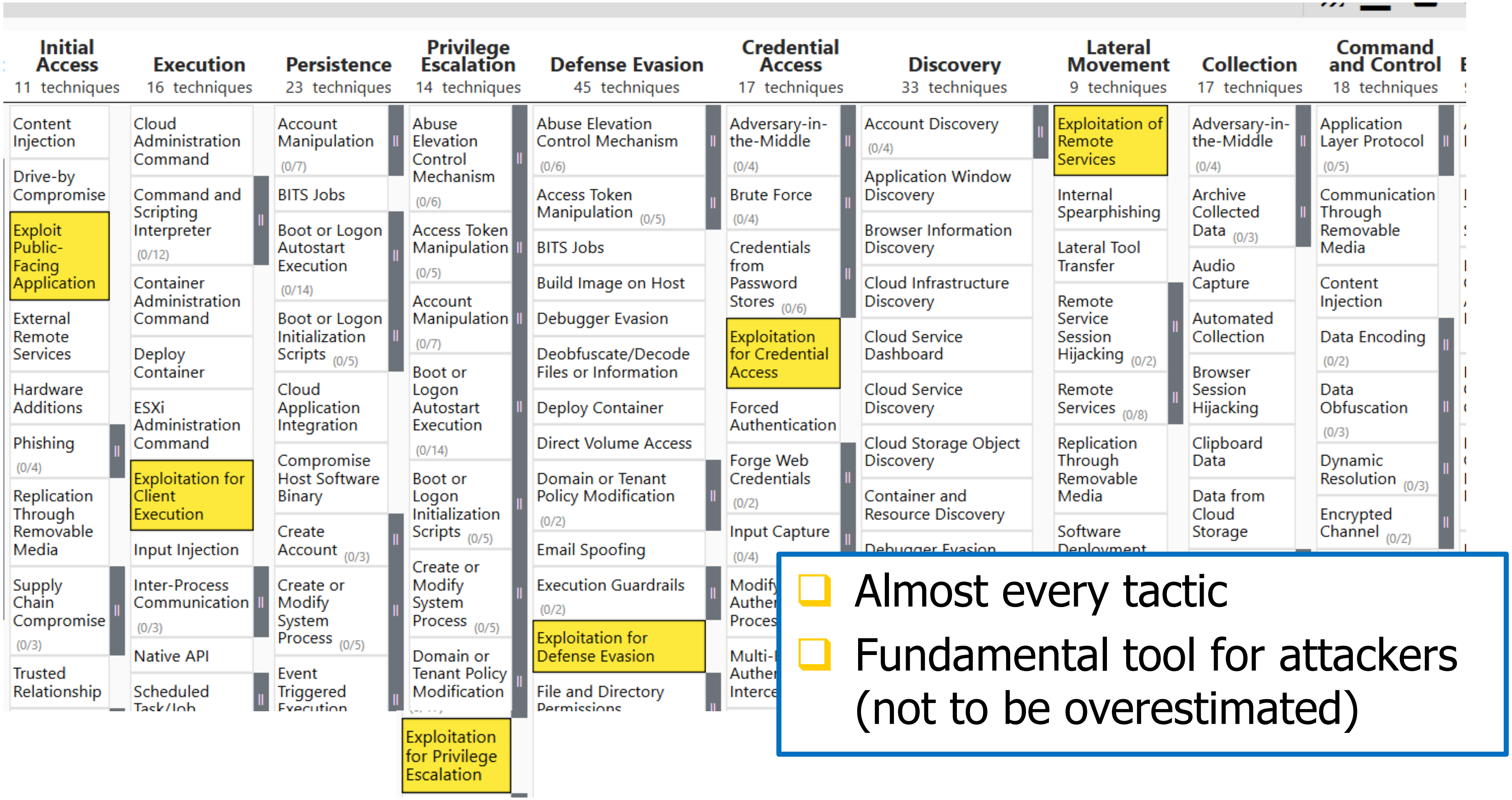
Vulnerabilities: Important Case Studies

Role of Vulnerabilities in MITRE ATT&CK



Initial Access 11 techniques	Execution 16 techniques	Persistence 23 techniques	Privilege Escalation 14 techniques	Defense Evasion 45 techniques	Credential Access 17 techniques	Discovery 33 techniques	Lateral Movement 9 techniques	Collection 17 techniques	Command and Control 18 techniques
Content Injection	Cloud Administration Command	Account Manipulation (0/7)	Abuse Elevation Control Mechanism (0/6)	Abuse Elevation Control Mechanism (0/6)	Adversary-in-the-Middle (0/4)	Account Discovery (0/4)	Exploitation of Remote Services	Adversary-in-the-Middle (0/4)	Application Layer Protocol (0/5)
Drive-by Compromise	Command and Scripting Interpreter (0/12)	BITS Jobs	Access Token Manipulation (0/5)	Access Token Manipulation (0/5)	Brute Force (0/4)	Application Window Discovery	Internal Spearphishing	Archive Collected Data (0/3)	Communication Through Removable Media
Exploit Public-Facing Application	Container Administration Command	Boot or Logon Autostart Execution (0/14)	Account Manipulation (0/7)	BITS Jobs	Credentials from Password Stores (0/6)	Browser Information Discovery	Lateral Tool Transfer	Audio Capture	Content Injection
External Remote Services	Deploy Container	Boot or Logon Initialization Scripts (0/5)	Boot or Logon Autostart Execution (0/14)	Build Image on Host	Exploitation for Credential Access	Cloud Infrastructure Discovery	Remote Service Session Hijacking (0/2)	Automated Collection	Data Encoding (0/2)
Hardware Additions	ESXi Administration Command	Cloud Application Integration	Boot or Logon Initialization Scripts (0/5)	Debugger Evasion	Forced Authentication	Cloud Service Dashboard	Remote Services (0/8)	Browser Session Hijacking	Data Obfuscation (0/3)
Phishing (0/4)	Exploitation for Client Execution	Compromise Host Software Binary	Create or Modify System Process (0/5)	Deobfuscate/Decode Files or Information	Forge Web Credentials (0/2)	Cloud Service Discovery	Replication Through Removable Media	Clipboard Data	Dynamic Resolution (0/3)
Replication Through Removable Media	Input Injection	Create Account (0/3)	Domain or Tenant Policy Modification	Deploy Container	Input Capture (0/4)	Cloud Storage Object Discovery	Software Deployment	Data from Cloud Storage	Encrypted Channel (0/2)
Supply Chain Compromise (0/3)	Inter-Process Communication (0/3)	Create or Modify System Process (0/5)	Exploitation for Privilege Escalation	Direct Volume Access	Modify Authen Process	Container and Resource Discovery			
Trusted Relationship	Native API	Event Triggered Execution		Domain or Tenant Policy Modification (0/2)	Multi- Authen Interce	Debugger Evasion			
	Scheduled Task/Job			Email Spoofing					
				Execution Guardrails (0/2)					
				Exploitation for Defense Evasion					
				File and Directory Permissions					

- Almost every tactic
- Fundamental tool for attackers (not to be overestimated)

Our path

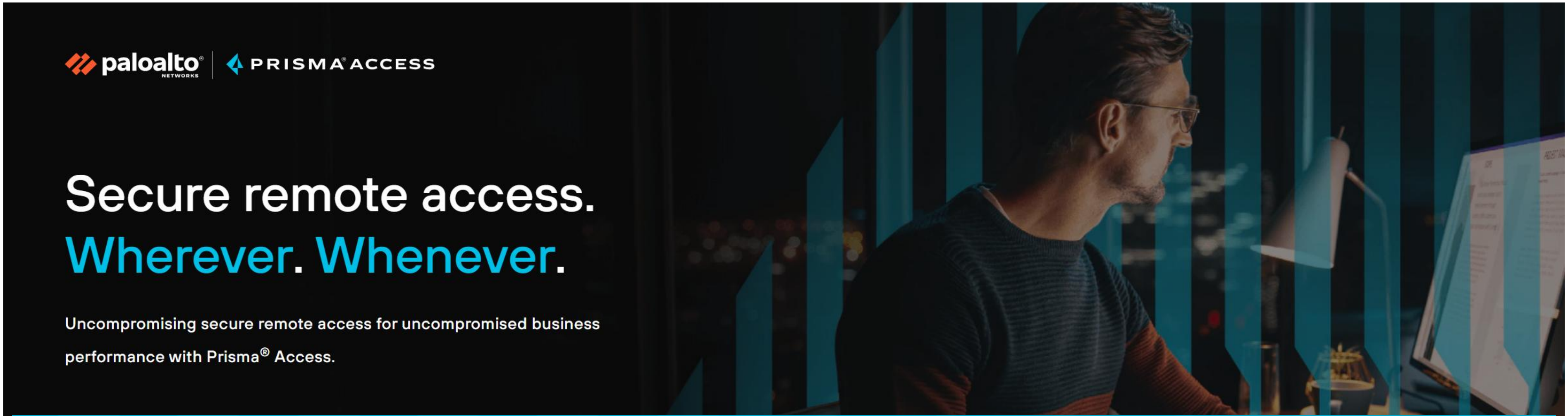
- We will look at specific examples
- Tiny little bug in software → Catastrophe

- Discussion of implications for cybersecurity and testing

- Much broader discussion later

PaloAlto Networks Global Protect (April 2024)

PaloAlto Networks Global Protect (I)



paloalto NETWORKS | PRISMA ACCESS

Secure remote access.

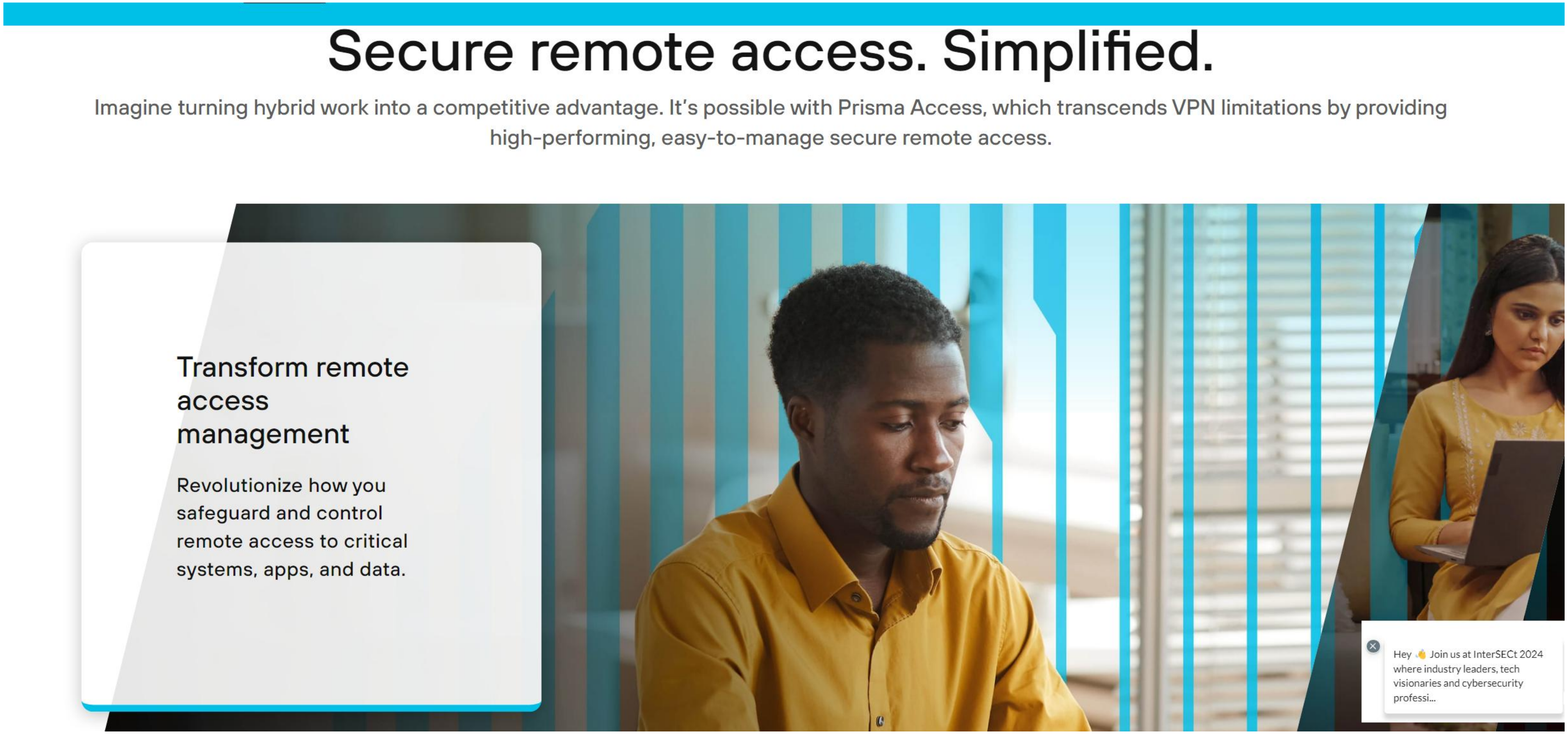
Wherever. Whenever.

Uncompromising secure remote access for uncompromised business performance with Prisma® Access.

Your hybrid workforce needs secure access to apps and data, yet traditional VPN is falling short.

When VPN solutions frustrate your users and IT teams, they bring down productivity and raise security risks.

PaloAlto Networks Global Protect (II)



Secure remote access. Simplified.

Imagine turning hybrid work into a competitive advantage. It's possible with Prisma Access, which transcends VPN limitations by providing high-performing, easy-to-manage secure remote access.

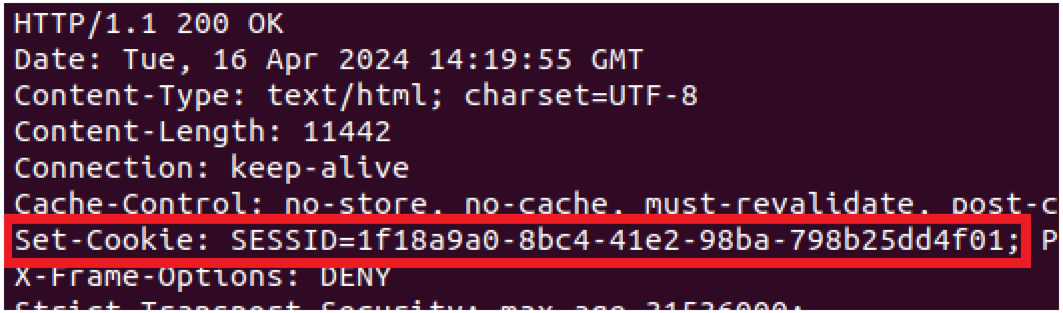
Transform remote access management

Revolutionize how you safeguard and control remote access to critical systems, apps, and data.

Hey 👋 Join us at InterSECt 2024 where industry leaders, tech visionaries and cybersecurity professi...

