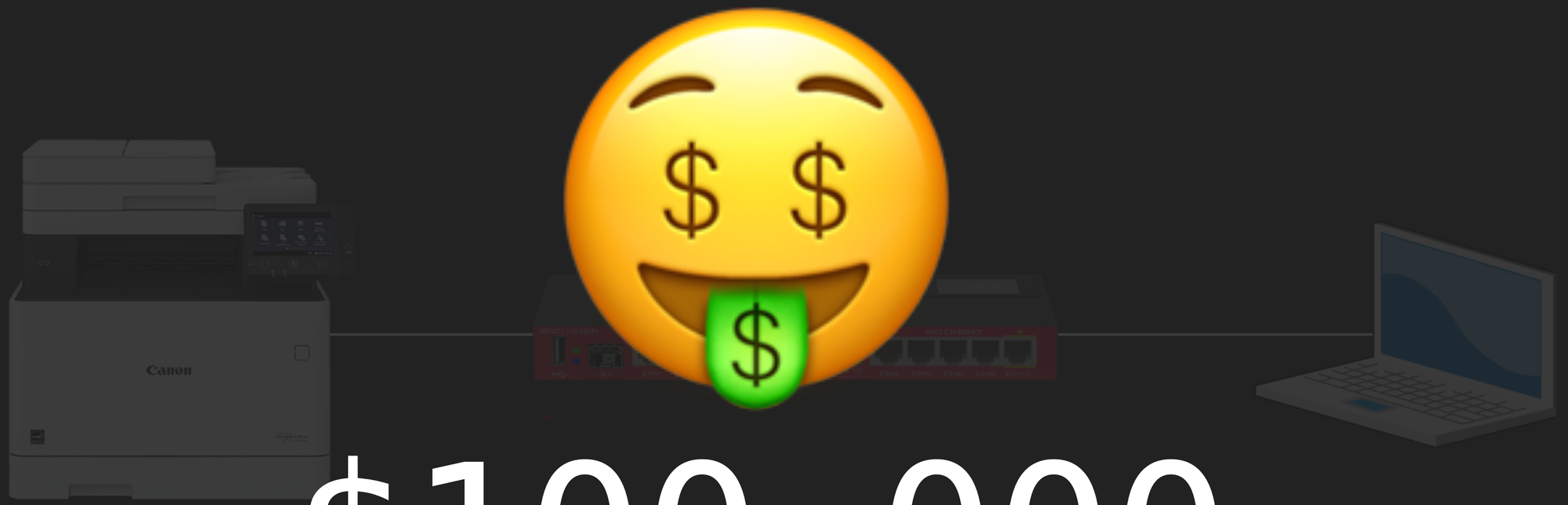

RA
payload
Canon
DEVCORE



DEVCORE


Canon
RouterBoard
RB2011UiAS-IN
POE
GIGABIT ETHERNET
FAST ETHERNET
SFP
ETH1
ETH2
ETH3
ETH4
ETH5
ETH6
ETH7
ETH8
ETH9
ETH10

DEVCORE



$100,000

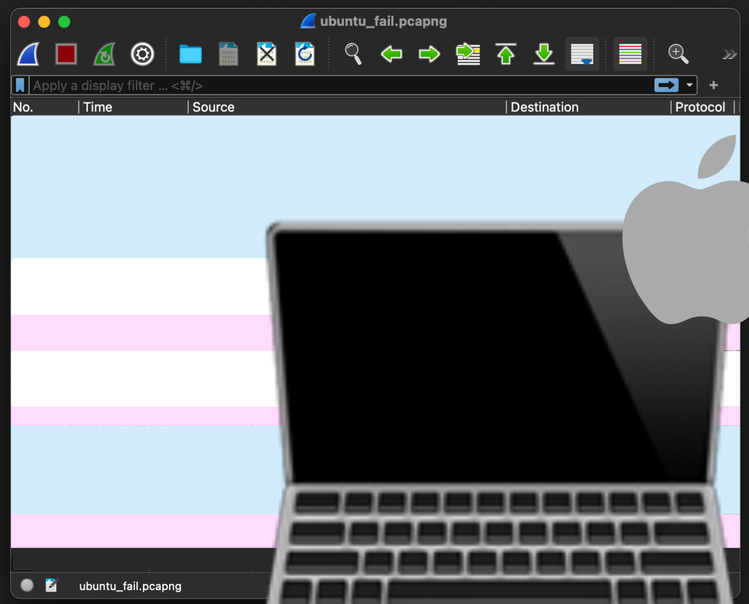
But...

But

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



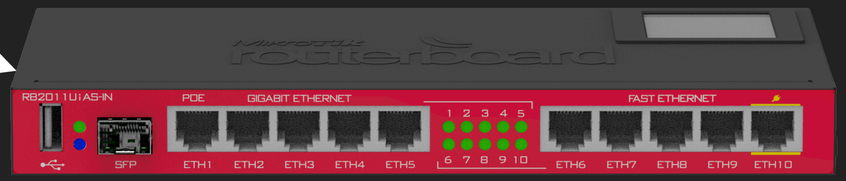
DEVCORE



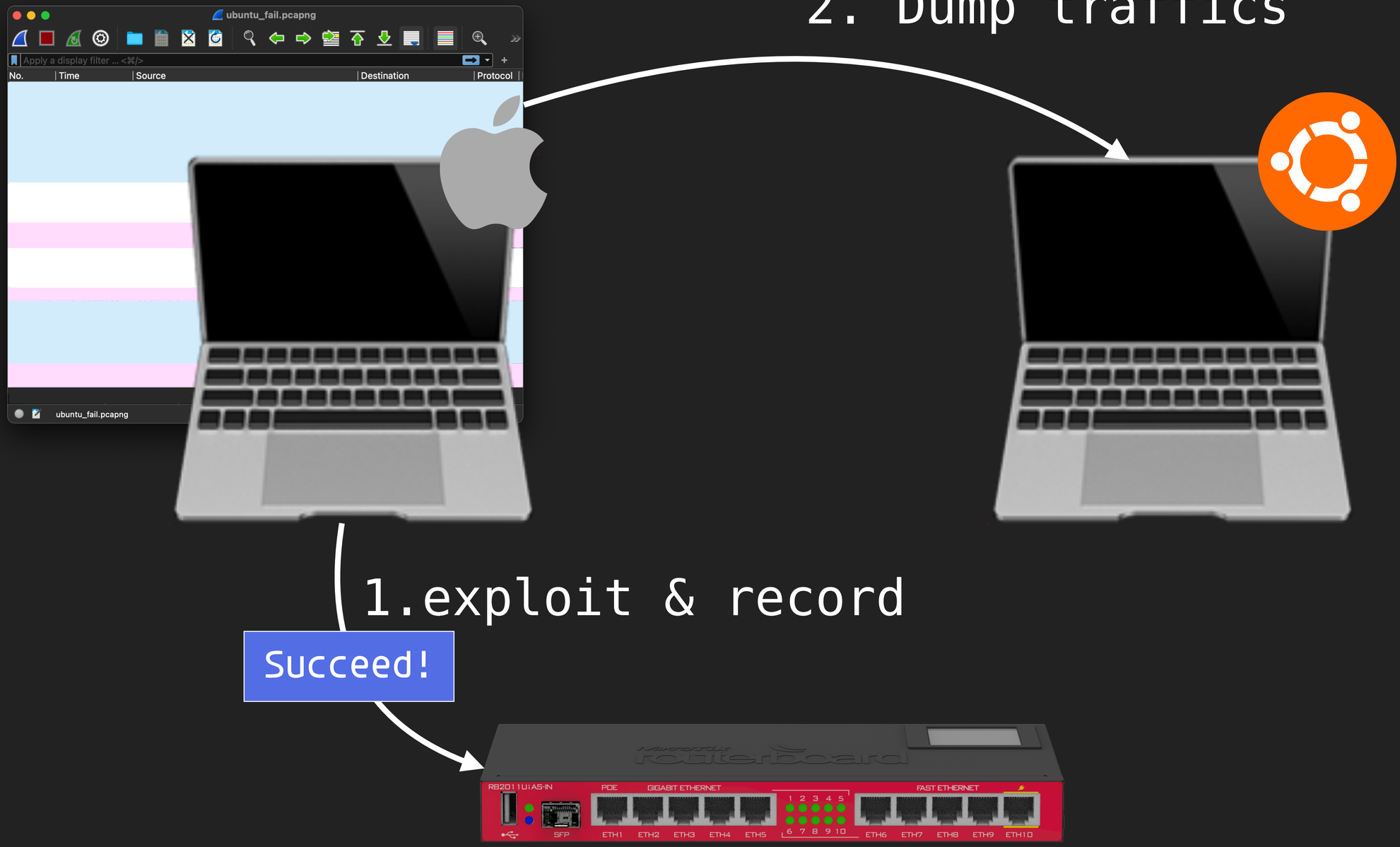


1.exploit & record

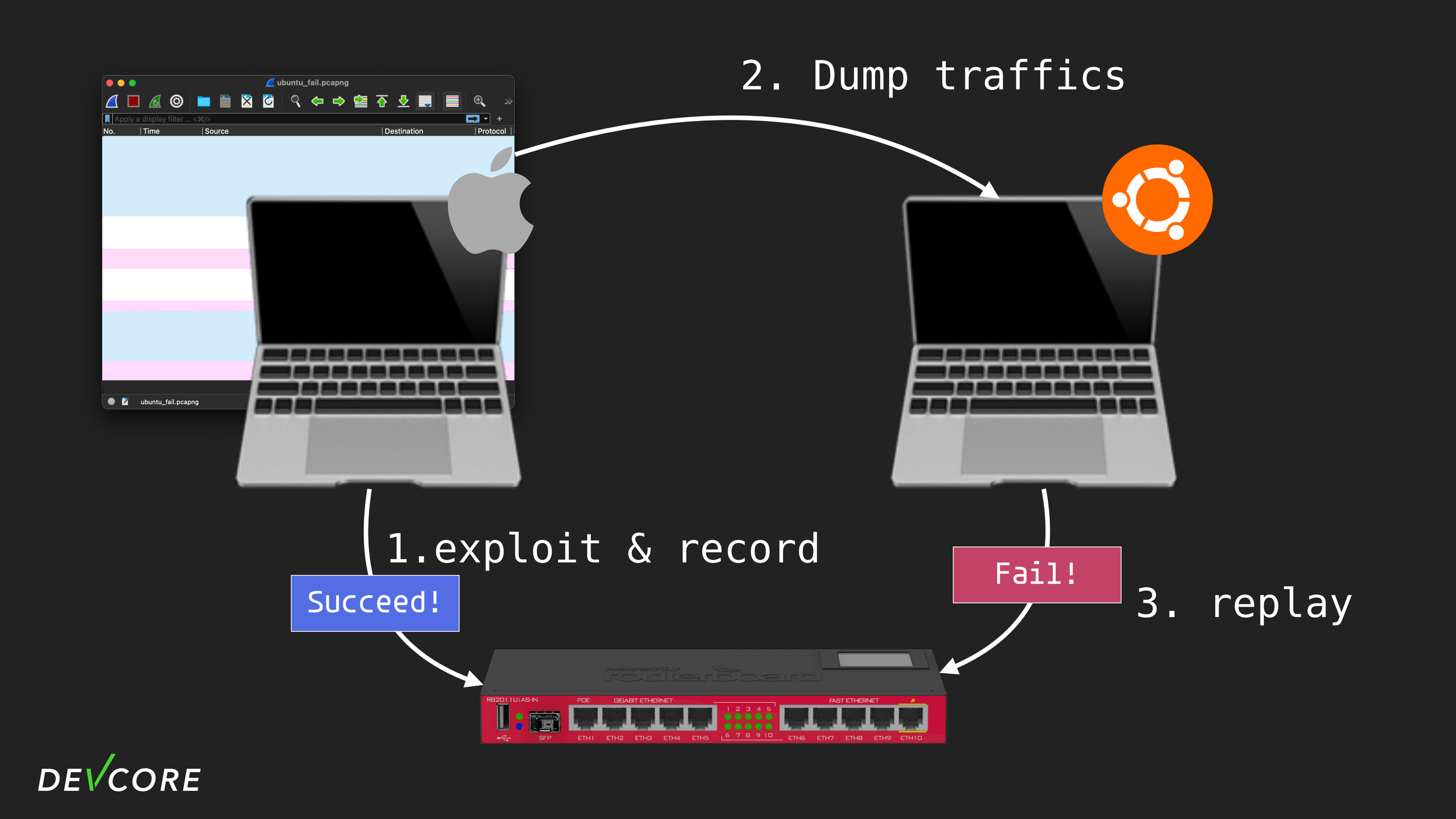
Succeed!