Tiny little bug

- Web application does not check that `Cookie` values have the expected syntactic structure

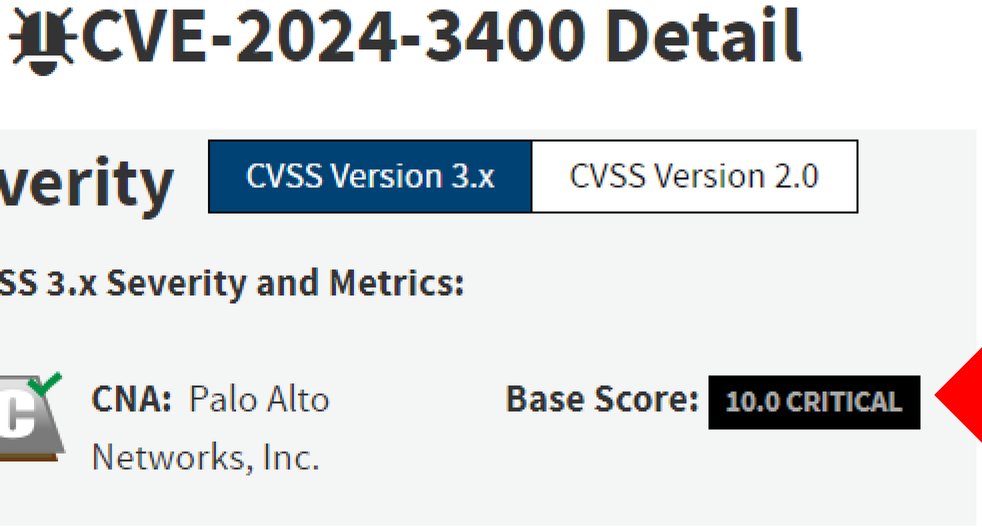


```
HTTP/1.1 200 OK
Date: Tue, 16 Apr 2024 14:19:55 GMT
Content-Type: text/html; charset=UTF-8
Content-Length: 11442
Connection: keep-alive
Cache-Control: no-store, no-cache, must-revalidate, post-c
Set-Cookie: SESSID=1f18a9a0-8bc4-41e2-98ba-798b25dd4f01; P
X-Frame-Options: DENY
Strict-Transport-Security: max-age=31536000;
```

- Well, what impact this tiny little problem could possibly have?

Oh my…

- **Unauthenticated** attacker may execute **arbitrary** code with **root** privileges on the firewall



CVE-2024-3400 Detail

Severity

CVSS 3.x Severity and Metrics:

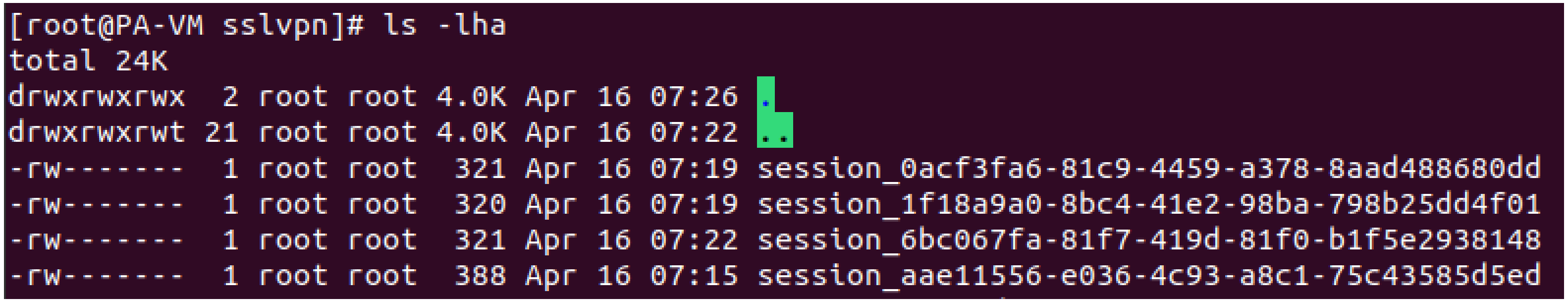
CNA: Palo Alto Networks, Inc.

Base Score: 10.0 CRITICAL

How the appliance works (reverse engineered)

Session State Management

- Session State of session **ID** is stored in file named `/tmp/sslvpn/session_`**`ID`**



```
[root@PA-VM sslvpn]# ls -lha
total 24K
drwxrwxrwx  2 root root 4.0K Apr 16 07:26 .
drwxrwxrwt 21 root root 4.0K Apr 16 07:22 ..
-rw-------  1 root root  321 Apr 16 07:19 session_0acf3fa6-81c9-4459-a378-8aad488680dd
-rw-------  1 root root  320 Apr 16 07:19 session_1f18a9a0-8bc4-41e2-98ba-798b25dd4f01
-rw-------  1 root root  321 Apr 16 07:22 session_6bc067fa-81f7-419d-81f0-b1f5e2938148
-rw-------  1 root root  388 Apr 16 07:15 session_aae11556-e036-4c93-a8c1-75c43585d5ed
```

Telemetry data (I)

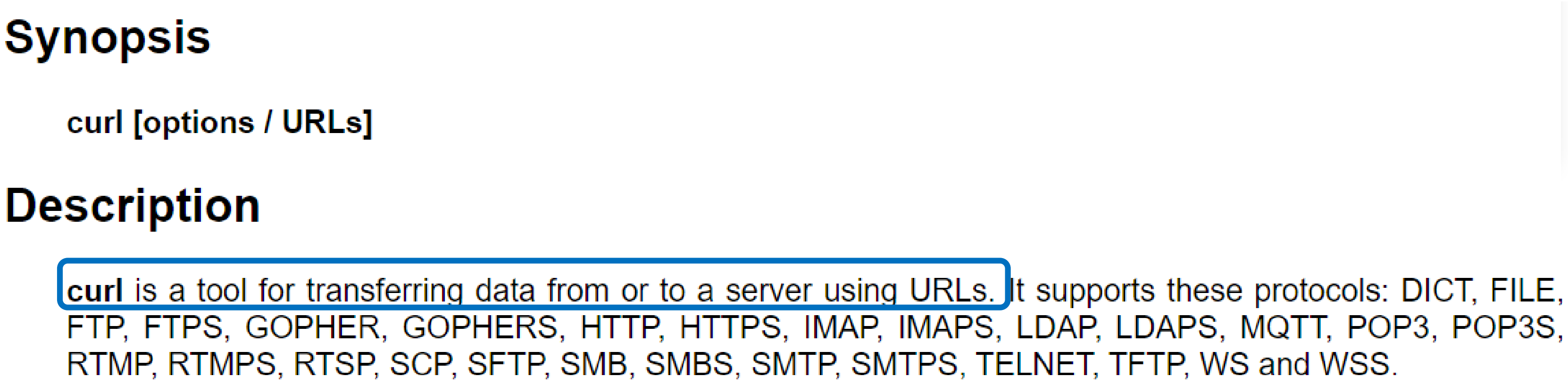
- Telemetry data ("what happened"):
  - Stored in these directories (files of a certain format)

    ```
    /opt/panlogs/tmp/device_telemetry/hour
    /opt/panlogs/tmp/device_telemetry/day
    ```

  - Sent **every hour** to a configuration-specified URL

Telemetry data (II)

- Sent **every hour** to a configuration-specified URL
  - Periodic `cron` job
  - File upload with `curl` command



Synopsis

curl [options / URLs]

Description

curl is a tool for transferring data from or to a server using URLs. It supports these protocols: DICT, FILE, FTP, FTPS, GOPHER, GOPHERS, HTTP, HTTPS, IMAP, IMAPS, LDAP, LDAPS, MQTT, POP3, POP3S, RTMP, RTMPS, RTSP, SCP, SFTP, SMB, SMBS, SMTP, SMTPS, TELNET, TFTP, WS and WSS.

curl command within code (I)

```
...
curl_cmd = "/usr/bin/curl -v
       -H \"Content-Type: application/octet-stream\"
       -X PUT \"%s\"
       --data-binary @%s
       --capath %s
       --interface %s"\
       %(signedUrl, fname, capath, source_ip_str)
...
pansys(curl_cmd, shell=True, timeout=250)
...
```

curl command within code (II)

```
...
pansys(curl_cmd, shell=True, timeout=250)
...
```

```
// nested within several function invocations
...
p = subprocess.Popen(cmd,
      stdout=subprocess.PIPE,
      bufsize=1, shell=True,
      stderr=subprocess.PIPE,
      close_fds=close_fds,
      universal_newlines=True)
...
```

Description of parameters by ChatGPT
https://chat.openai.com/share/f3fd3d41-8660-44e8-b3b7-46b0fe713d6f

Putting it all together

Arbitrary Empty File Creation

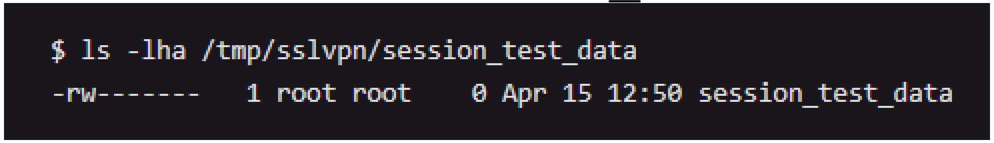
- Web application does not check that Cookie values have the expected syntactic structure

+

- Session State of session **ID** is stored in file named `/tmp/sslvpn/session_`**`ID`**

- Send HTTP Request with carefully structured cookie value will create an empty file with **attacker-controlled name**

```
Cookie: SESSID=test_data
```



```
$ ls -lha /tmp/sslvpn/session_test_data
-rw-------   1 root root    0 Apr 15 12:50 session_test_data
```

Bad Practices (I)

1. Web application runs as root
2. …

- Not necessarily wrong
- In this case **essential** for the exploit

- Not only for its impact

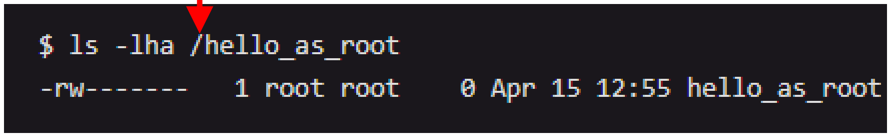
Directory Traversal

- ❑ Web application runs as root

+

- ❑ Send HTTP Request with carefully structured cookie value will create an empty file with **attacker-controlled name** **in any arbitrary attacker-chosen directory**

```
Cookie: SESSID=./../../../hello_as_root
```



```
$ ls -lha /hello_as_root
-rw-------    1 root root     0 Apr 15 12:55 hello_as_root
```

```
For some reason, with directory traversal
the heading "session_" is not inserted in filename
```

What we have so far

- We can write an empty file of our choice in any directory

↓

- We can write an empty file of our choice in telemetry directories

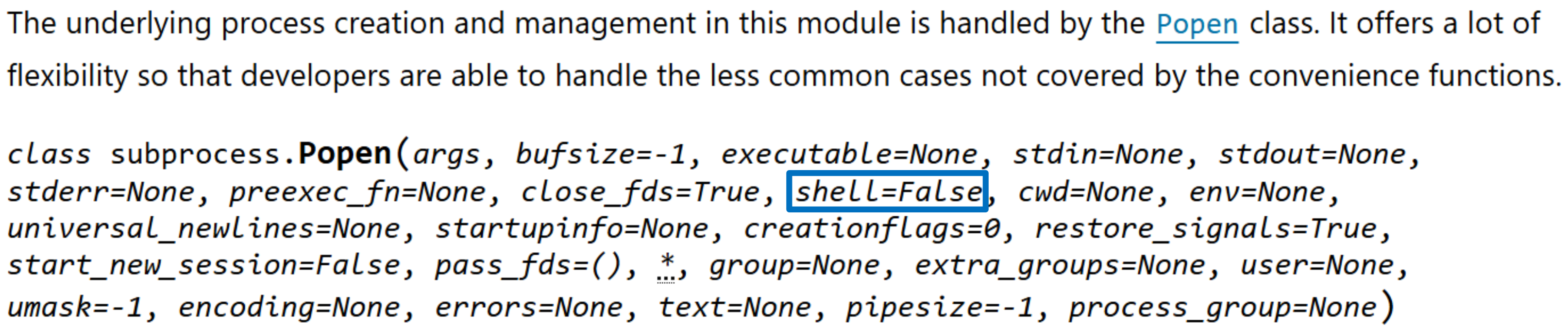
↓

- `curl` will send an empty telemetry file of our choice to the configuration-specified URL

- Not useful (yet)

subprocess.Popen() (I)

```
// nested within several function invocations
...
p = subprocess.Popen(cmd,
      stdout=subprocess.PIPE,
      bufsize=1, shell=True,
      stderr=subprocess.PIPE,
      close_fds=close_fds,
      universal_newlines=True)
...
```



The underlying process creation and management in this module is handled by the Popen class. It offers a lot of flexibility so that developers are able to handle the less common cases not covered by the convenience functions.

```
class subprocess.Popen(args, bufsize=-1, executable=None, stdin=None, stdout=None,
stderr=None, preexec_fn=None, close_fds=True, shell=False, cwd=None, env=None,
universal_newlines=None, startupinfo=None, creationflags=0, restore_signals=True,
start_new_session=False, pass_fds=(), *, group=None, extra_groups=None, user=None,
umask=-1, encoding=None, errors=None, text=None, pipesize=-1, process_group=None)
```

`subprocess.Popen()` (II)

- `cmd = "/usr/bin/curl -v … -X PUT …"`

- `subprocess.Popen(cmd,...,shell=False,...)`
  - `cmd` is **a list** of strings
  - Spawns a shell that executes `cmd[0]` (with `cmd[1],cmd[2],…` as parameters)
  - This is the default

- `subprocess.Popen(cmd,...,shell=True,...)`
  - `cmd` is a string
  - Spawns a shell that executes `cmd`

Linux shell feature: Backticks

```
2                 1
echo "Today is `date`"
echo "Number of files: `ls | wc -l`"
```

- **Command** enclosed within **backticks**
- Bash behavior:
  1. Execute **that command** first
  2. Replace it with its output
  3. Execute enclosing command

BOOM!

```
Cookie: SESSID=PATH_TO_TELEMETRY_DIR/aaa`command`
```

1. Cookie management will create file named `aaa`command`` in telemetry directory
2. Telemetry will invoke **a shell** for executing `curl ... -X PUT ... aaa`command`...`
3. Shell will execute **`command`** and then **`curl`** (as `root`!)

- **Unauthenticated remote command injection**

Bad Practices (II)

1. Web application runs as root
2. Executes **a shell** even though it needs to execute **one** single command

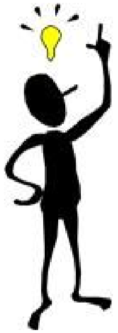
- ❑ Not necessarily wrong
- ❑ In this case they were **essential** for the exploit

DEFAULT subprocess.Popen()

```
Cookie: SESSID=PATH_TO_TELEMETRY_DIR/aaa`command`
```

1. Cookie management will create file named `` aaa`command` `` in telemetry directory
2. Telemetry will execute
   ```
   curl ... -X PUT ... aaa`command`...
   ```
3. `curl` will upload that (empty) file to the configuration URL

- ~~Unauthenticated remote command injection~~



Never Forget

- Web application does not check that `Cookie` values have the expected syntactic structure

  +

- Two bad practices

  =

- Unauthenticated Remote Command Execution (Pre-Auth RCE)

Understanding this vuln

- What is the **exploit**?
- What is the **payload**?
- What is the **threat model** for its exploitation?

- How difficult is **exploit development**?
- How difficult is **injection**?

- Can it be used for **Initial Access**? Which technique?
- For **Execution**?
- For **Persistence**?

Reasoning about detection

- How to detect **injection attempts**?
- ...how to **block** them?

- How to detect **injections after the fact**?

General Lessons about vulnerabilities

You never know

1. Web application runs as root
2. Executes **a shell** even though it needs to execute **one** single command

- Unsafe practices are unsafe. Period.
- One cannot predict their reach

- Avoid them as much as you can

Never ONE reason

- Vulnerabilities never result from **one** single reason

- They usually result from **many different** and **seemingly unrelated** reasons

NEVER trust input data

- Never. Not even implicitly

- **Structure** may not be what you expect
- **Values** may not be what you expect

- Much easier said than done

Testing (I)

- More intensive / accurate testing might have discovered **a bug**

- It would have **not** discovered **this vulnerability automatically**

Testing (II-a)

- Do **not** expect to discover a **vulnerability** in your testing campaigns
- You will usually observe **only bugs**

Testing (II-b)

- Bugs have to be **categorized**:
  1. Not vulnerability
  2. Hardly exploitable vulnerability
  3. Exploitable vulnerability
- Categorization requires **skills**, **time**, **manual work**

- Never quickly dismiss a bug as "this is not a vulnerability"
- Never start crying whenever you discover a bug

Very deep question

- You are testing your software product

- Which inputs should you inject?
  - Those representative of **normal** usage
  - **Wrong**, either partially or in full

Cybersec is a different (harder) game!

- **Fridge** design and development
- Requirement:
  - Operation with external temperature [-15,+40] °C

- You will **not** test with 200 °C
- You will **not** design it for tolerating 200 °C

Adversarial != Hostile

- **Non-SW artifacts: Hostile** environment:
  - Inputs may be "extreme" and perhaps "uncommon"

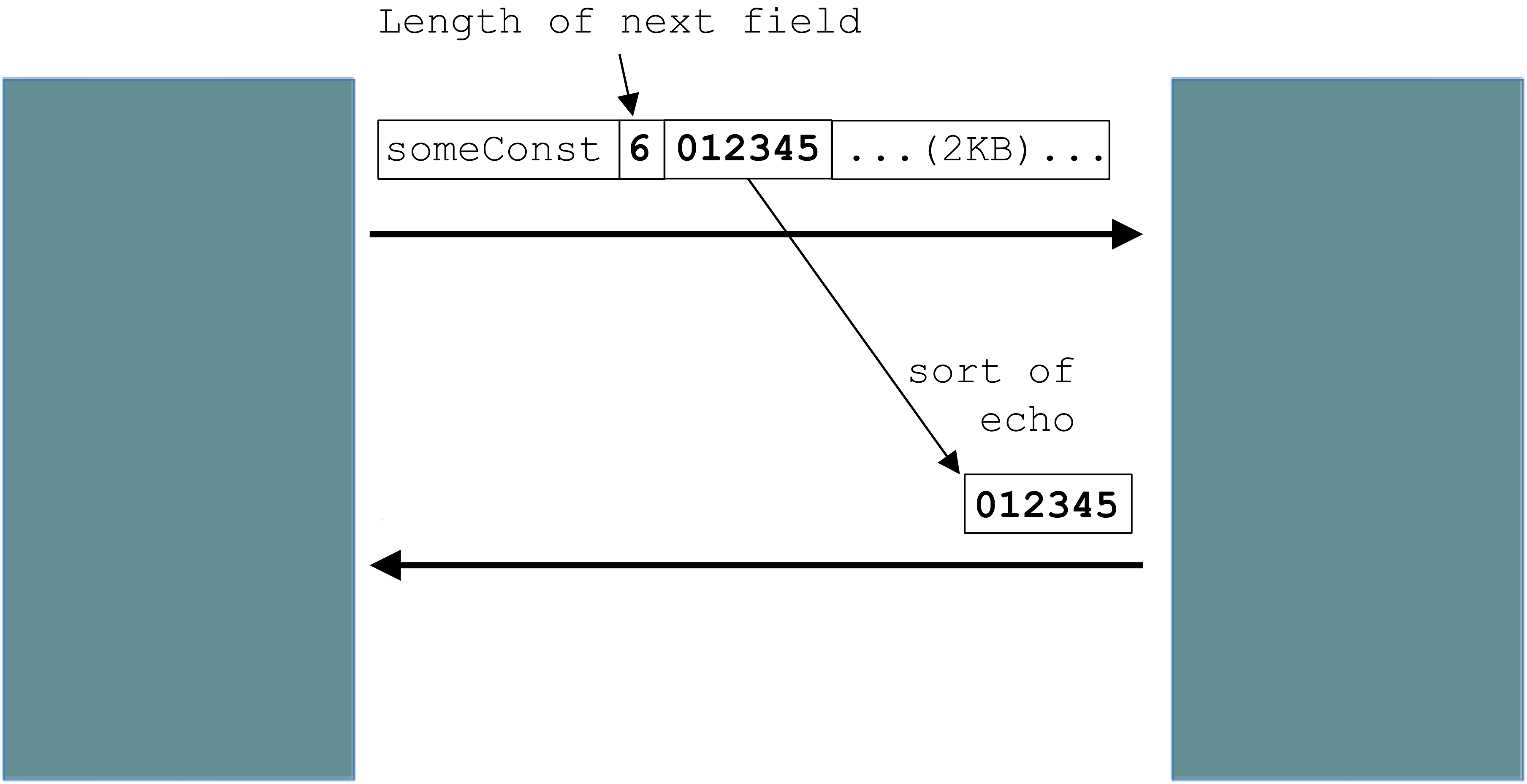
- **SW artifacts: Adversarial** environment:
  - Inputs **actively** and **carefully selected** to provoke undesired behavior
  - "The worst possible input, at the worst possible time"

Keep in mind

- Non-SW engineers must cope with **hostile** environments
- SW engineers must cope with **adversarial** environments
- They play different games!

Another Interesting Example

A simple network protocol (I)



Length of next field

someConst 6 **012345** ...(2KB)...

sort of echo

012345

A simple network protocol (II)

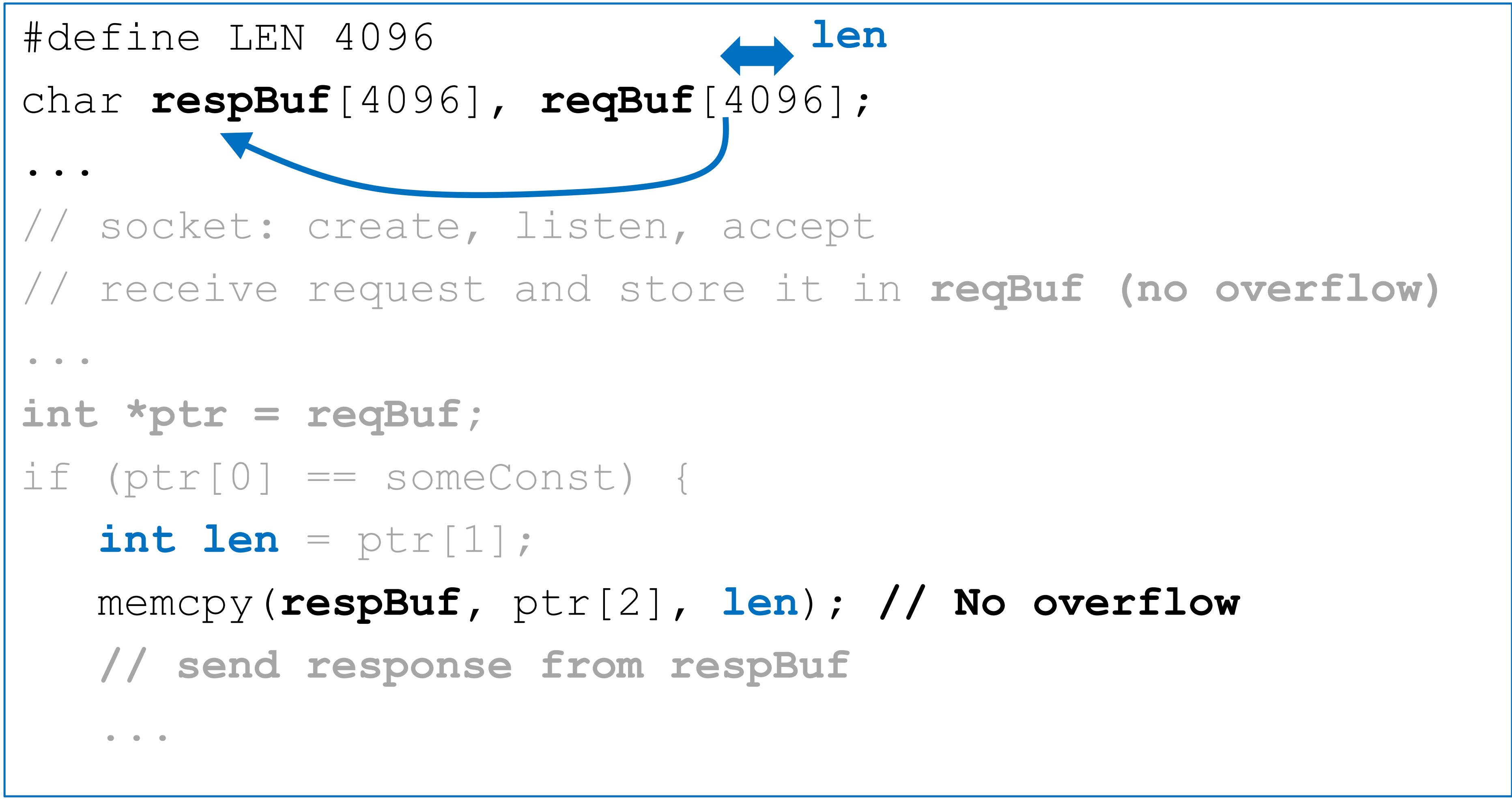
- Request message:
  1. Integer `someConst` (predefined constant)
  2. Integer `echo_length`
  3. Byte array with `echo_length` elements
  4. Byte array with 2048 elements
  - Field 3 cannot be longer than 1024 elements

- Response message:
  1. Byte array in field 3 of request

Server Implementation Outline

```
#define LEN 4096
char respBuf[4096], reqBuf[4096];
...
// socket: create, listen, accept
// receive request and store it in reqBuf (no overflow)
...
int *ptr = reqBuf;
if (ptr[0] == someConst) {
   int len = ptr[1];
   memcpy(respBuf, ptr[2], len); // No overflow
   // send response from respBuf
   ...
```

What we are doing



```
#define LEN 4096
char respBuf[4096], reqBuf[4096];
...
// socket: create, listen, accept
// receive request and store it in reqBuf (no overflow)
...
int *ptr = reqBuf;
if (ptr[0] == someConst) {
    int len = ptr[1];
    memcpy(respBuf, ptr[2], len); // No overflow
    // send response from respBuf
    ...
```

Ouch…! (I)

len

```
#define LEN 4096
char respBuf[4096], reqBuf[4096];
...
// create and connect socket s
// receive request and store it in reqBuf (no overflow)
...
int *ptr = reqBuf;
if (ptr[0] == someConst) {
    int len = ptr[1];                   // ADVERSARY-CONTROLLED
    memcpy(respBuf, ptr[2], len); // Read overflow
    // send response from respBuf
    ...
```

Information Disclosure

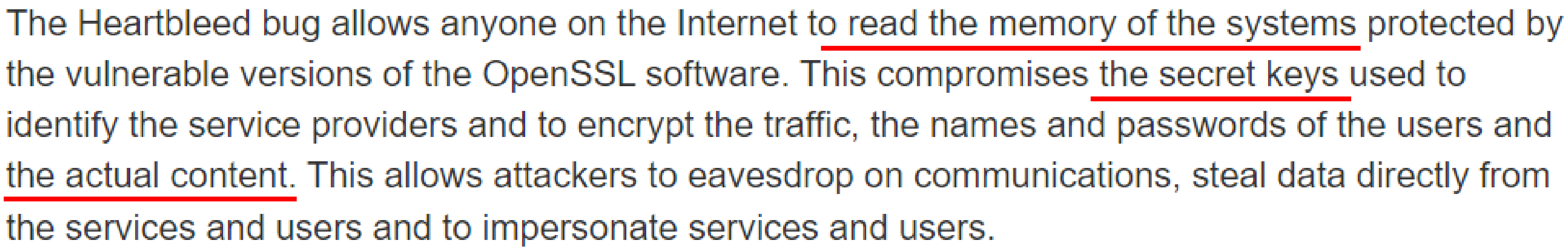
```
#define LEN 4096
char respBuf[4096], reqBuf[4096];
...
// create and connect socket s
// receive request and store it in                     low)
...
int *ptr = reqBuf;
if (ptr[0] == someConst) {
    int len = ptr[1];                 // ADVERSARY-CONTROLLED
    memcpy(respBuf, ptr[2], len);  // Read overflow
    // send response from respBuf
    ...
```

Some variables will be copied to respBuf...

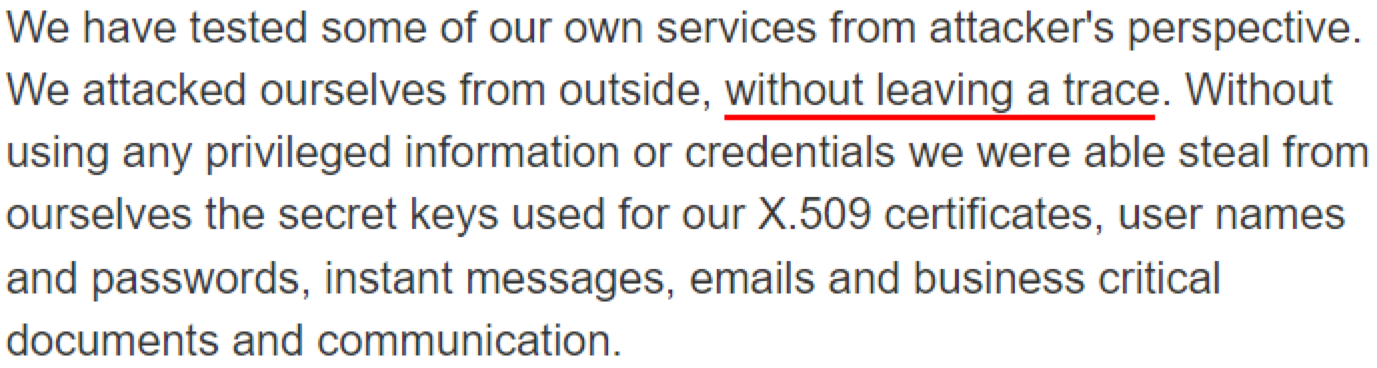
...and sent to the client

Catastrophe!