DEVCORE

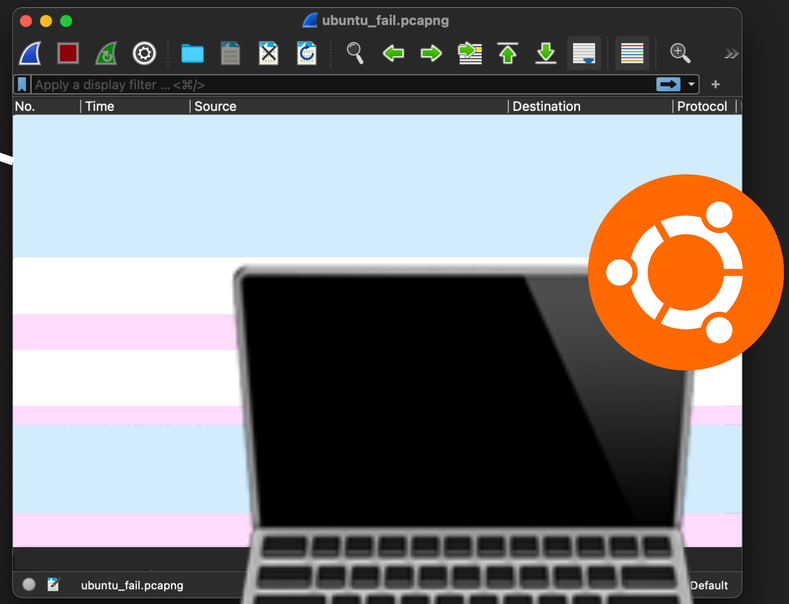

ubuntu_fail.pcapng
Apply a display filter ... <⌘/>
No.
Time
Source
Destination
Protocol
ubuntu_fail.pcapng
2. Dump traffics
1.exploit & record
Succeed!
RB2011UiAS-IN
PoE
GIGABIT ETHERNET
FAST ETHERNET
SFP
ETH1
ETH2
ETH3
ETH4
ETH5
ETH6
ETH7
ETH8
ETH9
ETH10

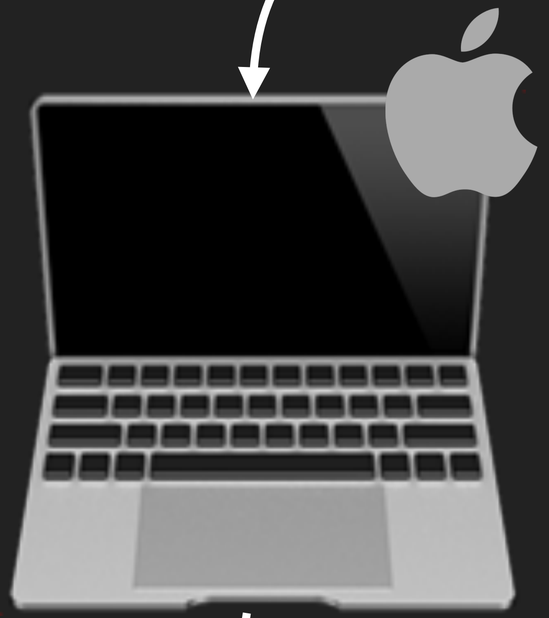
DEVCORE


ubuntu_fail.pcapng
Apply a display filter ... <⌘/>
No.
Time
Source
Destination
Protocol
ubuntu_fail.pcapng
2. Dump traffics
1.exploit & record
Succeed!
Fail!
3. replay
RB2011UiAS-IN
PoE
GIGABIT ETHERNET
FAST ETHERNET
SFP
ETH1
ETH2
ETH3
ETH4
ETH5
ETH6
ETH7
ETH8
ETH9
ETH10

DEVCORE

2. Dump traffics

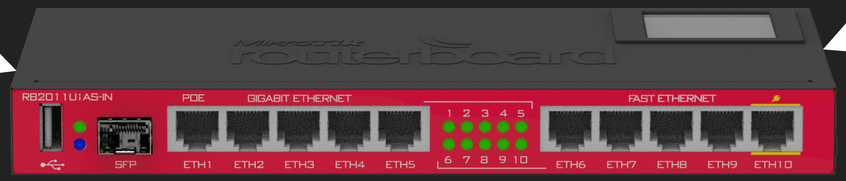




1.exploit & record

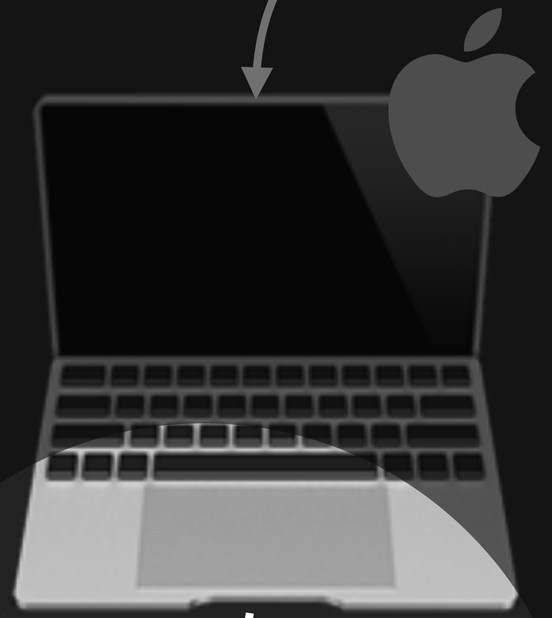
Fail!

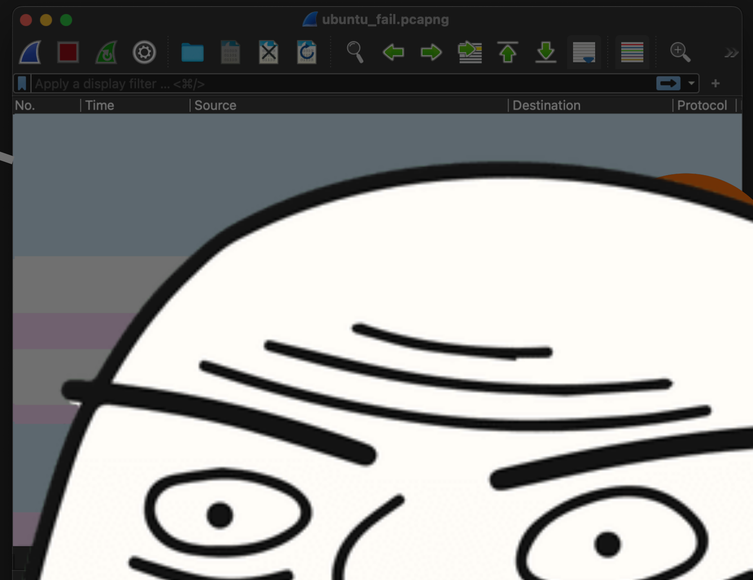
3. replay



DEVCORE

2. Dump traffics





1.explo

Succeed!

3. replay



DEVCORE

Guess:

An OS reorders the packets

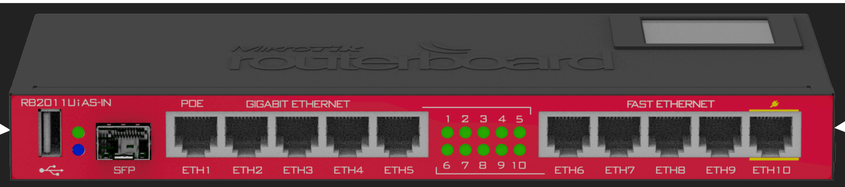


exploit

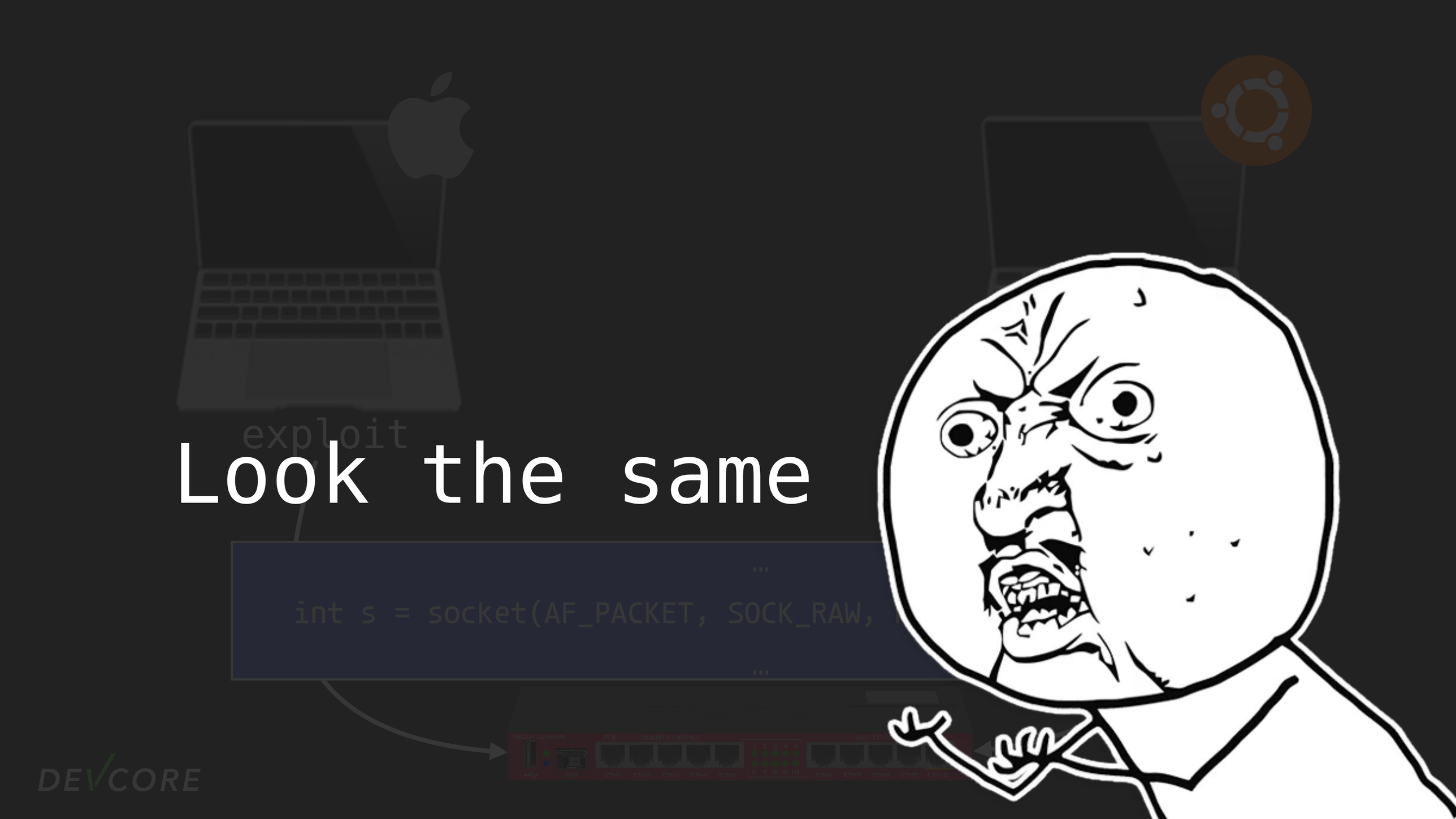


exploit

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



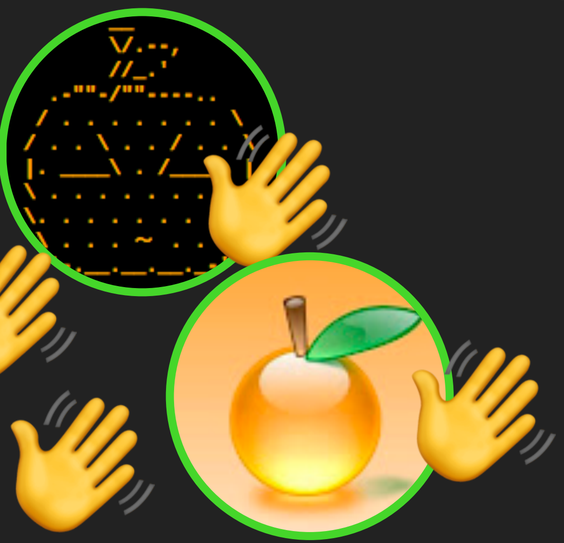
DEVCORE



Look the same

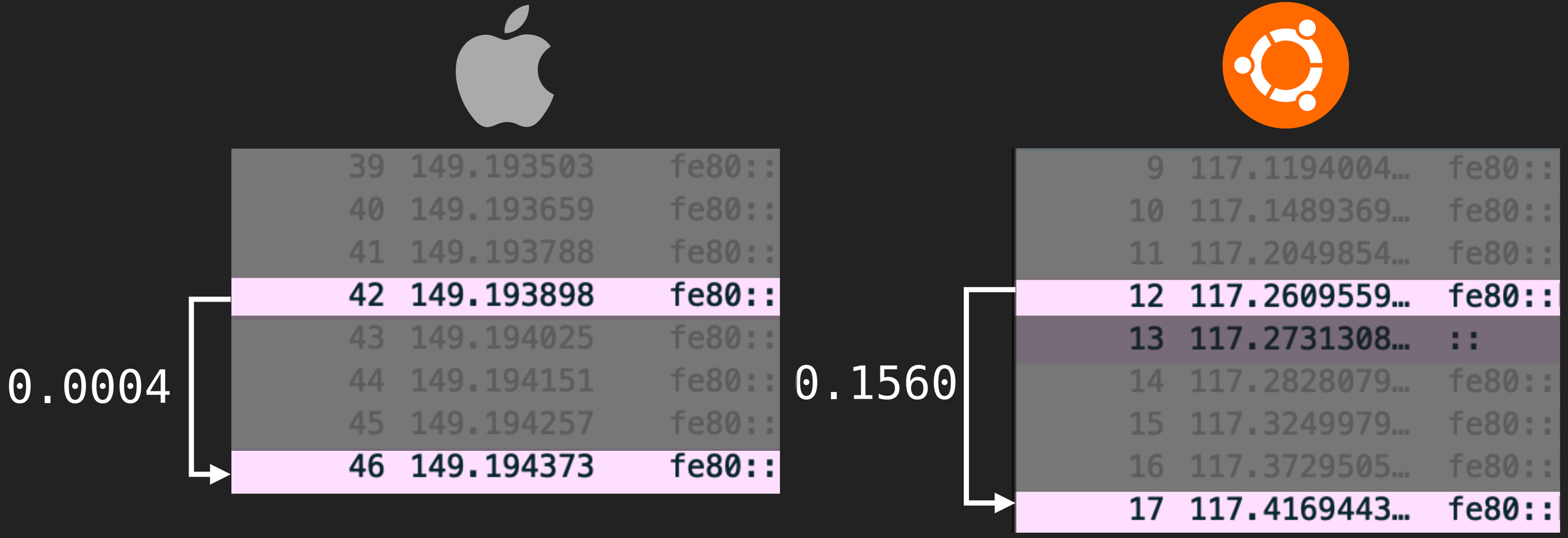
DEVCORE







DEVCORE



390x!!

DEVCORE


poll fds,
run callbacks
nv::Looper
tick
ThinRunner::runTimer

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;
```

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);
```

DEVCORE

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

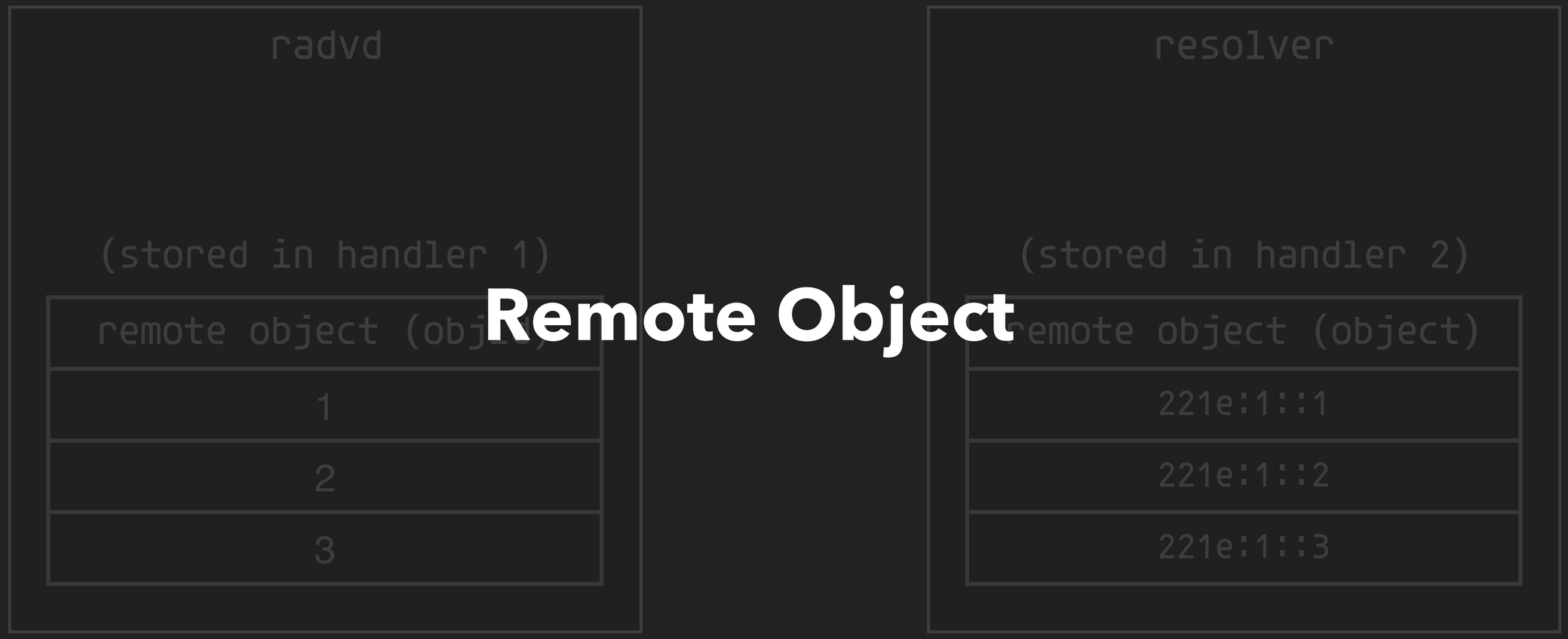


In callback function (recv RA from WAN)

```
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))
```

DEVCORE



Remote Object

radvd

(stored in handler 1)

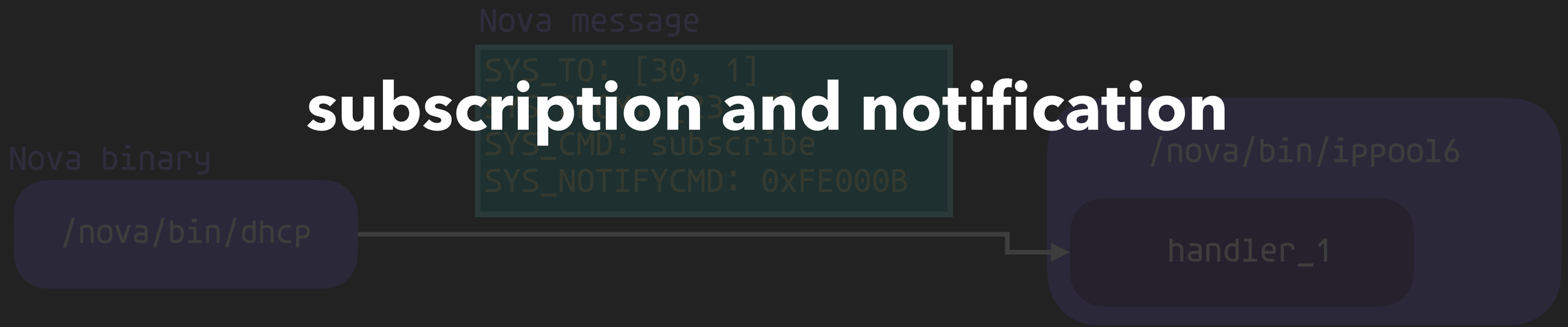
remote object (objid)
1
2
3

resolver

(stored in handler 2)

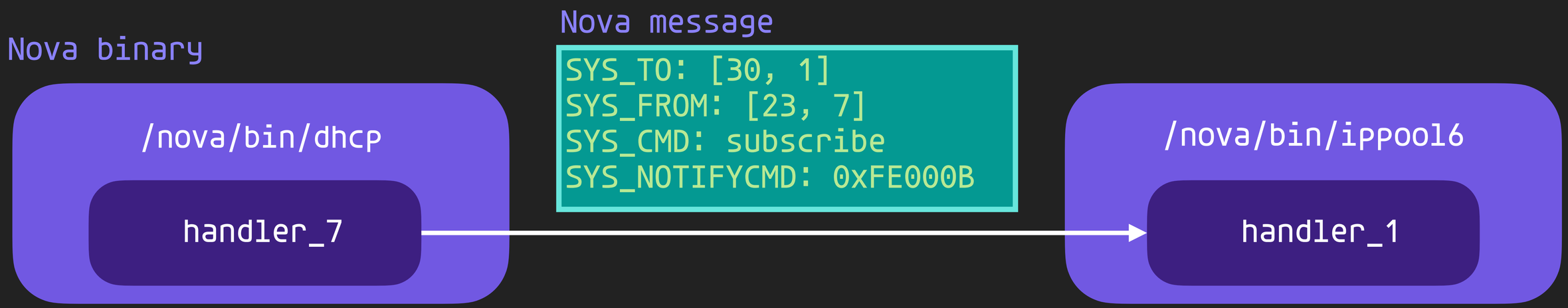
remote object (object)
221e:1::1
221e:1::2
221e:1::3