Basically **the same** approach led to:
Heartbleed (Specific Systems Affected) – Wikipedia

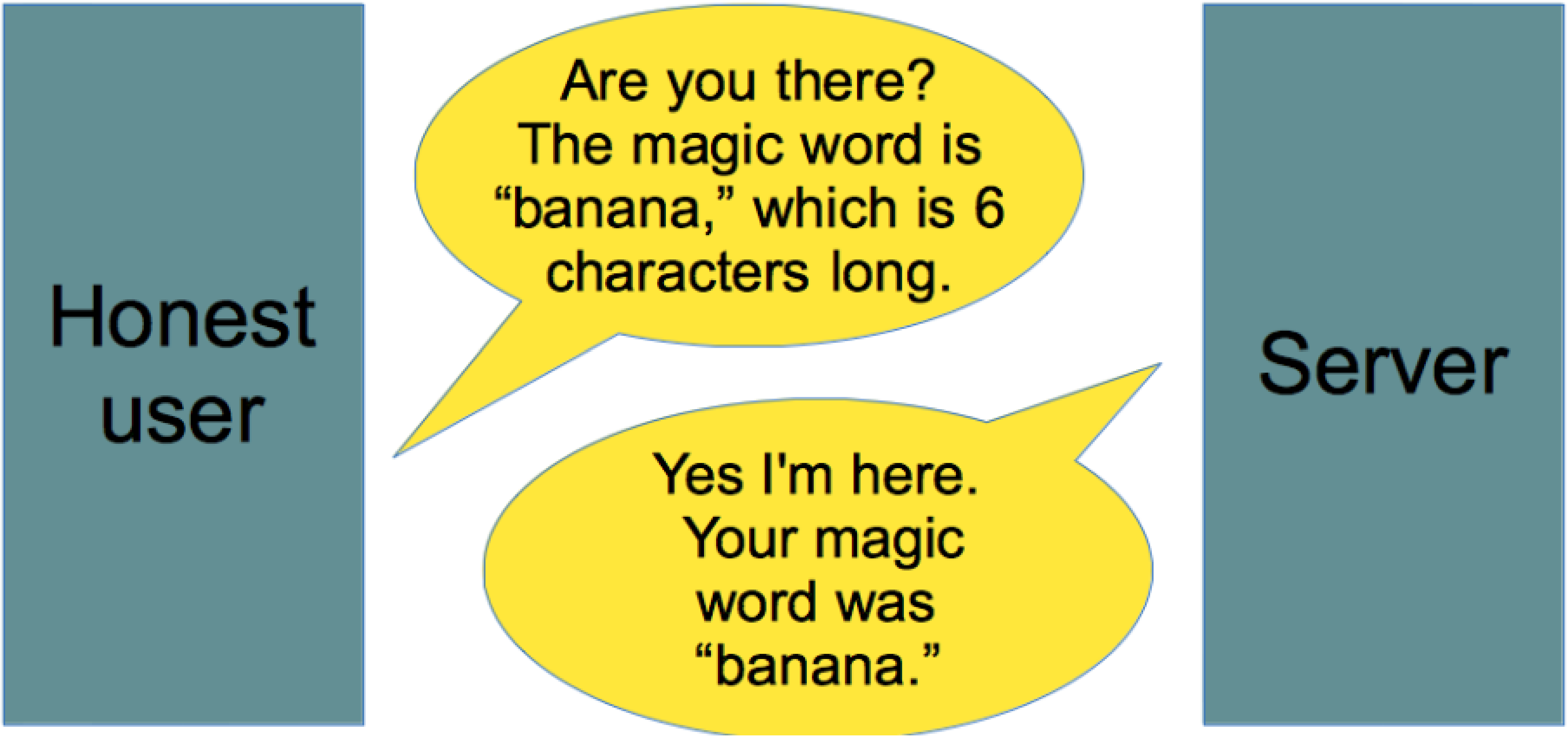


The Heartbleed bug allows anyone on the Internet to read the memory of the systems protected by the vulnerable versions of the OpenSSL software. This compromises the secret keys used to identify the service providers and to encrypt the traffic, the names and passwords of the users and the actual content. This allows attackers to eavesdrop on communications, steal data directly from the services and users and to impersonate services and users.



We have tested some of our own services from attacker's perspective. We attacked ourselves from outside, without leaving a trace. Without using any privileged information or credentials we were able steal from ourselves the secret keys used for our X.509 certificates, user names and passwords, instant messages, emails and business critical documents and communication.

TLS Heartbeat (RFC 6520)



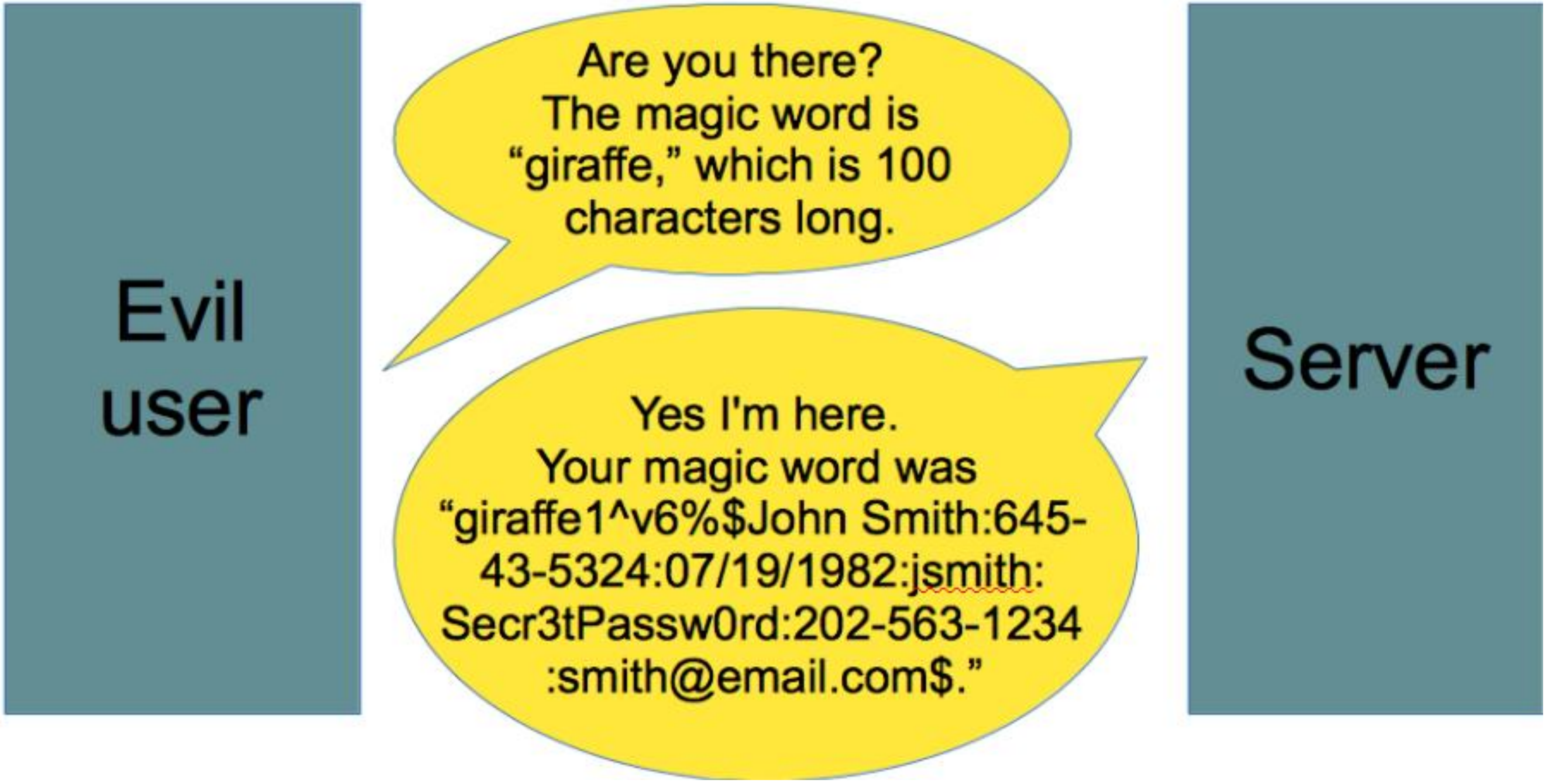
Honest user

Are you there? The magic word is “banana,” which is 6 characters long.

Yes I'm here. Your magic word was “banana.”

Server

OpenSSL Implementation



Evil user

Server

Are you there?
The magic word is
“giraffe,” which is 100
characters long.

Yes I'm here.
Your magic word was
“giraffe1^v6%$John Smith:645-43-5324:07/19/1982:jsmith:
Secr3tPassw0rd:202-563-1234
:smith@email.com$.”

Lessons Revisited

NEVER trust input data

- Never. Not even implicitly.

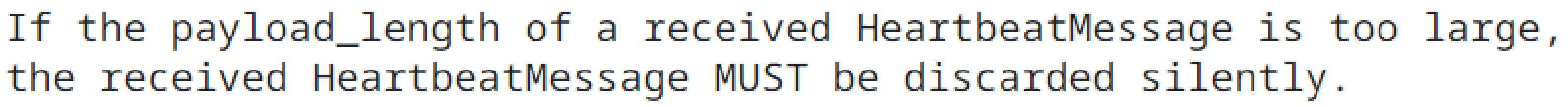
- **Structure** may not be what you expect
- **Values** may not be what you expect

- Much easier said than done

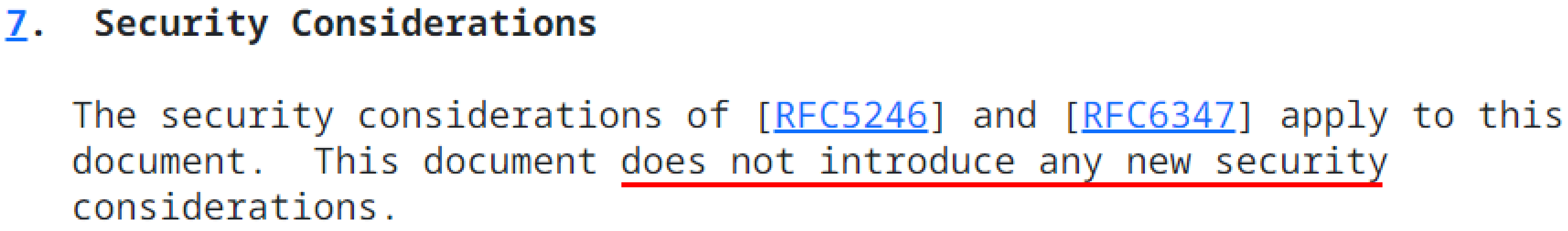
Never ONE reason

- Vulnerabilities never result from **one** single reason
- They always result from **many different** and **seemingly unrelated** reasons

- RFC 6520 was maybe not too explicit



If the payload_length of a received HeartbeatMessage is too large, the received HeartbeatMessage MUST be discarded silently.



7. Security Considerations

The security considerations of [RFC5246] and [RFC6347] apply to this document. This document does not introduce any new security considerations.

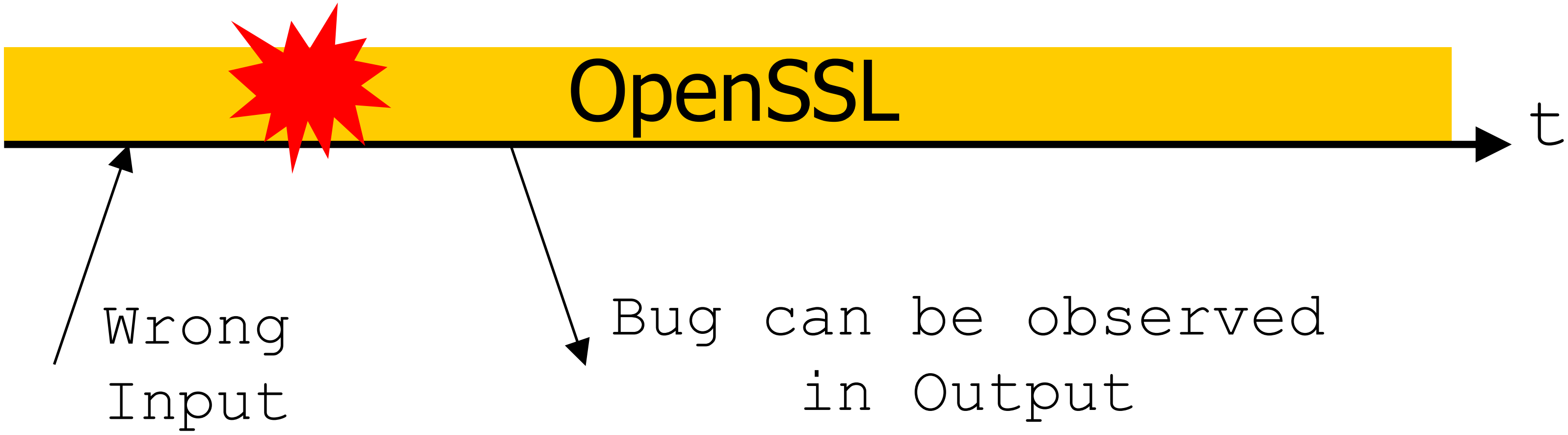
Testing

- More intensive / accurate testing might have discovered **a bug**
- It would have **not** discovered this **vulnerability automatically**

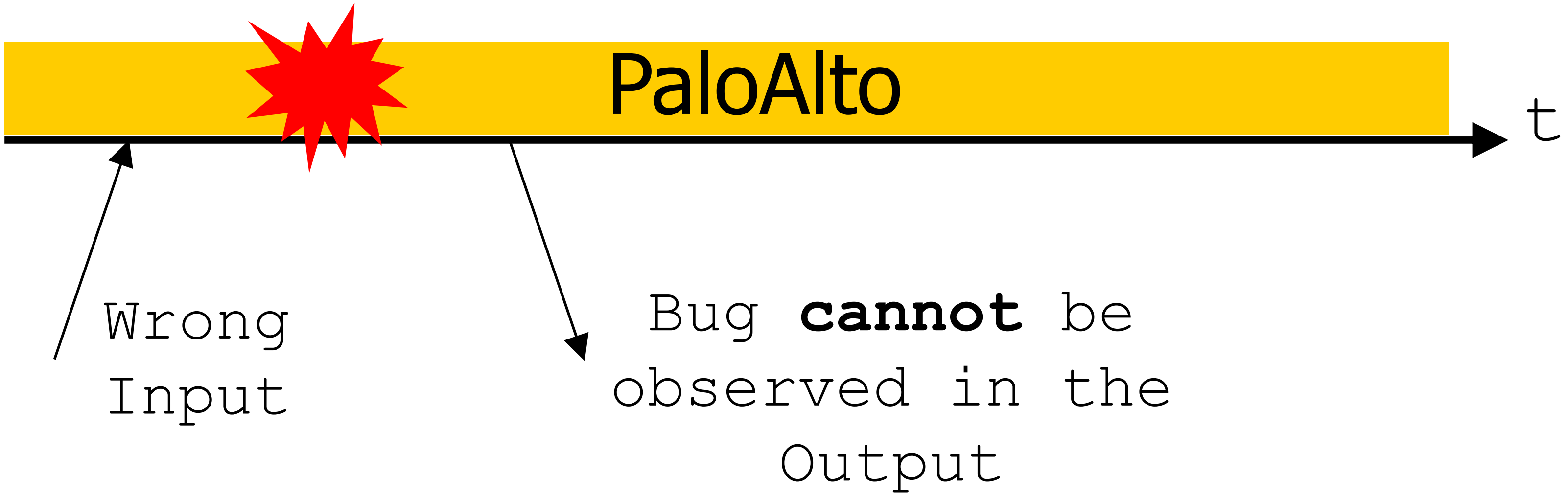
- This **vulnerability** maybe could have been discovered **automatically**

- OpenSSL only has two [fulltime] people to write, maintain, test, and review 500,000 lines of business critical code (2014)

Important remark about testing (I)



Important remark about testing



- Wrong input contaminates **internal state** but this is **not** visible on the corresponding output
- Discovery by testing enormously complex