Subscription



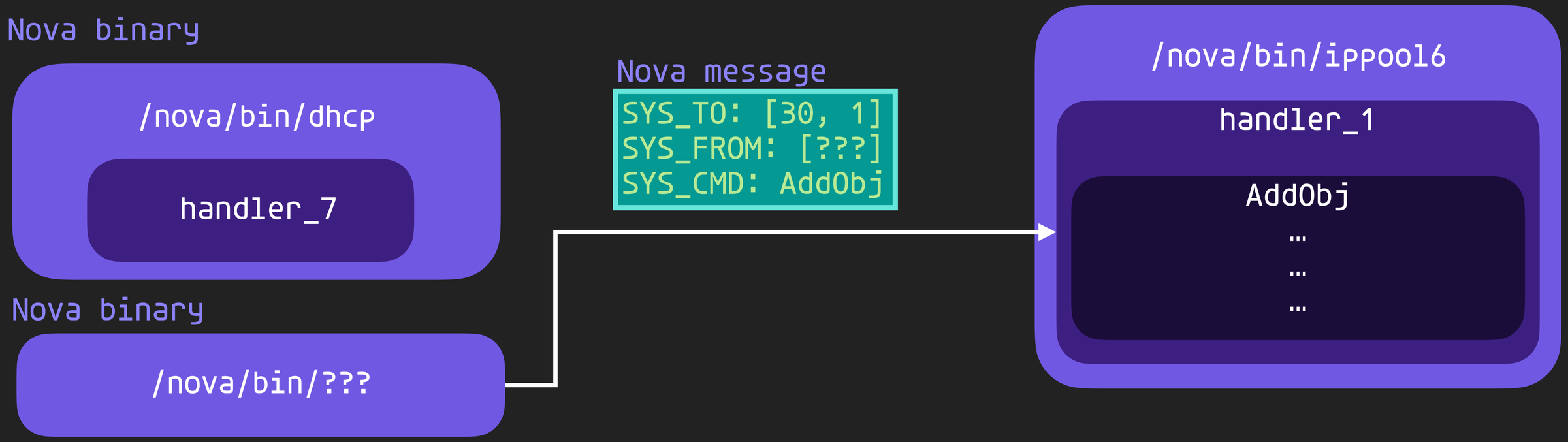
subscription and notification

Subscription



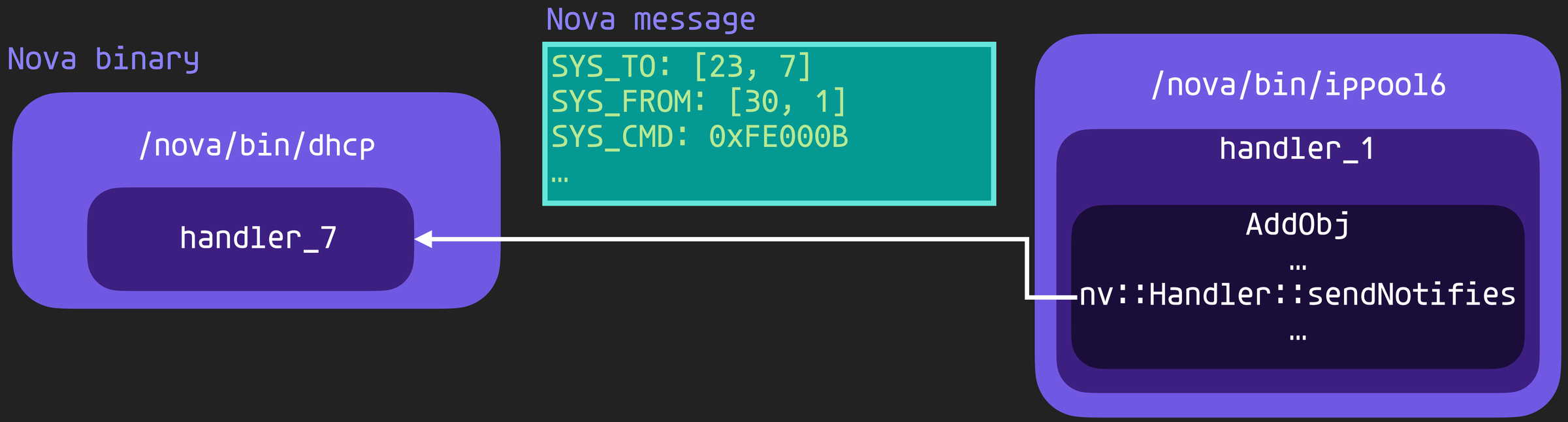
DEVCORE

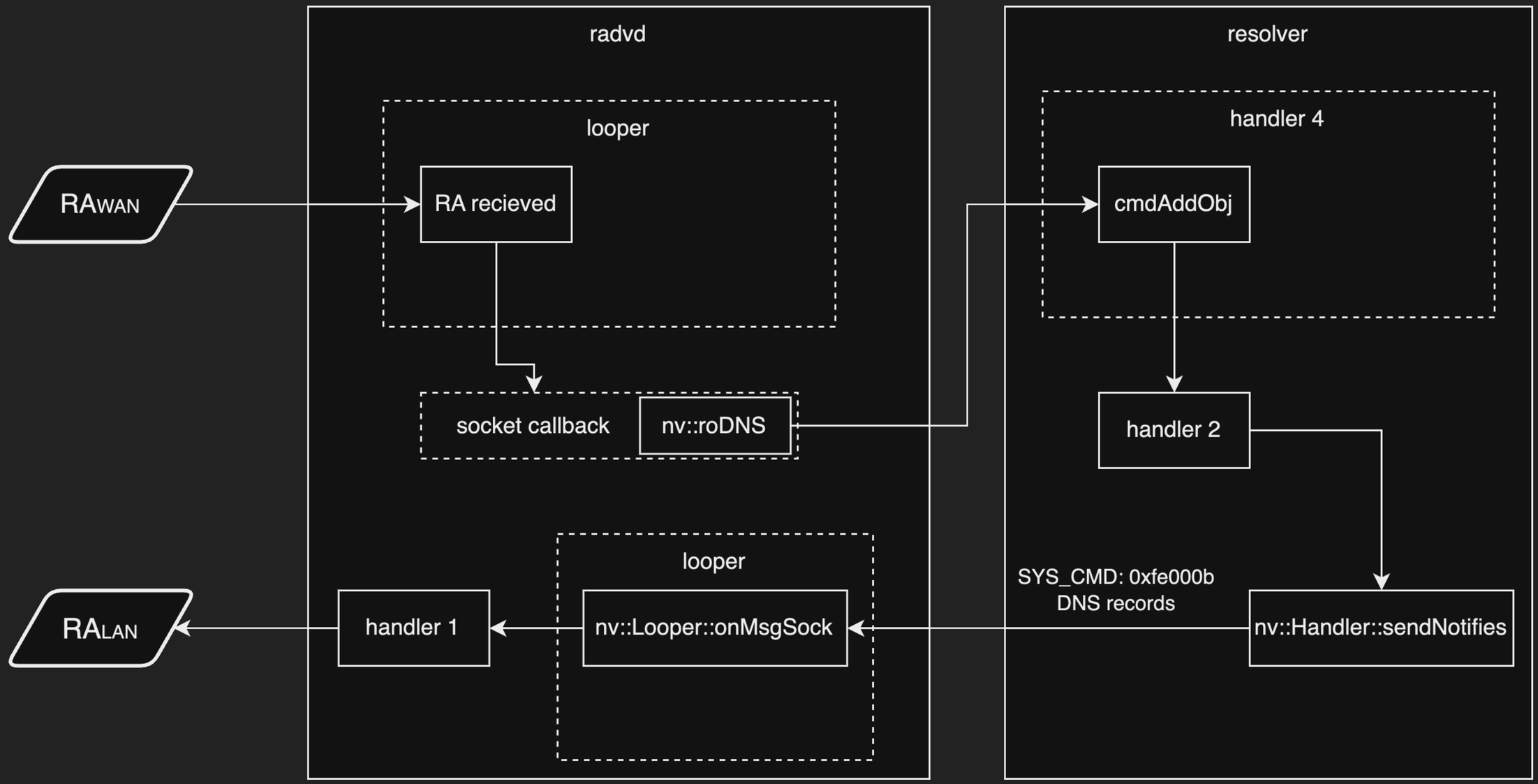
Subscription



DEVCORE

Subscription




radvd
looper
RA recieved
RAWAN
socket callback
nv::roDNS
looper
handler 1
nv::Looper::onMsgSock
RALAN
resolver
handler 4
cmdAddObj
handler 2
SYS_CMD: 0xfe000b
DNS records
nv::Handler::sendNotifies

DEVCORE

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(ha
    vector_base::swap_                                  _raw);
    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))
```

DEVCORE

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_v
    v163 = (void **)handler_1->DNS_raw.end;
    for ( IPAddr6 = handler_1->DNS_raw.start; IPAddr6 != v163; IPAddr6 += 4 )
    {
      *(_DWORD *)v181
      v183[0] = 0;
      v191[0] = 0;
      v191[1] = 0;
      v165 = (nv::Rem
      v166 = handler_
      v167 = v165;
      v168 = handler_1->DNS_remoteObject.end - (void *)v166;
      if ( v168 >> 2 == sizeOfAllocatedMem(v166) >> 2 )
```

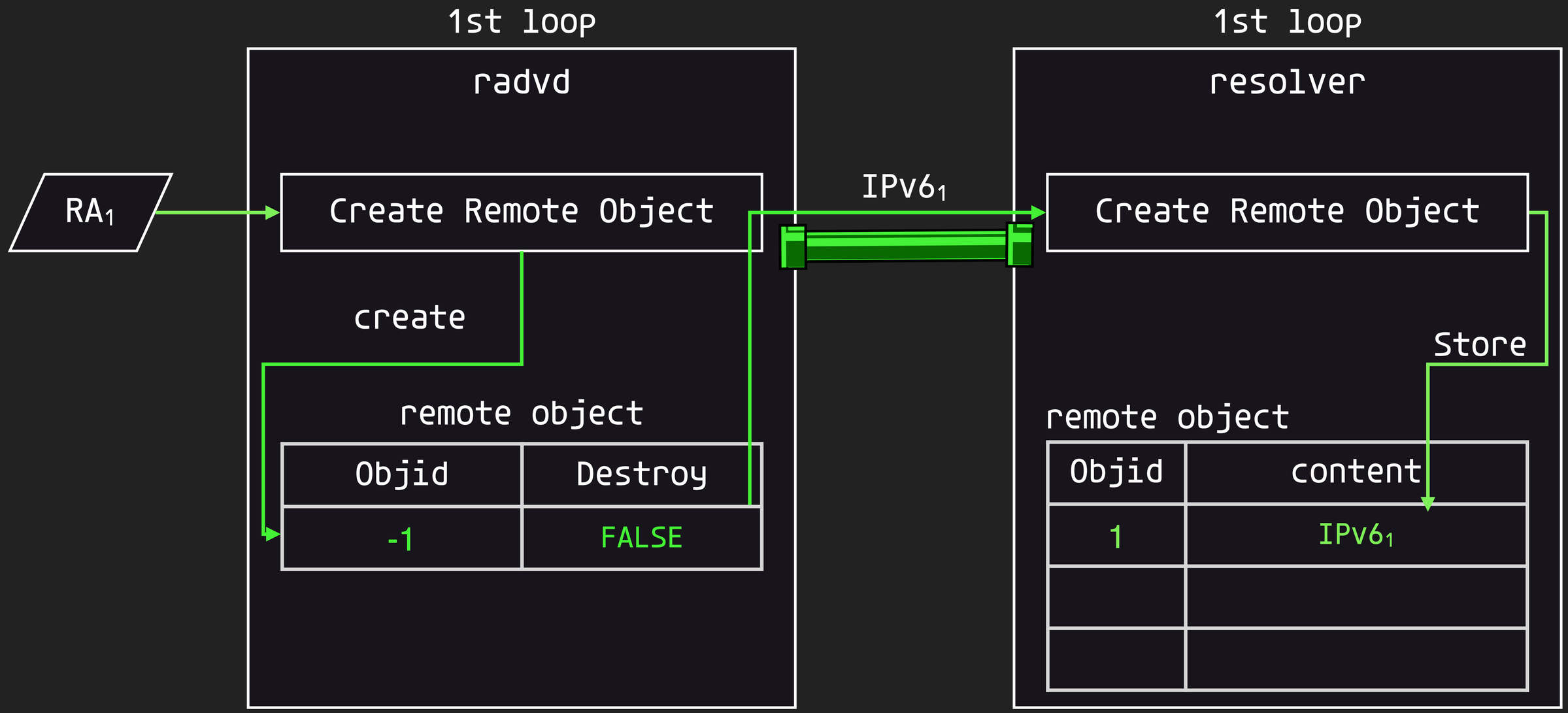
nv::RemoteObject::~RemoteObject

```
void __fastcall nv::RemoteObjectBackend::cleanup(nv::RemoteObjectBackend *this)
{
  this->destroy = 1;
  if ( this->objid != -1 )
    nv::RemoteObjectBackend::postRemove(this);
}
```

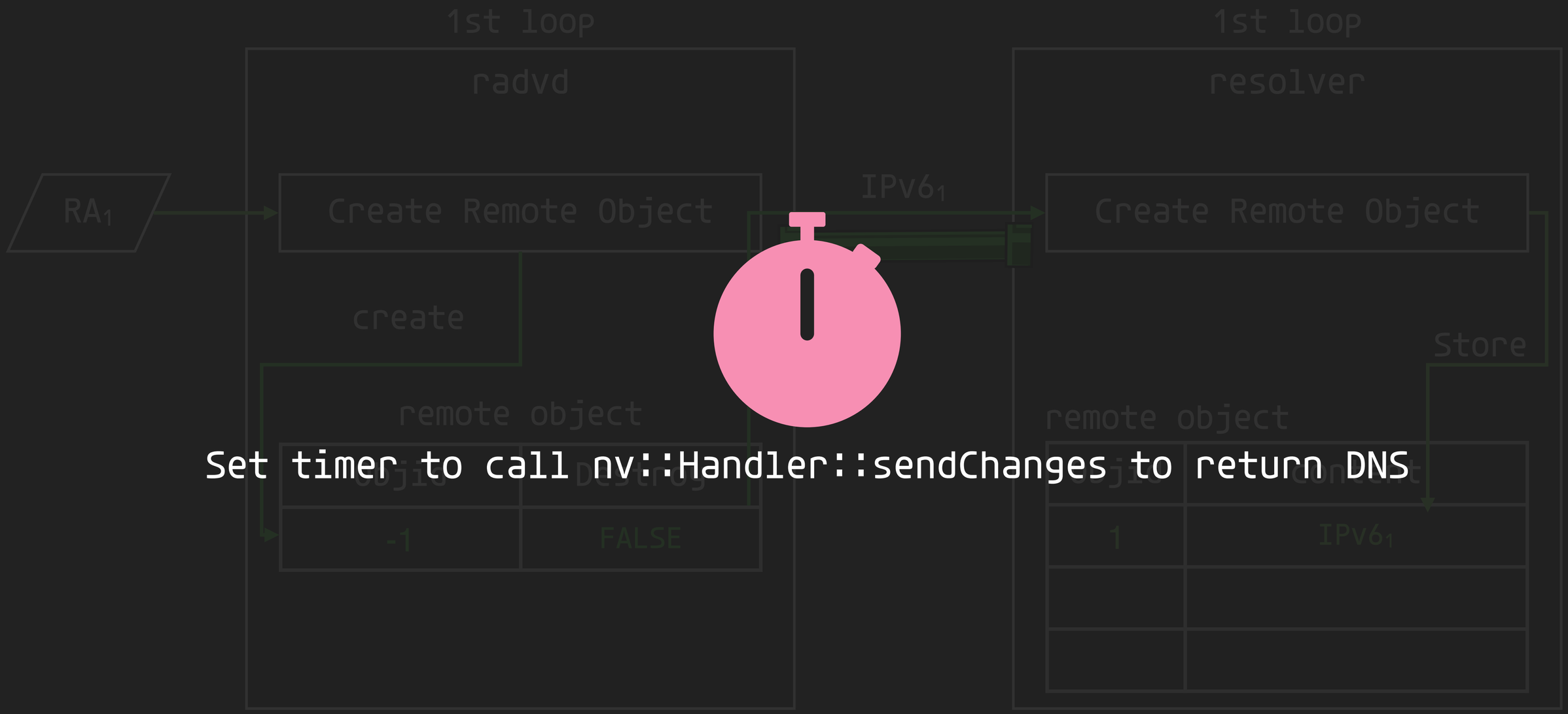
(Non-blocking)

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

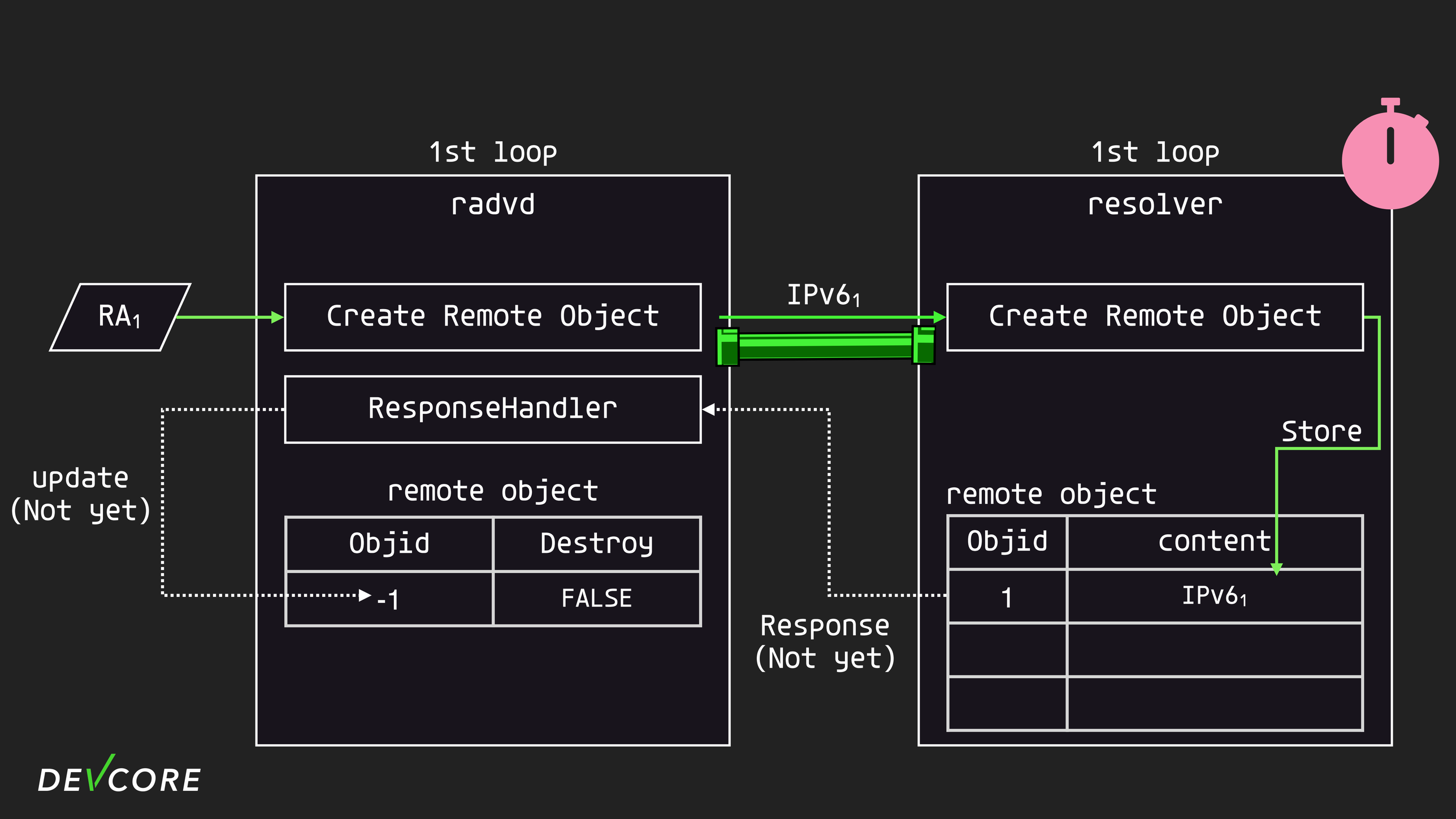
DEVCORE

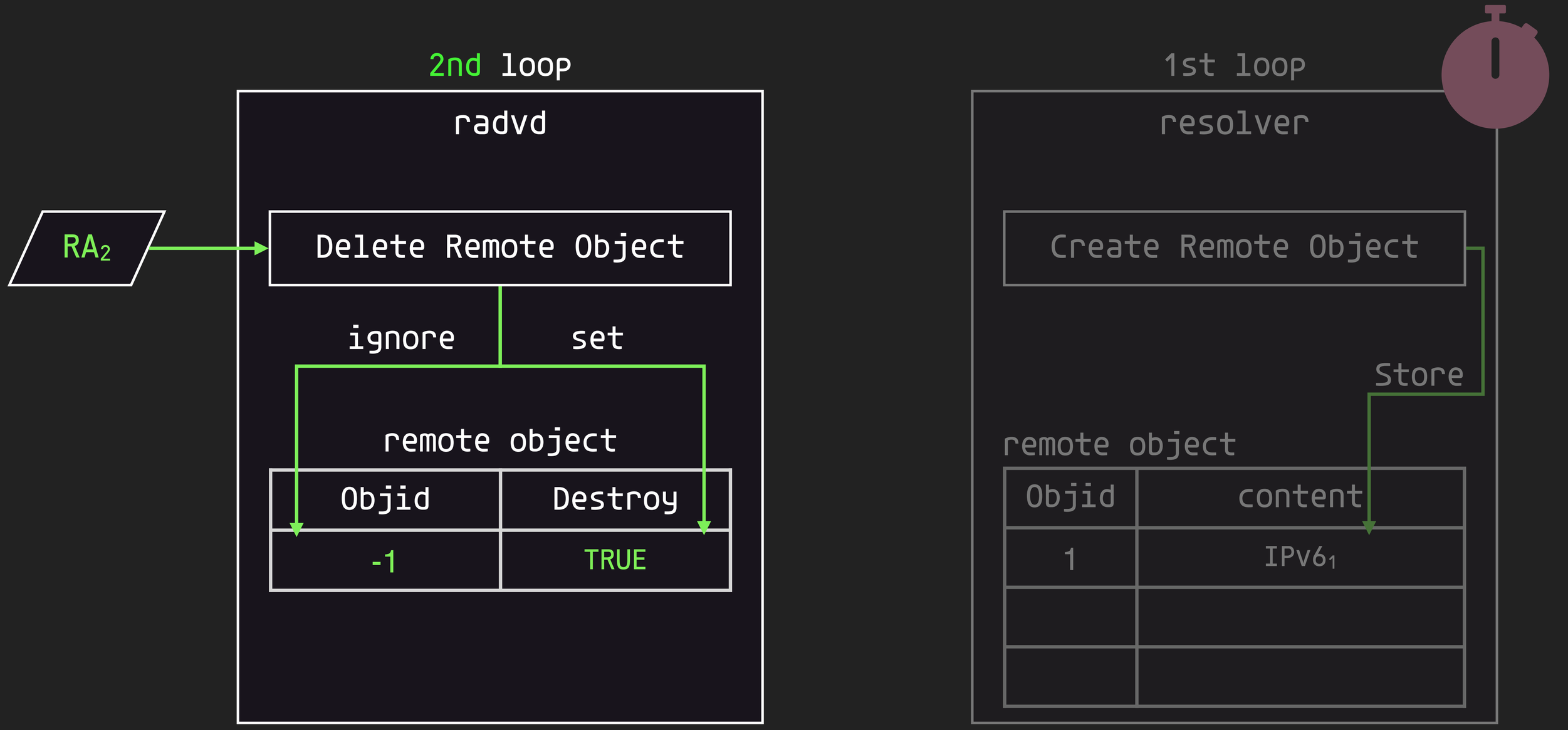

1st loop
radvd
RA1
Create Remote Object
create
remote object
Objid
Destroy
-1
FALSE
IPv61
1st loop
resolver
Create Remote Object
Store
remote object
Objid
content
1
IPv61