You never know

- Unsafe practices are unsafe. Period.
- One cannot predict their reach

- Avoid them as much as you can

- **Memory unsafe programming languages** are an **unsafe practice**

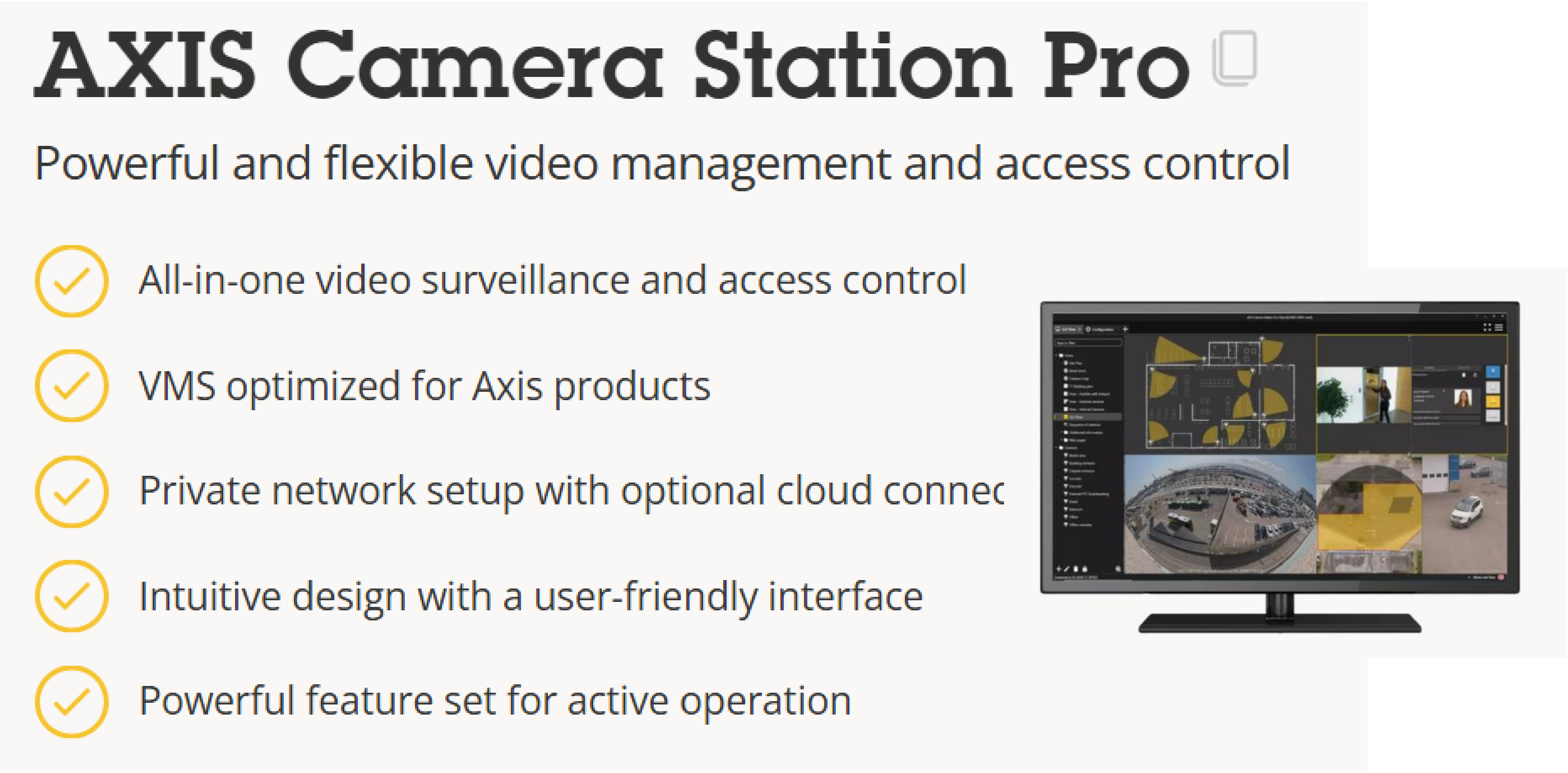
Back to Palo Alto vuln

- Was it caused by **memory safety** issues?
- Would a memory-safe programming language have helped?

More Examples

Axis Camera Station Pro (August 2025)

Axis Cameras (I)

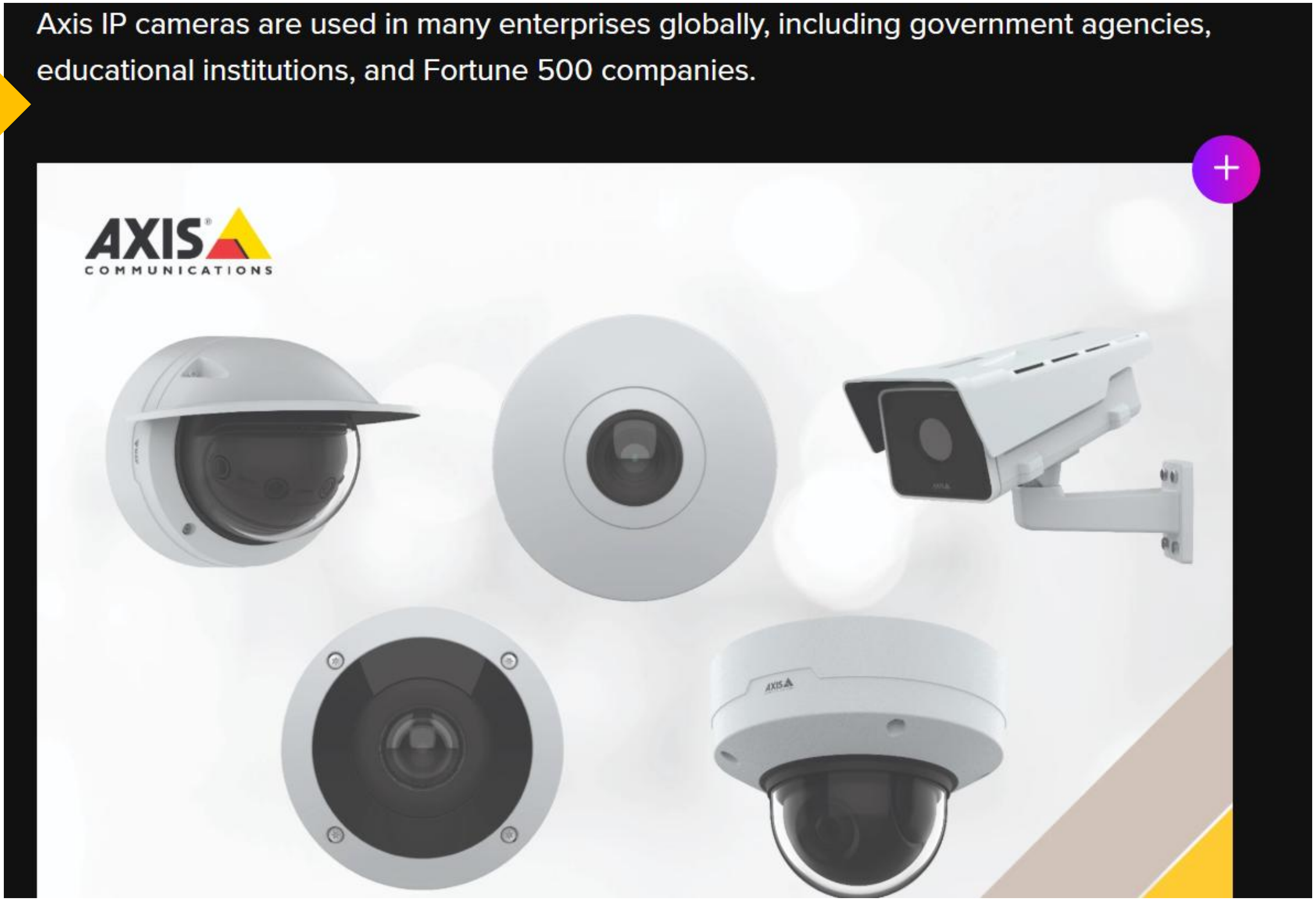


AXIS Camera Station Pro

Powerful and flexible video management and access control

- ✓ All-in-one video surveillance and access control
- ✓ VMS optimized for Axis products
- ✓ Private network setup with optional cloud connec
- ✓ Intuitive design with a user-friendly interface
- ✓ Powerful feature set for active operation

Axis Cameras (II)



Axis IP cameras are used in many enterprises globally, including government agencies, educational institutions, and Fortune 500 companies.

Client-Server Overview (I)

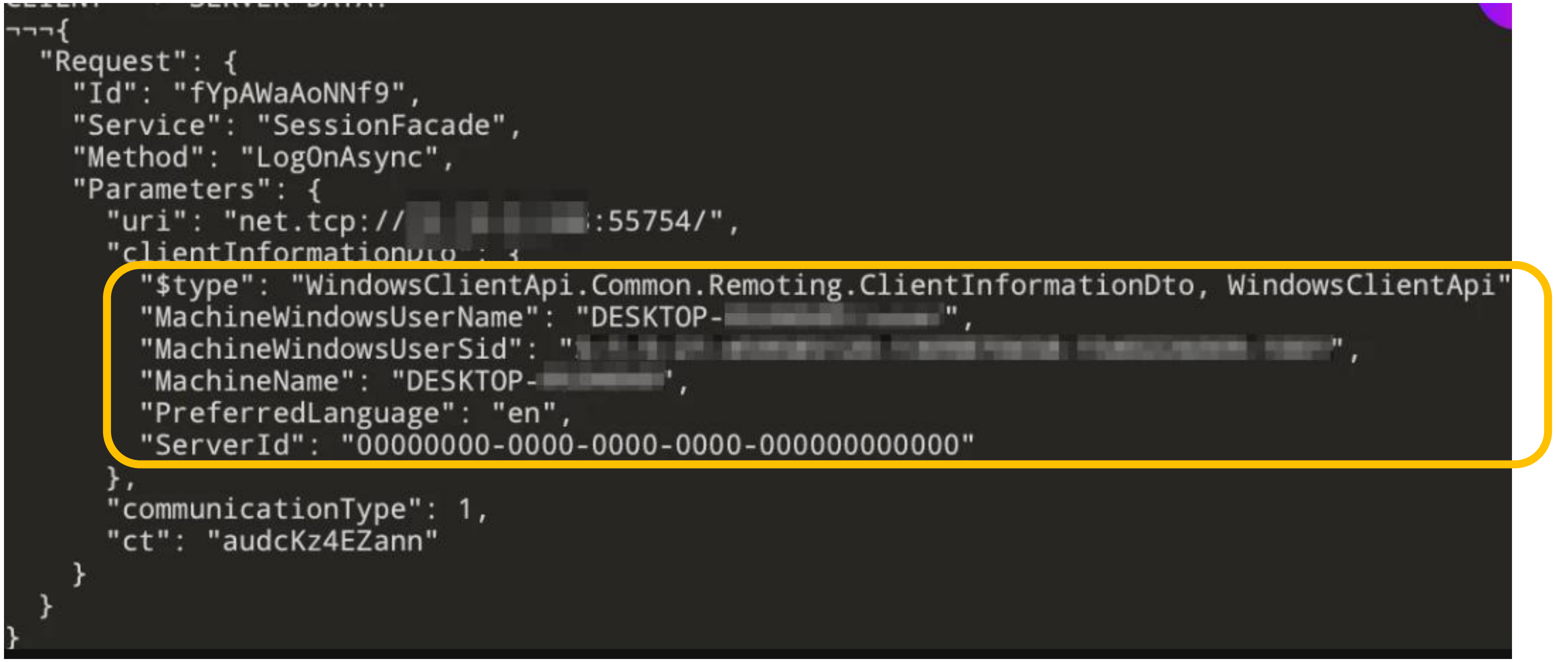
1. Client station connects to Server station with **HTTPS**
   - Secrecy
   - Integrity
   - **Mutual** authentication of the machines (stronger than common usage of HTTPS)

2. **Authentication** of the Client **account**

3. Client-Server **application** communication with Axis protocol

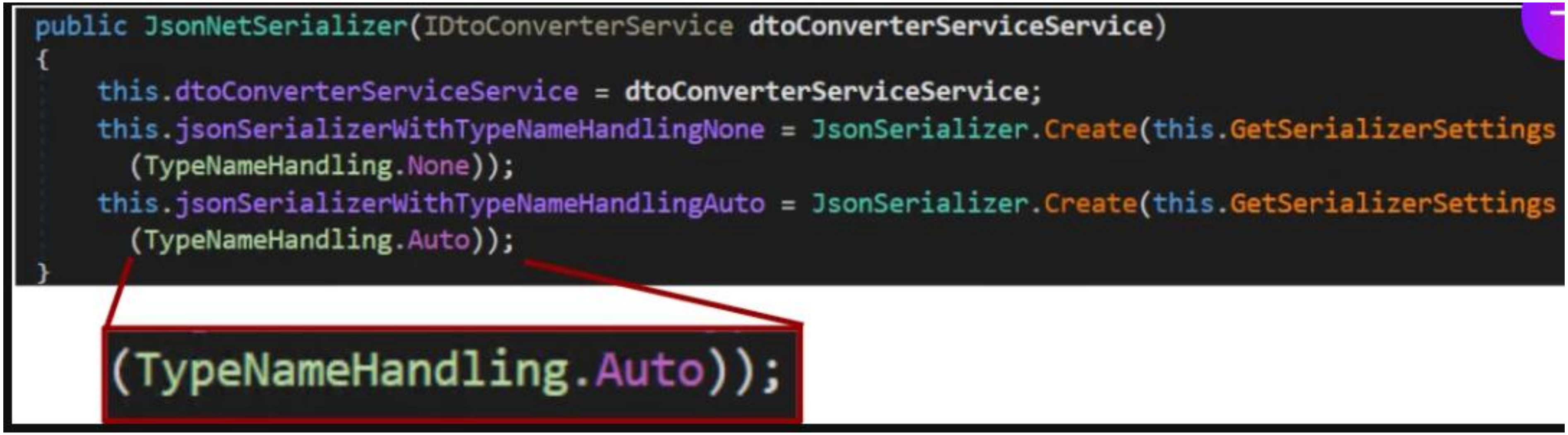
Client-Server Overview (II)

- Axis application protocol allows clients and servers to exchange **arbitrarily complex .NET objects** serialized in JSON
- Client sends object to Server that reconstructs it (and vice versa)



```
CLIENT -> SERVER DATA:
¬¬¬{
  "Request": {
    "Id": "fYpAWaAoNNf9",
    "Service": "SessionFacade",
    "Method": "LogOnAsync",
    "Parameters": {
      "uri": "net.tcp://[redacted]:55754/",
      "clientInformationDto": {
        "$type": "WindowsClientApi.Common.Remoting.ClientInformationDto, WindowsClientApi"
        "MachineWindowsUserName": "DESKTOP-[redacted]",
        "MachineWindowsUserSid": "[redacted]",
        "MachineName": "DESKTOP-[redacted]",
        "PreferredLanguage": "en",
        "ServerId": "00000000-0000-0000-0000-000000000000"
      },
      "communicationType": 1,
      "ct": "audcKz4EZann"
    }
  }
}
```

Deserialization



```
public JsonNetSerializer(IDtoConverterService dtoConverterServiceService)
{
    this.dtoConverterServiceService = dtoConverterServiceService;
    this.jsonSerializerWithTypeNameHandlingNone = JsonSerializer.Create(this.GetSerializerSettings
      (TypeNameHandling.None));
    this.jsonSerializerWithTypeNameHandlingAuto = JsonSerializer.Create(this.GetSerializerSettings
      (TypeNameHandling.Auto));
}
```

```
(TypeNameHandling.Auto));
```

- Receiver:
  1. Reads type of the serialized object
  2. Reconstructs object locally

BOOM!
Post-Auth RCE

- Receiver:
  1. Reads type of the serialized object
  2. Reconstructs object locally

- Receiver does **not** check whether the object has one of the **expected** types
  ⇒ It will reconstruct **whatever object** it happens to receive

- Demo video:
  - Object that while being reconstructed
    invokes Powershell commands automatically

NEVER trust input data (...boring...)

- Never. Not even implicitly.

- **Structure** may not be what you expect
- **Values** may not be what you expect

- Much easier said than done

Further weaknesses...