DEVCORE

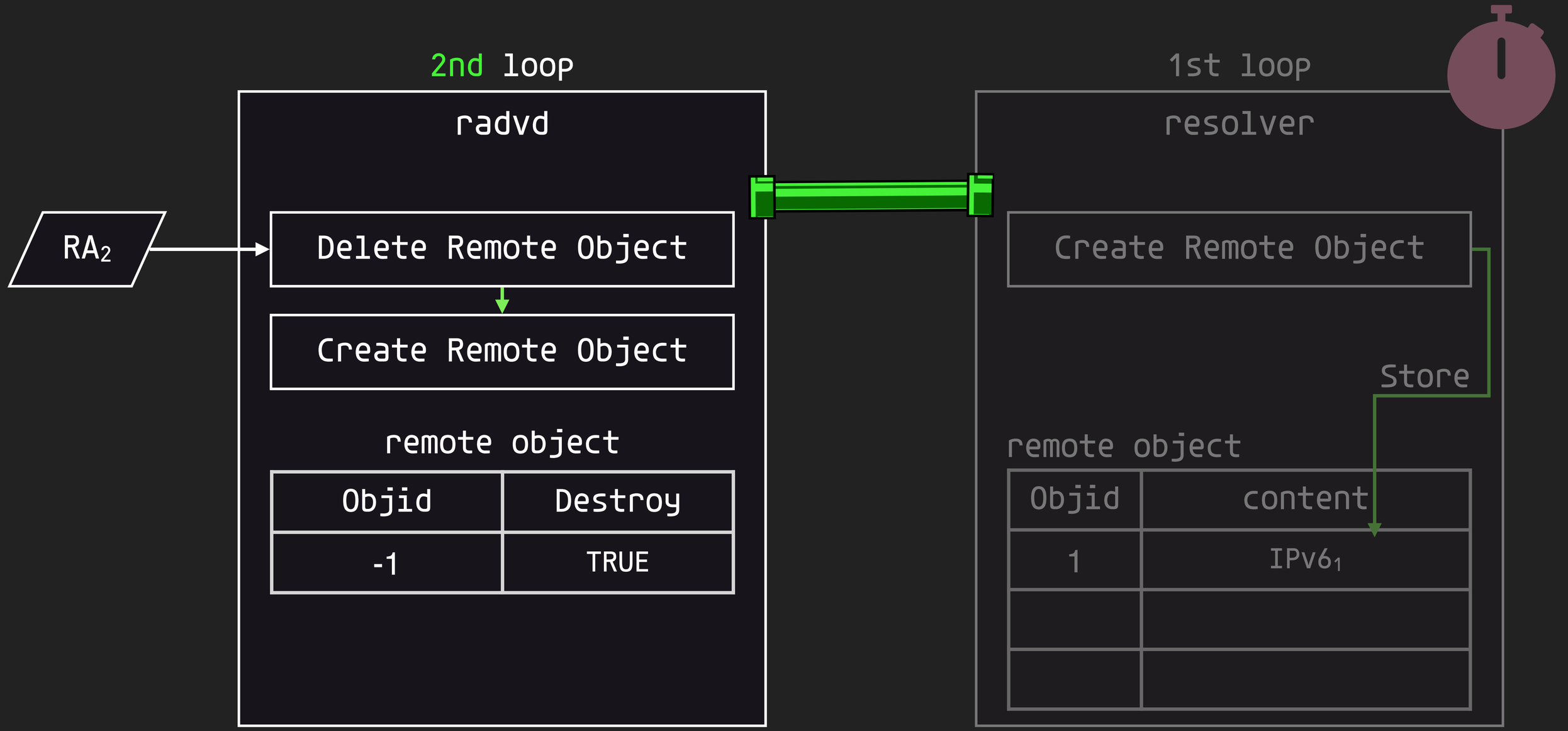


Set timer to call nv::Handler::sendChanges to return DNS

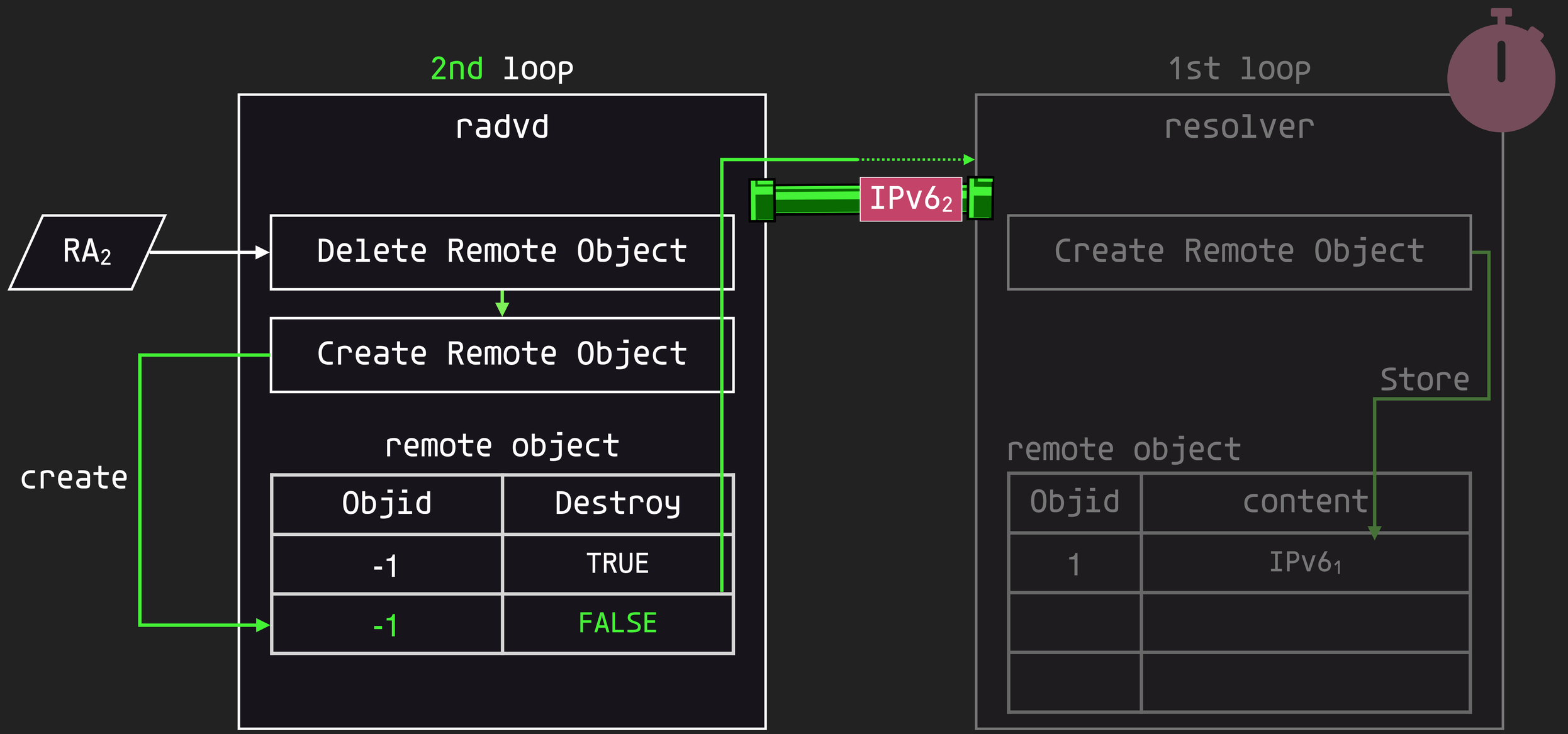

1st loop
radvd
1st loop
resolver
RA1
Create Remote Object
IPv61
Create Remote Object
ResponseHandler
Store
update
(Not yet)
remote object
Objid
Destroy
-1
FALSE
remote object
Objid
content
1
IPv61
Response
(Not yet)
DEVCORE

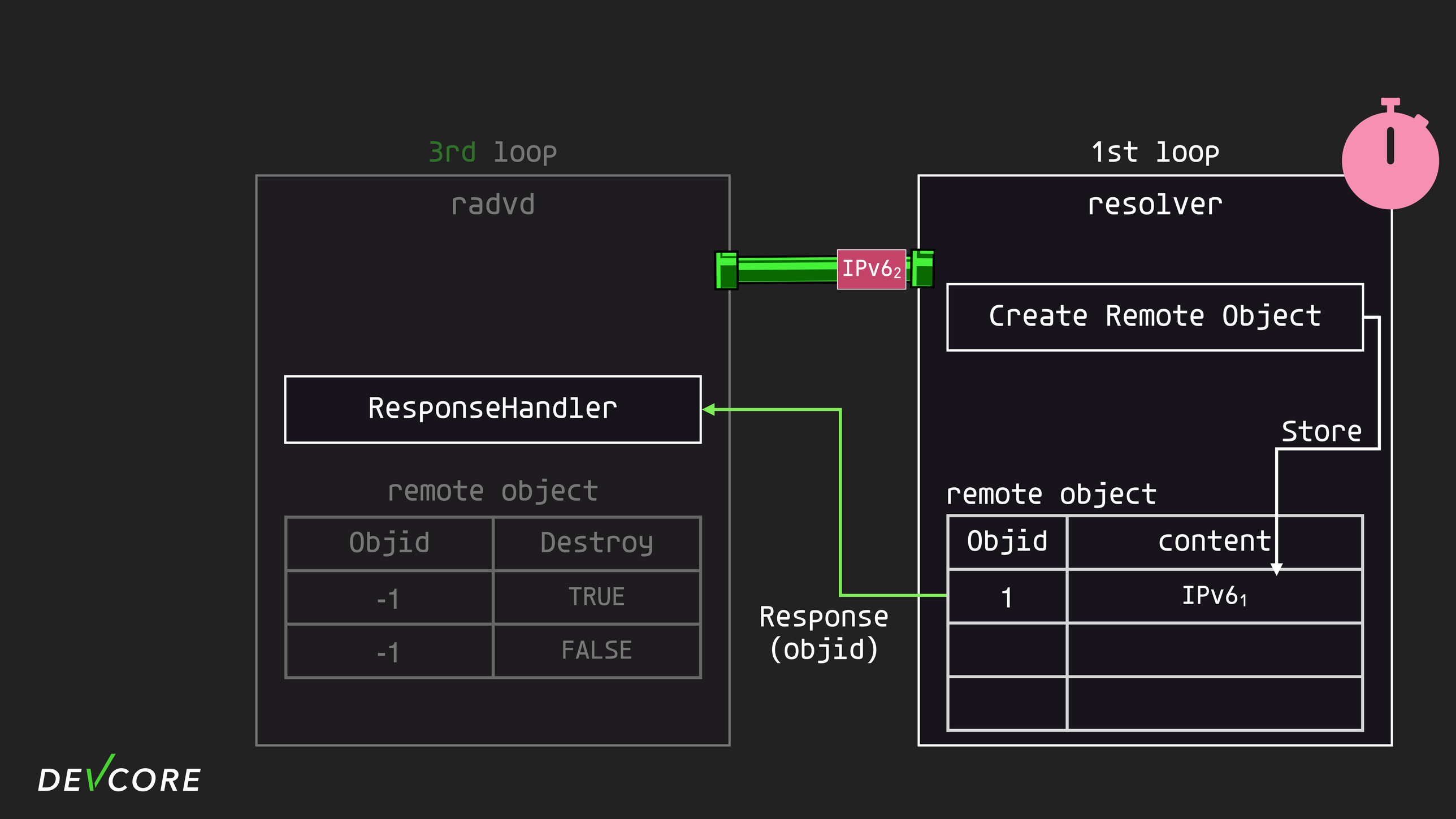

2nd loop
radvd
RA2
Delete Remote Object
ignore
set
remote object
Objid
Destroy
-1
TRUE
1st loop
resolver
Create Remote Object
Store
remote object
Objid
content
1
IPv61

DEVCORE

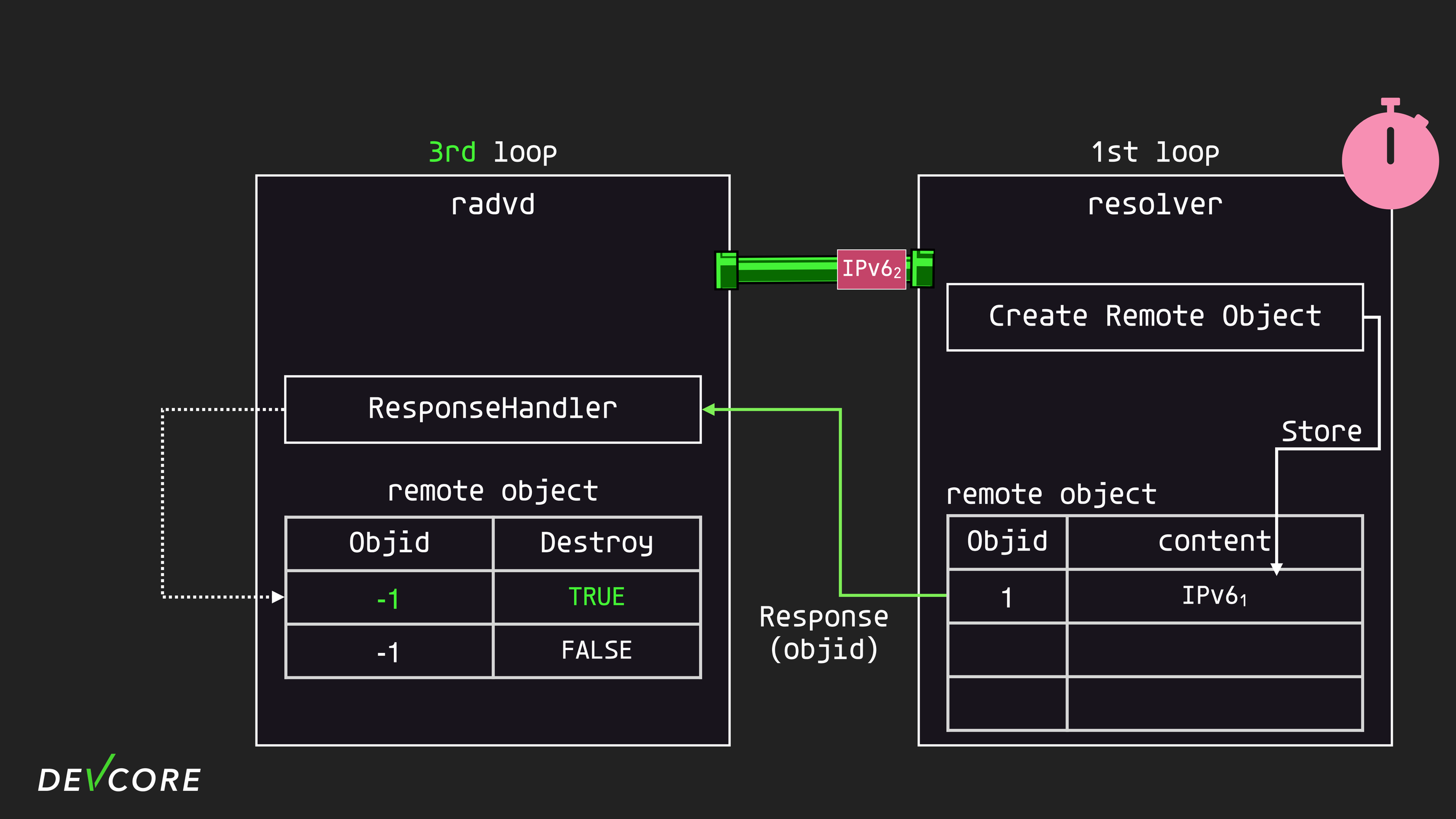

2nd loop
radvd
RA2
Delete Remote Object
Create Remote Object
remote object
Objid
Destroy
-1
TRUE
1st loop
resolver
Create Remote Object
Store
remote object
Objid
content
1
IPv61

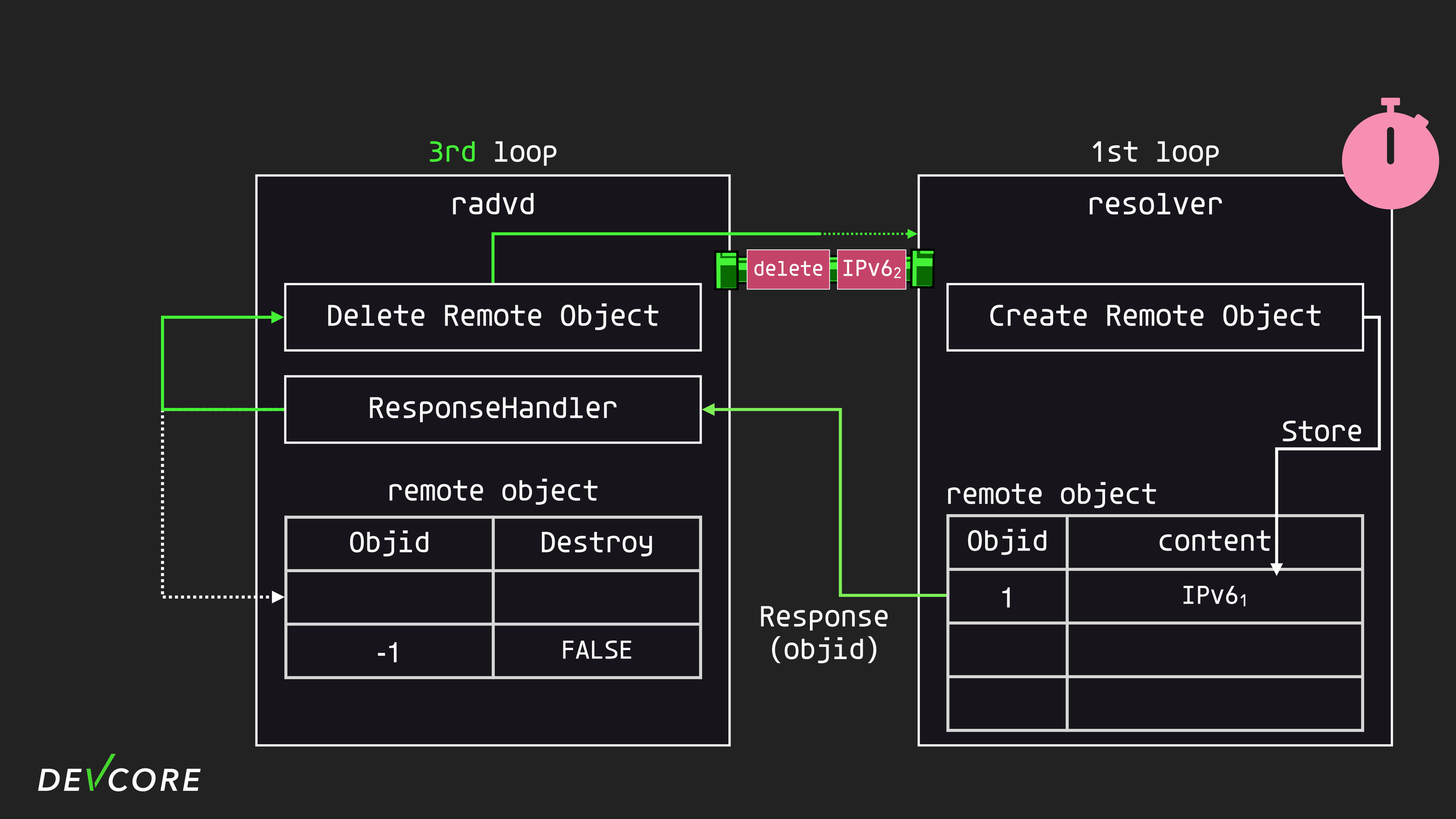
DEVCORE


2nd loop
radvd
RA2
Delete Remote Object
Create Remote Object
create
remote object
Objid
Destroy
-1
TRUE
-1
FALSE
IPv62
1st loop
resolver
Create Remote Object
Store
remote object
Objid
content
1
IPv61

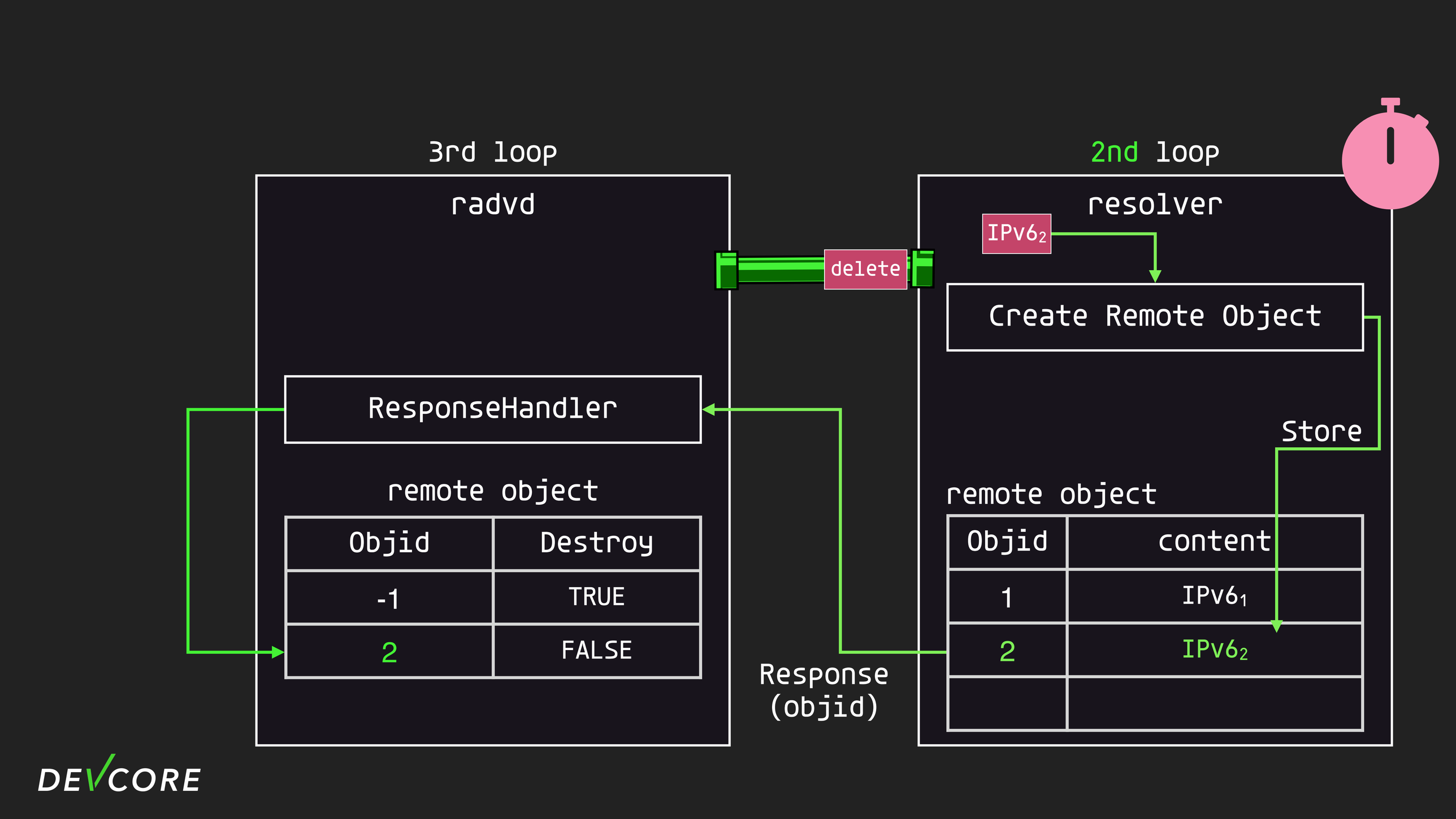

3rd loop
radvd
ResponseHandler
remote object
Objid
Destroy
-1
TRUE
-1
FALSE
IPv6₂
Response
(objid)
1st loop
resolver
Create Remote Object
Store
remote object
Objid
content
1
IPv6₁