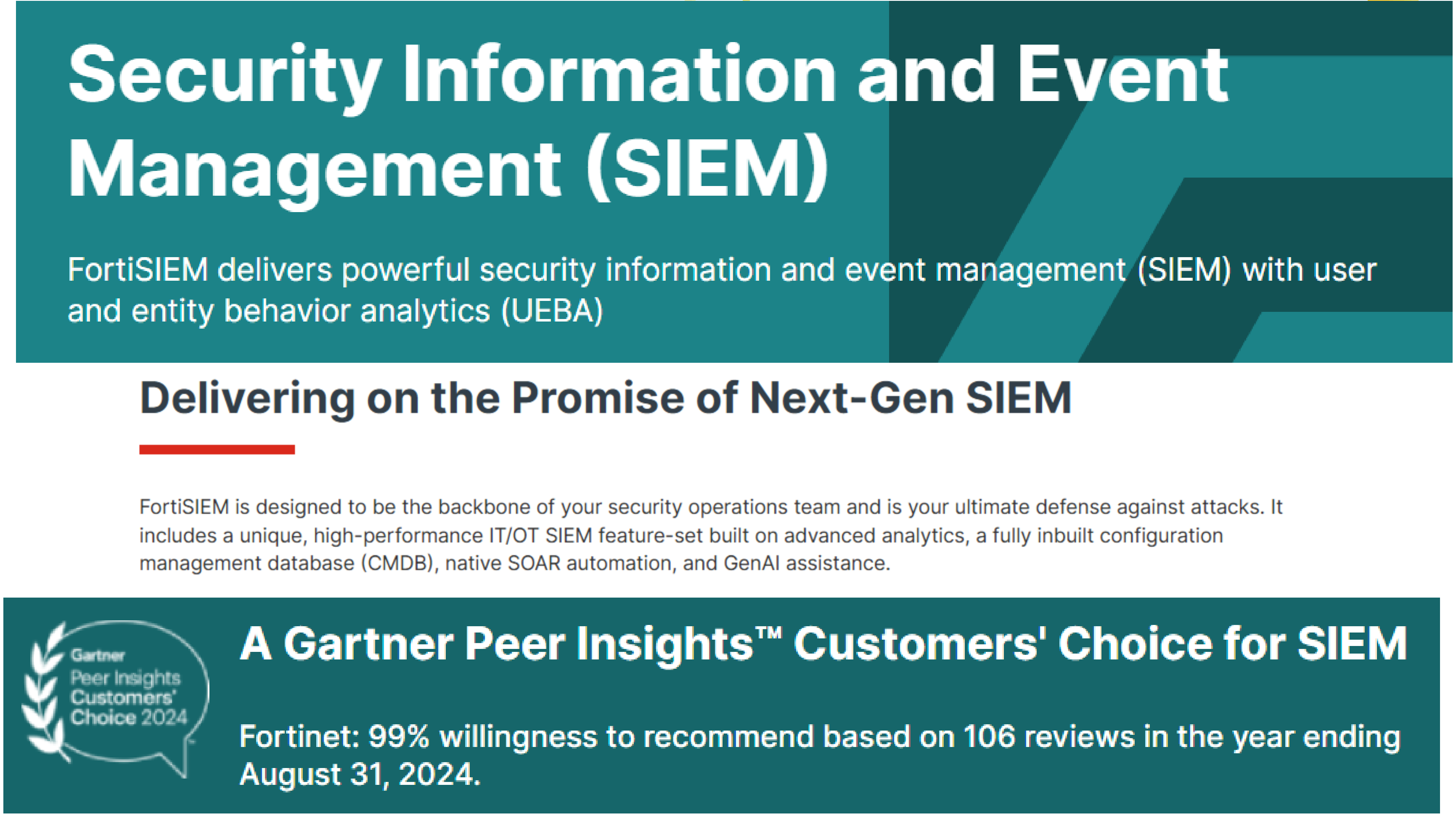
- Client and Server use **self-signed certificates**
- **Hidden** URL with **anonymous** authentication

↓

- Anyone can interact with any accessible Server **Pre-Auth(!)**
- ...and can execute code (Server runs as `SYSTEM`)

FortiNet SIEM (August 2025)

FortiNet SIEM (August 2025)



Security Information and Event Management (SIEM)

FortiSIEM delivers powerful security information and event management (SIEM) with user and entity behavior analytics (UEBA)

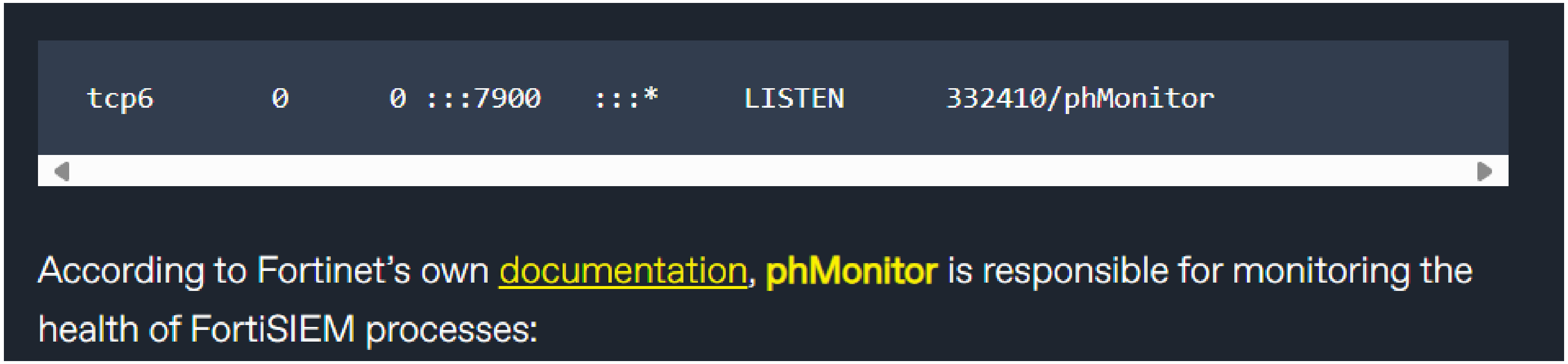
Delivering on the Promise of Next-Gen SIEM

FortiSIEM is designed to be the backbone of your security operations team and is your ultimate defense against attacks. It includes a unique, high-performance IT/OT SIEM feature-set built on advanced analytics, a fully inbuilt configuration management database (CMDB), native SOAR automation, and GenAI assistance.

A Gartner Peer Insights™ Customers' Choice for SIEM

Fortinet: 99% willingness to recommend based on 106 reviews in the year ending August 31, 2024.

Monitoring Process (I)

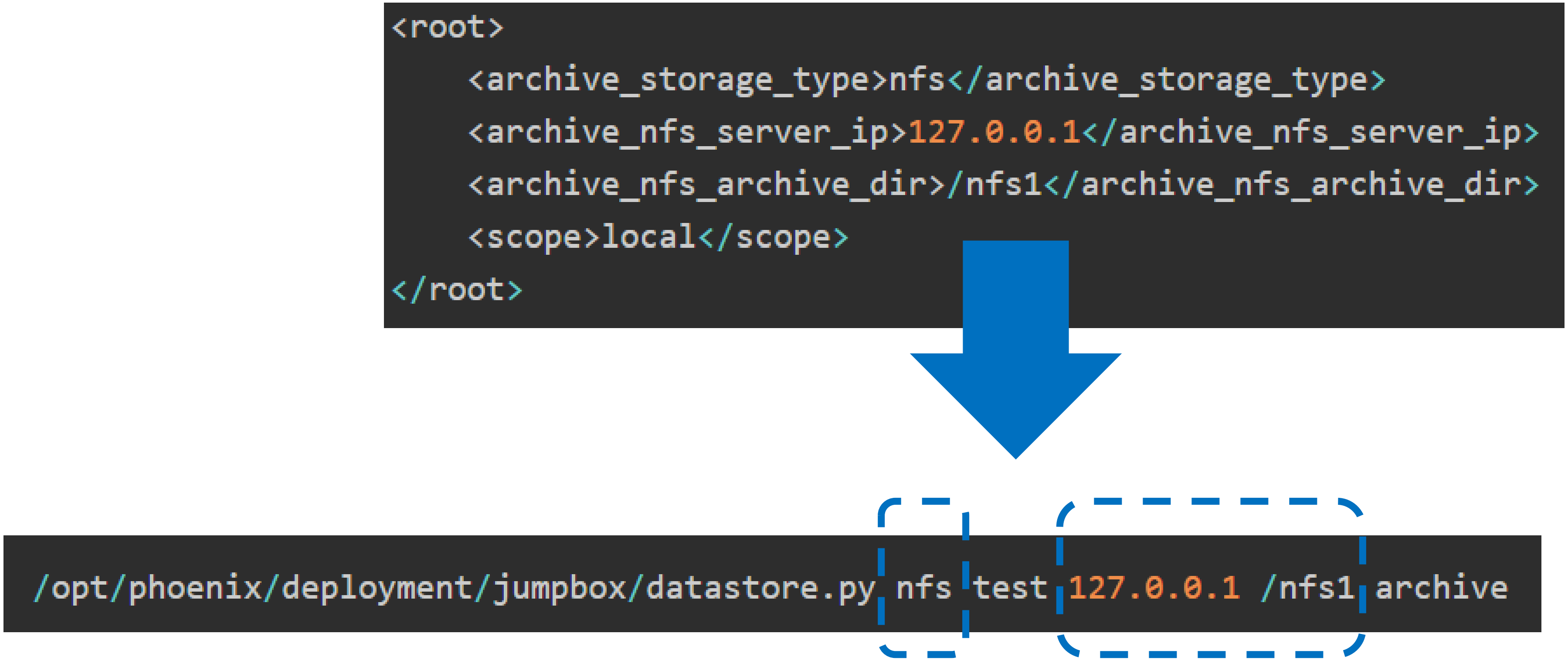


```
tcp6       0      0 :::7900    :::*      LISTEN      332410/phMonitor
```

According to Fortinet's own documentation, **phMonitor** is responsible for monitoring the health of FortiSIEM processes:

- This process has an endpoint that **invokes a shell command**
- Part of the command is extracted from the serialized **XML content** received

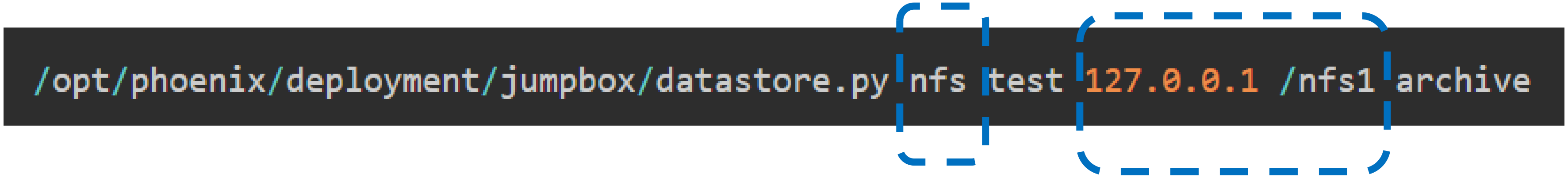
Monitoring Process (II)



```
<root>
    <archive_storage_type>nfs</archive_storage_type>
    <archive_nfs_server_ip>127.0.0.1</archive_nfs_server_ip>
    <archive_nfs_archive_dir>/nfs1</archive_nfs_archive_dir>
    <scope>local</scope>
</root>
```

```
/opt/phoenix/deployment/jumpbox/datastore.py nfs test 127.0.0.1 /nfs1 archive
```

Input sanitization



```
/opt/phoenix/deployment/jumpbox/datastore.py nfs test 127.0.0.1 /nfs1 archive
```

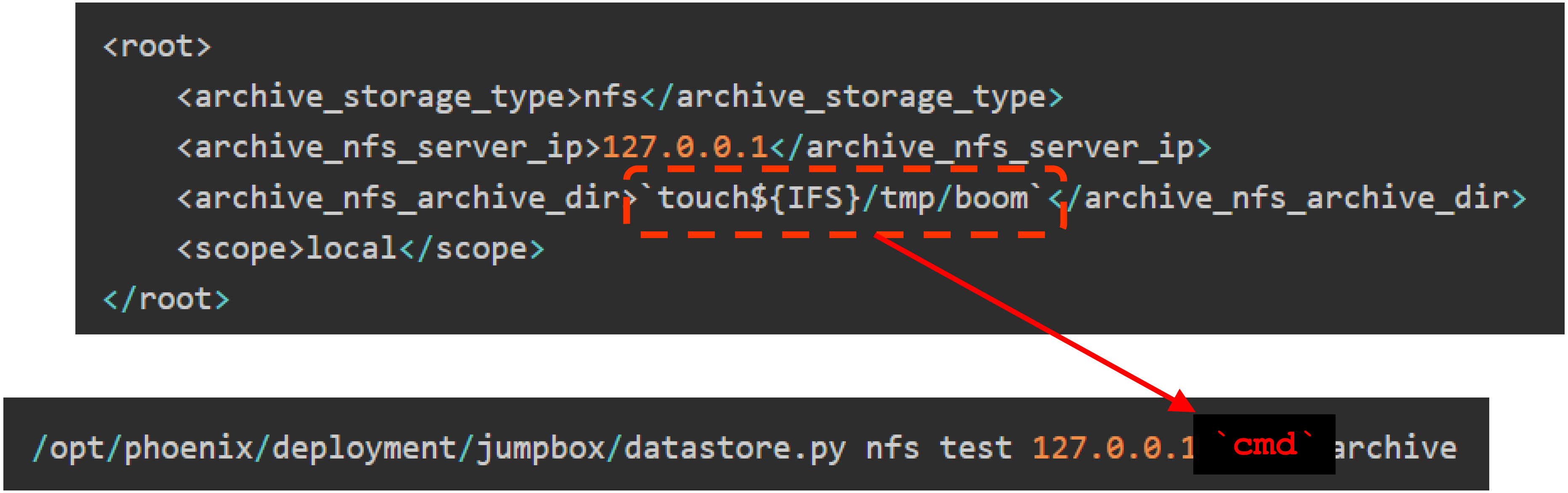
```
String cmd = "/opt/phoenix.../datastore.py" +
          v1 + "test" + v2 + v3 + "archive"
/* spawn shell that executes cmd */
...
```

- cmd contains input data

- Shell special characters are removed from input data to make sure it will **never** be interpreted as a **command**

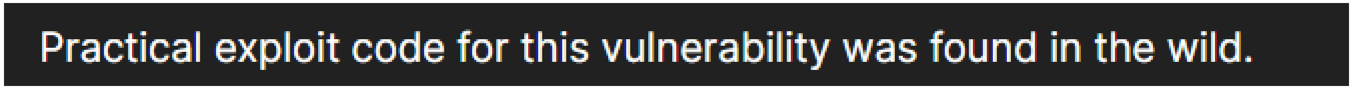
Very weak sanitization→RCE

- Shell special characters are removed from input data to make sure it will **never** be interpreted as a **command**
- Backticks not escaped (!)



```
<root>
    <archive_storage_type>nfs</archive_storage_type>
    <archive_nfs_server_ip>127.0.0.1</archive_nfs_server_ip>
    <archive_nfs_archive_dir>`touch${IFS}/tmp/boom`</archive_nfs_archive_dir>
    <scope>local</scope>
</root>
```

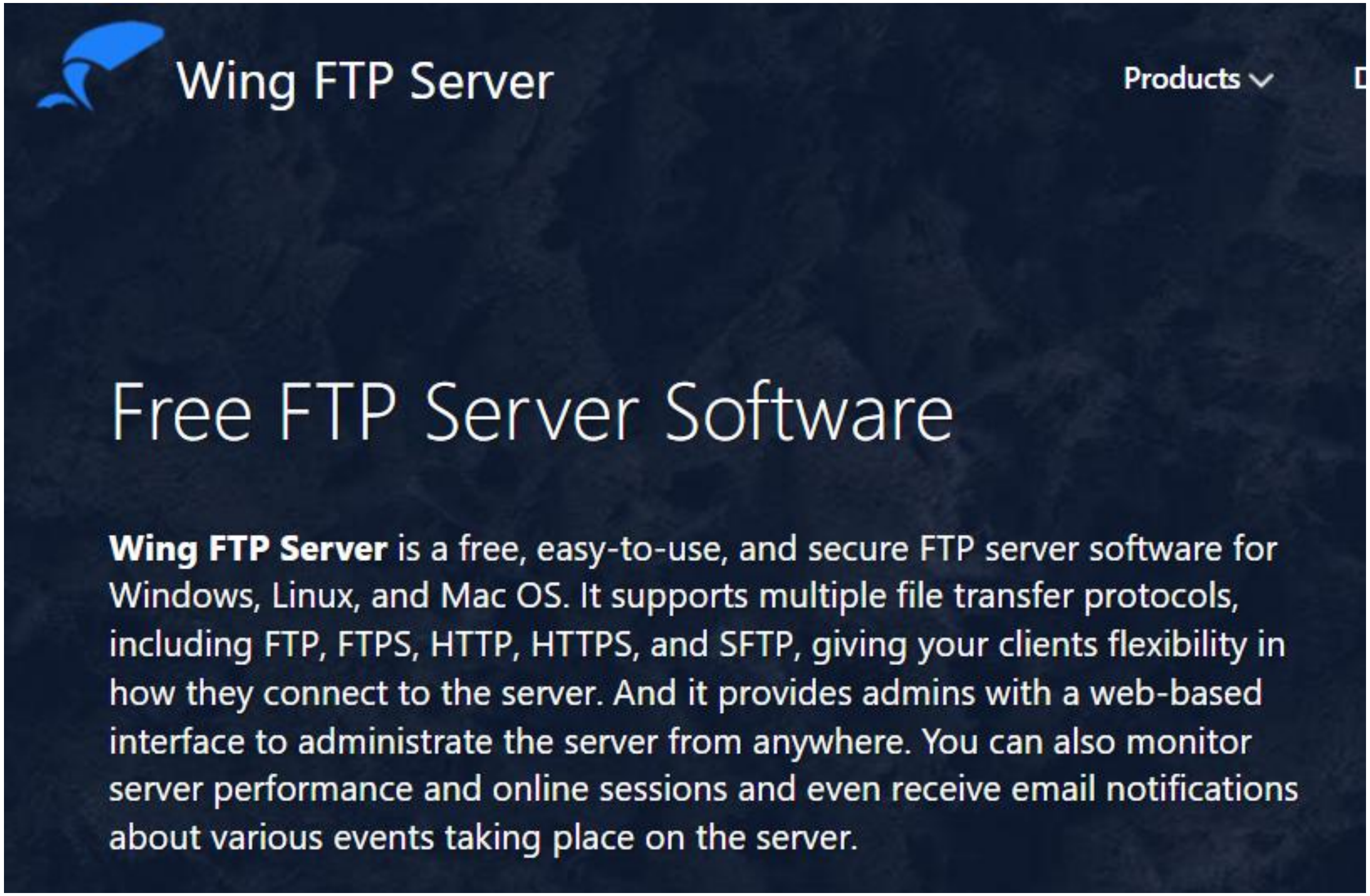
```
/opt/phoenix/deployment/jumpbox/datastore.py nfs test 127.0.0.1 `cmd` archive
```



Practical exploit code for this vulnerability was found in the wild.

Wing FTP Server (June 2025)

Wing FTP Server (June 2025)



Free FTP Server Software

Wing FTP Server is a free, easy-to-use, and secure FTP server software for Windows, Linux, and Mac OS. It supports multiple file transfer protocols, including FTP, FTPS, HTTP, HTTPS, and SFTP, giving your clients flexibility in how they connect to the server. And it provides admins with a web-based interface to administrate the server from anywhere. You can also monitor server performance and online sessions and even receive email notifications about various events taking place on the server.

Session Management

- Written in LUA

- Saving session state:
  1. LUA serialization
     (session objects linearized in byte stream)
  2. File write

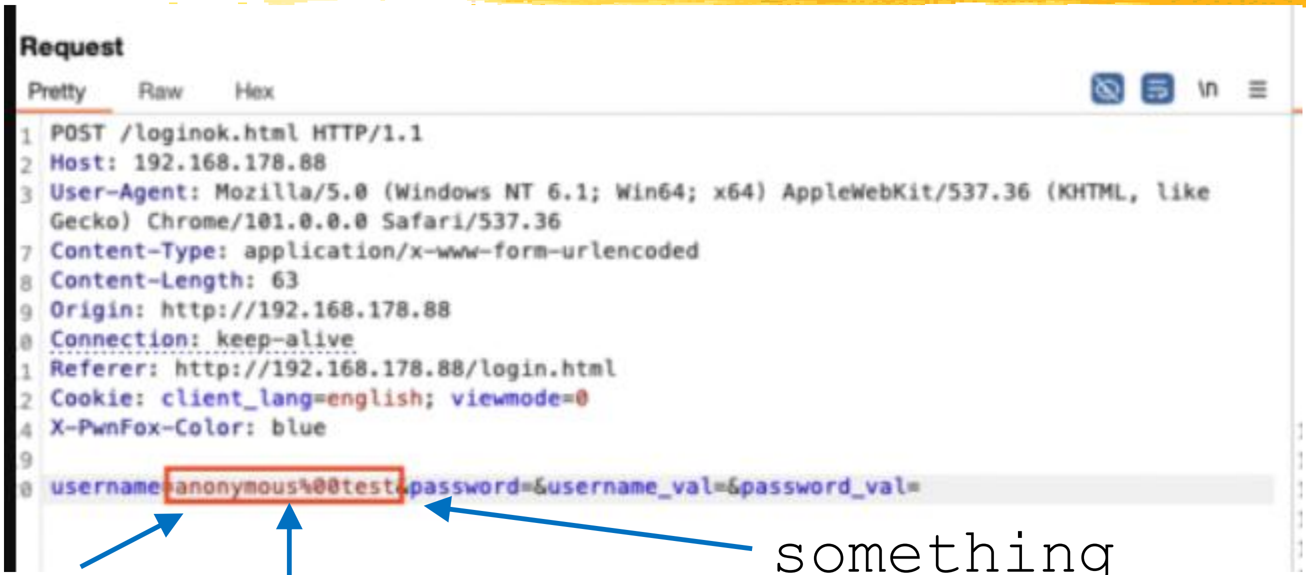
- Restoring session state:
  1. File read
  2. LUA deserialization
     (serialized objects reconstructed from byte stream)

Nice feature of LUA

- Strings are **not NULL-terminated**
- They always have **length** information at runtime

- **Prevent** many **overflow bugs** common in C/C++

Tiny little bug



```
POST /loginok.html HTTP/1.1
Host: 192.168.178.88
User-Agent: Mozilla/5.0 (Windows NT 6.1; Win64; x64) AppleWebKit/537.36 (KHTML, like
Gecko) Chrome/101.0.0.0 Safari/537.36
Content-Type: application/x-www-form-urlencoded
Content-Length: 63
Origin: http://192.168.178.88
Connection: keep-alive
Referer: http://192.168.178.88/login.html
Cookie: client_lang=english; viewmode=0
X-PwnFox-Color: blue

username=anonymous%00test&password=&username_val=&password_val=
```

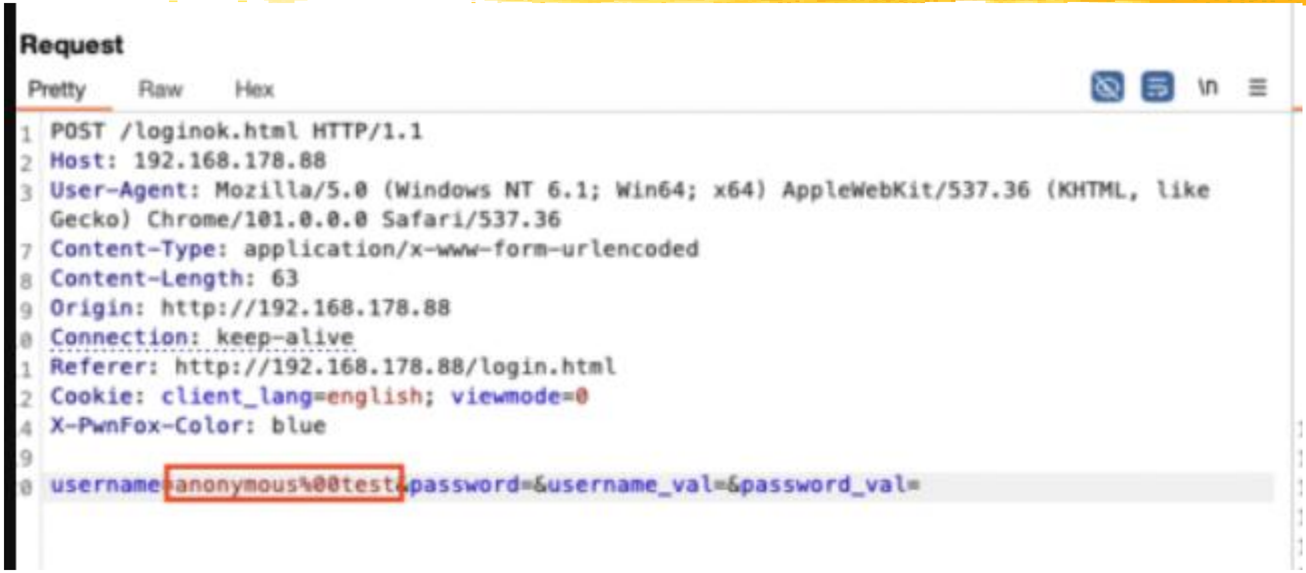
`username`

`NULL(URL-encoded)`

`something`

- IF `username` is in AuthDB
- THEN server code uses **entire parameter value** (including NULL and what follows)

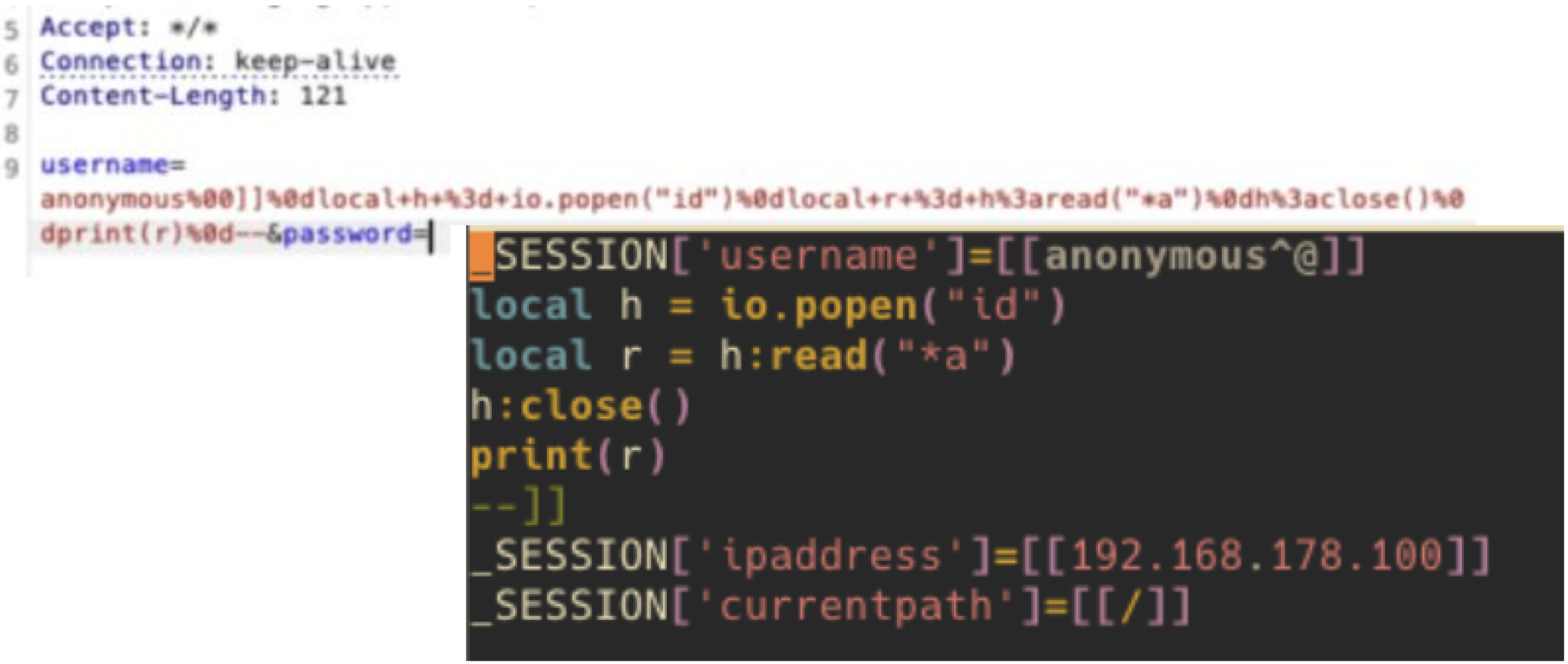
Other point of view



```
POST /loginok.html HTTP/1.1
Host: 192.168.178.88
User-Agent: Mozilla/5.0 (Windows NT 6.1; Win64; x64) AppleWebKit/537.36 (KHTML, like
Gecko) Chrome/101.0.0.0 Safari/537.36
Content-Type: application/x-www-form-urlencoded
Content-Length: 63
Origin: http://192.168.178.88
Connection: keep-alive
Referer: http://192.168.178.88/login.html
Cookie: client_lang=english; viewmode=0
X-PwnFox-Color: blue

username=anonymous%00test&password=&username_val=&password_val=
```

- Input data is **implicitly trusted** to have the **expected structure**
- Why should an username have a trailing NULL followed by something else?

BOOM!
Post-Auth RCE

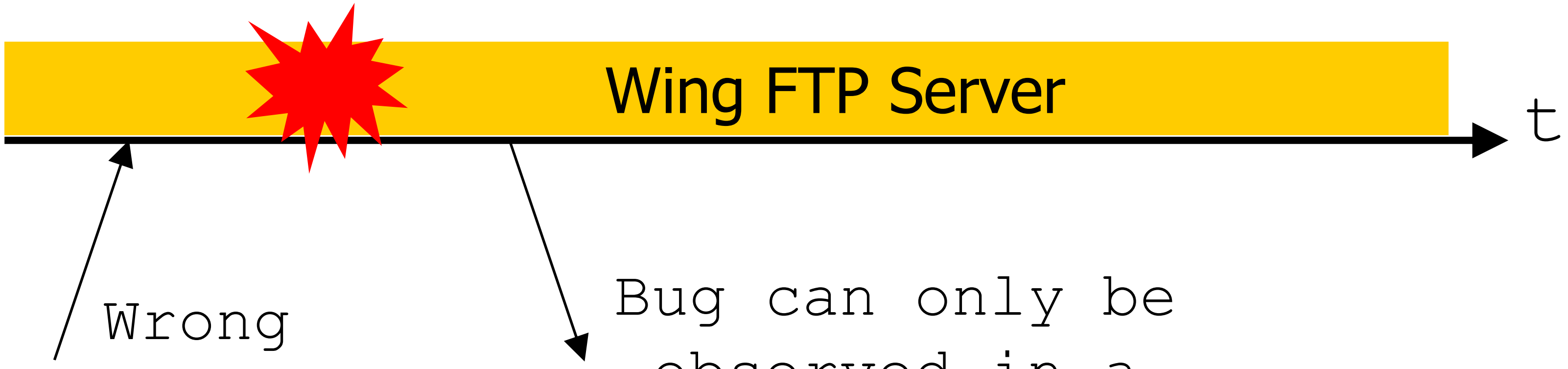


```
Accept: */*
Connection: keep-alive
Content-Length: 121

username=
anonymous%00]]%0dlocal+h+%3d+io.popen("id")%0dlocal+r+%3d+h%3aread("*a")%0dh%3aclose()%0
dprint(r)%0d--&password=
```

```
_SESSION['username']=[[anonymous^@]]
local h = io.popen("id")
local r = h:read("*a")
h:close()
print(r)
--]]
_SESSION['ipaddress']=[[192.168.178.100]]
_SESSION['currentpath']=[[/]]
```