DEVCORE

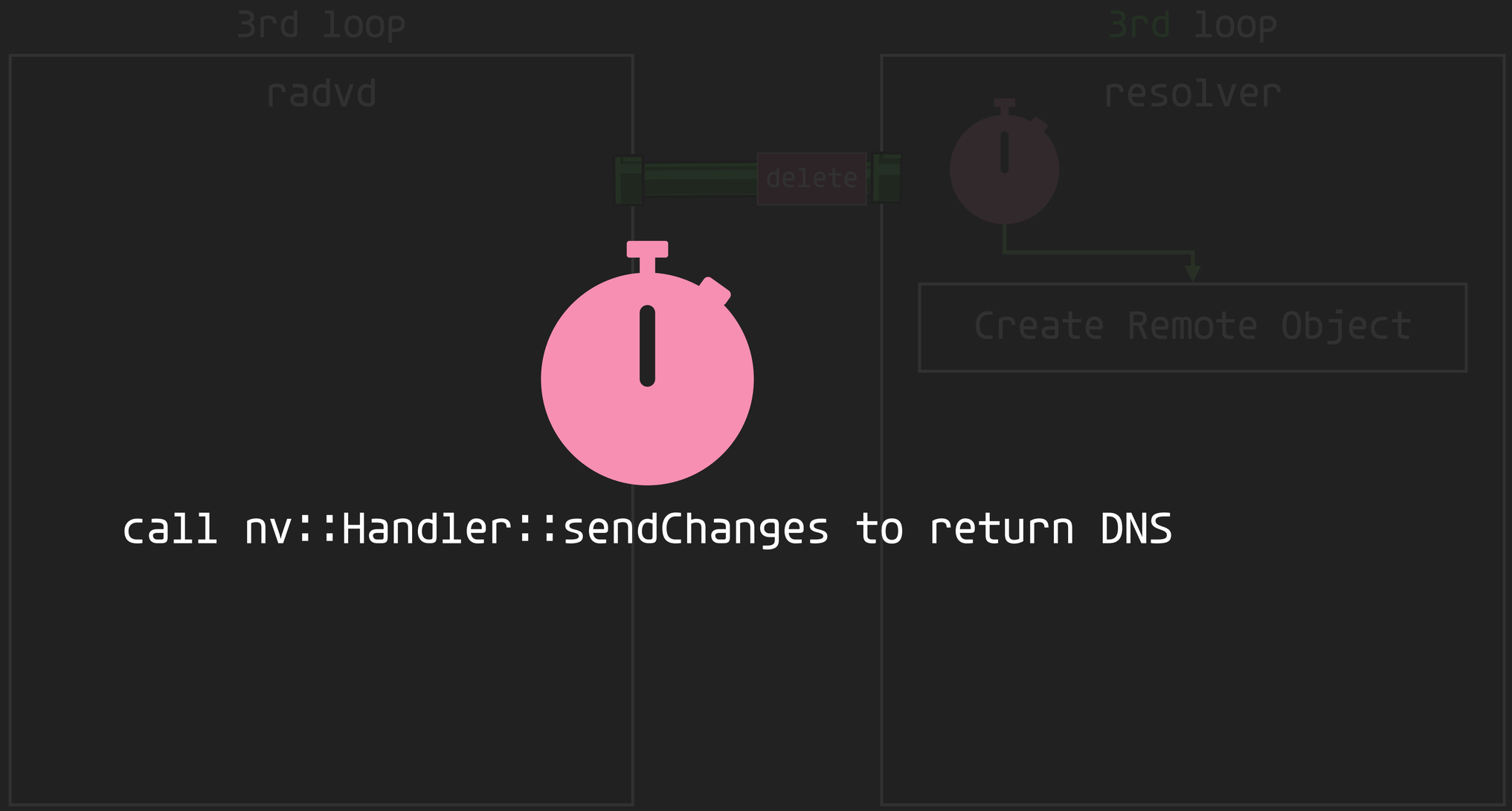

3rd loop
radvd
1st loop
resolver
IPv6₂
Create Remote Object
ResponseHandler
Store
remote object
Objid
Destroy
-1
TRUE
-1
FALSE
Response
(objid)
remote object
Objid
content
1
IPv6₁


3rd loop
radvd
Delete Remote Object
ResponseHandler
remote object
Objid
Destroy
-1
FALSE
delete
IPv6₂
Response
(objid)
1st loop
resolver
Create Remote Object
Store
remote object
Objid
content
1
IPv6₁

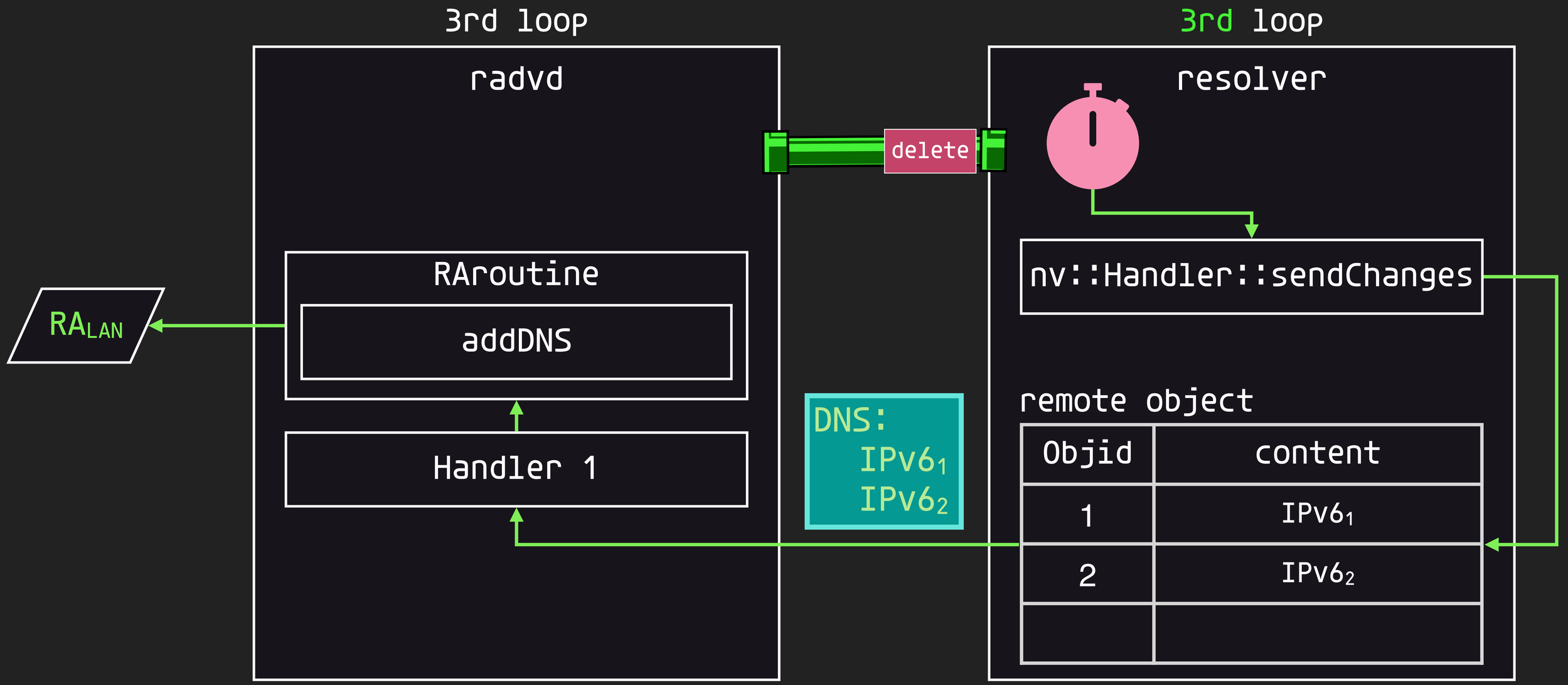
DEVCORE


3rd loop
radvd
2nd loop
resolver
IPv6₂
delete
Create Remote Object
ResponseHandler
Store
remote object
Objid
Destroy
-1
TRUE
2
FALSE
Response
(objid)
remote object
Objid
content
1
IPv6₁
2
IPv6₂

DEVCORE



call nv::Handler::sendChanges to return DNS


3rd loop
radvd
3rd loop
resolver
delete
RAroutine
addDNS
RALAN
Handler 1
nv::Handler::sendChanges
DNS:
IPv61
IPv62
remote object
Objid
content
1
IPv61
2
IPv62

DEVCORE

Race Condition

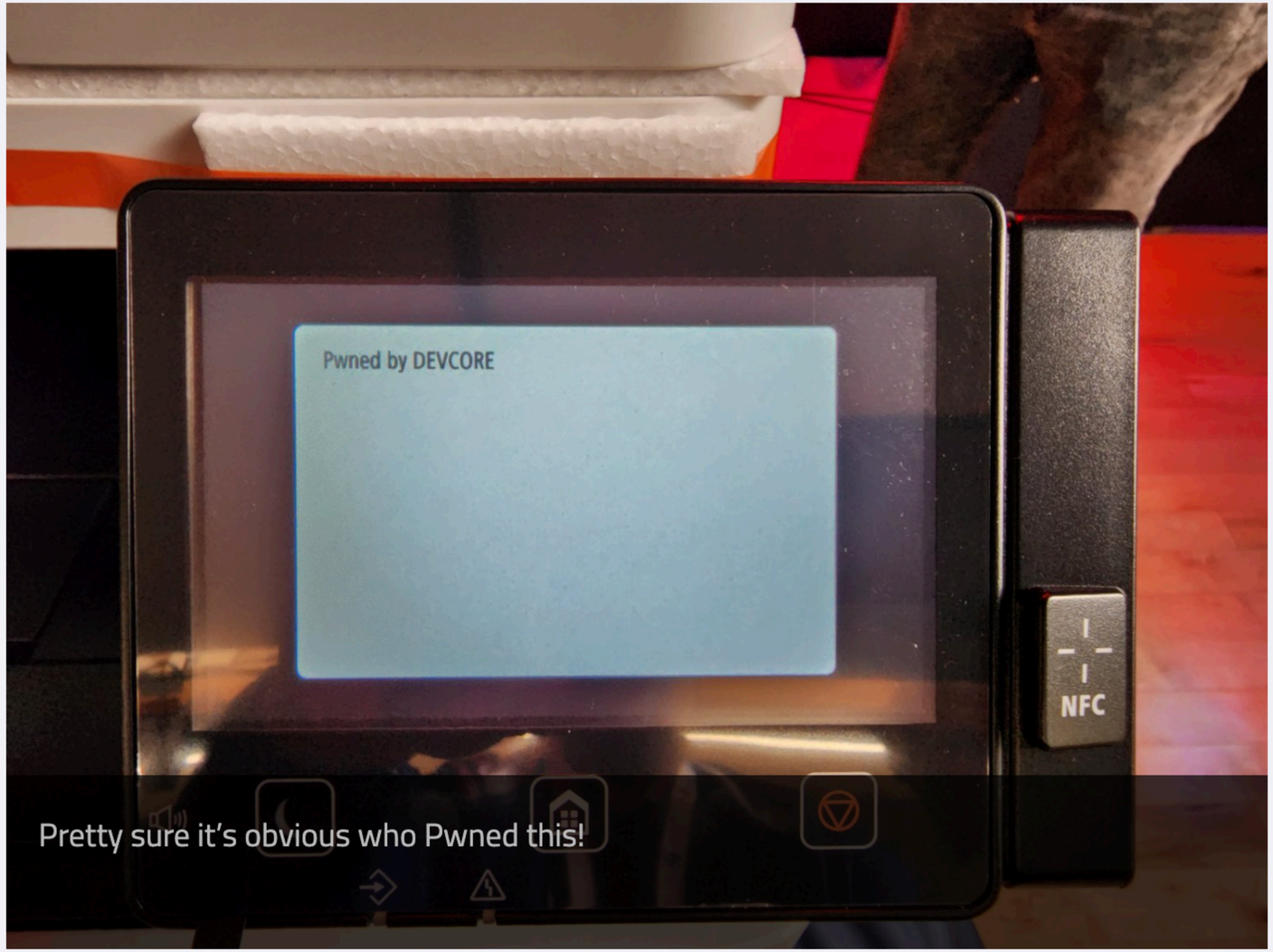
- Pattern
  - Use non-blocking methods to create/delete the remote object
  - Subscribe to the remote object
- Impact:
  - Maybe it can be used to bypass some checks

Race Condition

- Pattern
  - Use non-blocking me… the remote object
  - Subscribe to the r…
- Impact:
  - Maybe it can be us… 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.



DEVCORE

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

DEVCORE