- ❑ LUA code **injected** in session file
- ❑ It will be executed upon deserializing

Important remark about testing



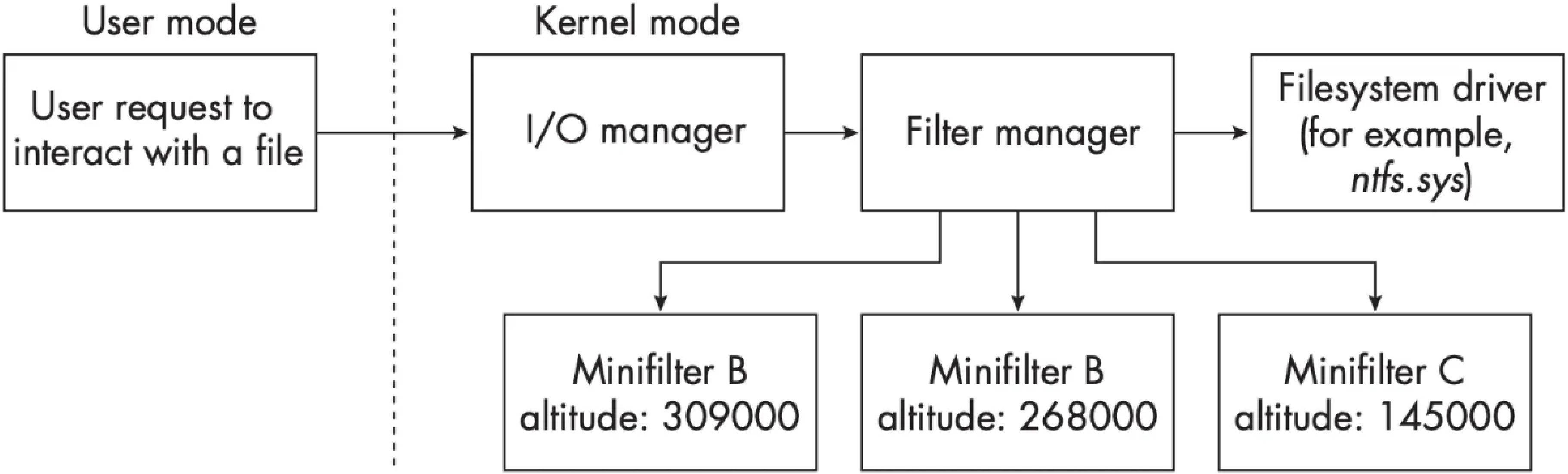
Bug can only be observed in a **future Output**

- Wrong input contaminates **internal state** but this is **not** visible on the corresponding output

- Discovery by testing enormously complex

Windows Cloud Files Privilege Escalation (October 2025)

Windows
Mini Filter Drivers (I)


User mode
Kernel mode
User request to interact with a file
I/O manager
Filter manager
Filesystem driver (for example, *ntfs.sys*)
Minifilter B altitude: 309000
Minifilter B altitude: 268000
Minifilter C altitude: 145000

Again, Image taken from Evading EDRs book. Thanks to Matt Hand :)

- They **monitor**, **filter**, and/or **modify** the behavior of file I/O operations.
- By **intercepting requests** targeted at a file system ..., a minifilter can extend or replace functionality provided by the original target of the request.

Windows
Mini Filter Drivers (II)

- **Anti-virus/anti-malware utilities** that scan files for viruses or malware on file creation, access, or modification.
- **Data encryption programs** that automatically encrypt and decrypt data being written to or read from disk.
- **Backup** filters used to create backups of data for disaster recovery and business continuity.
- **Compression** filters that compress and decompress data on the fly, which saves disk space and improves performance
- Mirroring / Streaming directories on **cloud storage** (e.g., OneDrive)

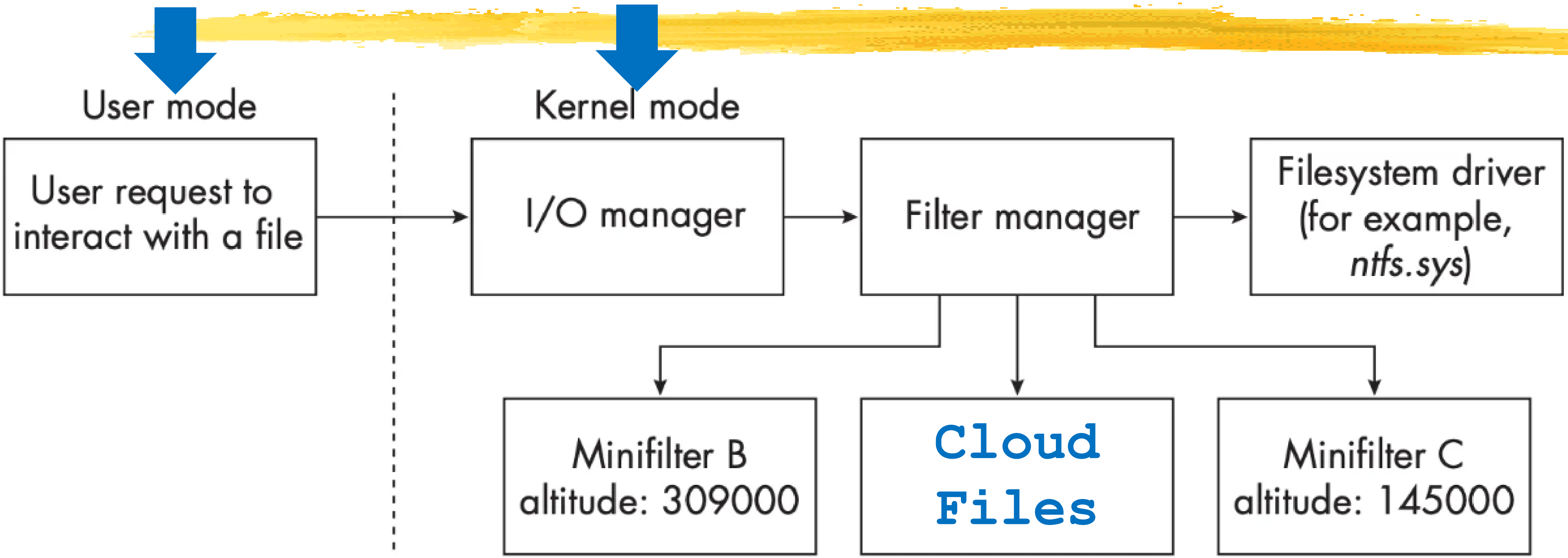
Cloud Files

Mini Filter Driver (I)

- Many complex operations
- ...
- Creates a **placeholder file** of a **specified filename**
- ...
- Called many times for mirroring a "sync root"

Cloud Files

Mini Filter Driver (II)


User mode
Kernel mode
User request to interact with a file
I/O manager
Filter manager
Filesystem driver (for example, *ntfs.sys*)
Minifilter B altitude: 309000
Cloud Files
Minifilter C altitude: 145000

Again, Image taken from Evading EDRs book. Thanks to Matt Hand :)

- Request parameters are **memory-mapped** to System memory
- Different virtual addresses mapped to the same physical memory

CVE-2020-1170 Cloud Filter Arbitrary File Creation

- ...leading to **privilege escalation**

- Create a directory `SOMEDIR` as a junction (symlink) to a system directory
- Specify `SOMEDIR\`**`evilfile.dll`** as filename

↓

- Minifilter will follow the junction and create **`evilfile.dll`** in **Attacker-chosen system directory**
  - Minifilter does not have any programmatic access check
  - Minifilter runs in Kernel mode

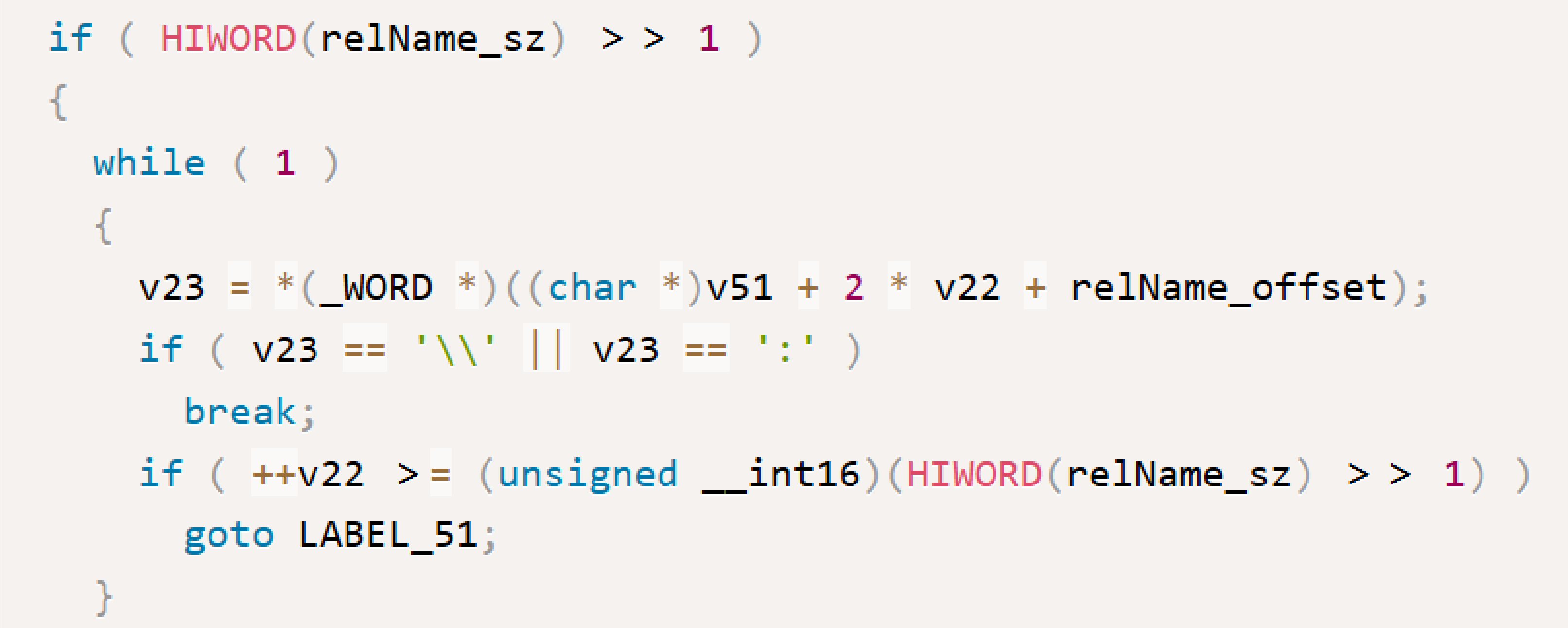
CVE-2020-1170 patched

- Many complex operations

- ...
- Creates a **placeholder file** of a **specified filename**
  - First checks that the **specified filename** does **not** contain character \ or :
- ...

Cloud Files
Mini Filter Driver: Check

- If first checks that the **specified filename** does not contain characters \ or :

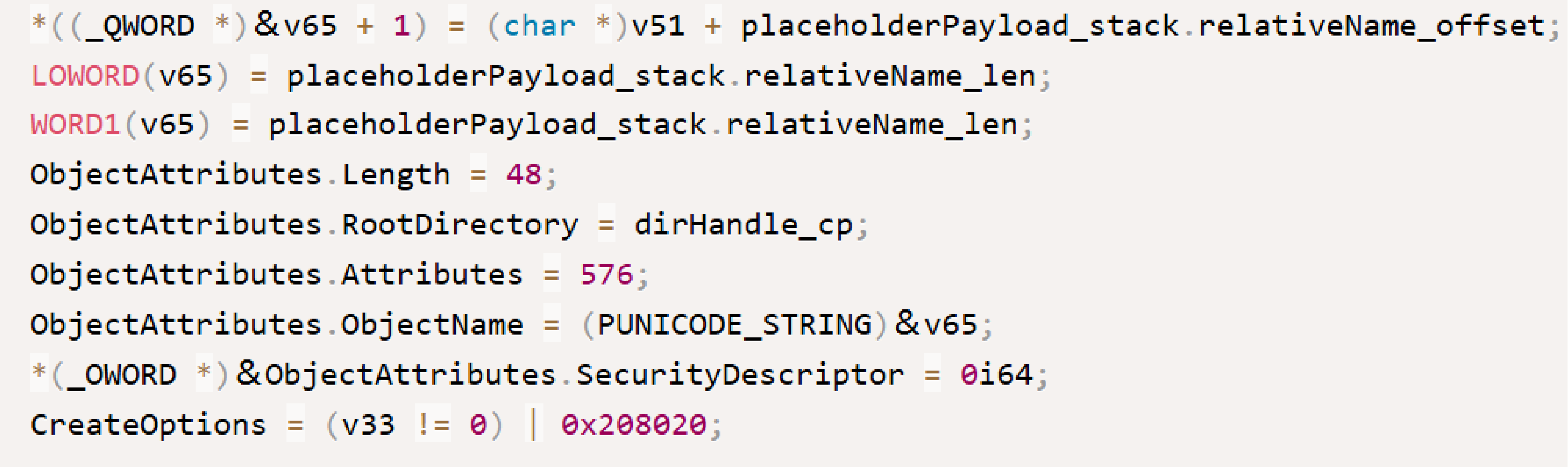


```
if ( HIWORD(relName_sz) >> 1 )
{
  while ( 1 )
  {
    v23 = *(_WORD *)((char *)v51 + 2 * v22 + relName_offset);
    if ( v23 == '\\' || v23 == ':' )
      break;
    if ( ++v22 >= (unsigned __int16)(HIWORD(relName_sz) >> 1) )
      goto LABEL_51;
  }
```

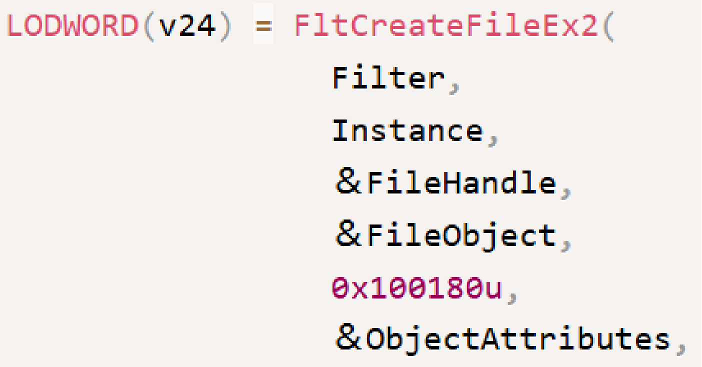
Cloud Files

Mini Filter Driver : Create

- Creates a **placeholder file** of a **specified filename**



```
*((_QWORD *)&v65 + 1) = (char *)v51 + placeholderPayload_stack.relativeName_offset;
LOWORD(v65) = placeholderPayload_stack.relativeName_len;
WORD1(v65) = placeholderPayload_stack.relativeName_len;
ObjectAttributes.Length = 48;
ObjectAttributes.RootDirectory = dirHandle_cp;
ObjectAttributes.Attributes = 576;
ObjectAttributes.ObjectName = (PUNICODE_STRING)&v65;
*(_OWORD *)&ObjectAttributes.SecurityDescriptor = 0i64;
CreateOptions = (v33 != 0) | 0x208020;
```

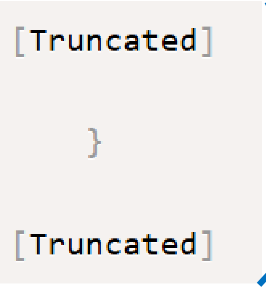


```
LODWORD(v24) = FltCreateFileEx2(
                 Filter,
                 Instance,
                 &FileHandle,
                 &FileObject,
                 0x100180u,
                 &ObjectAttributes,
```

Cloud Files

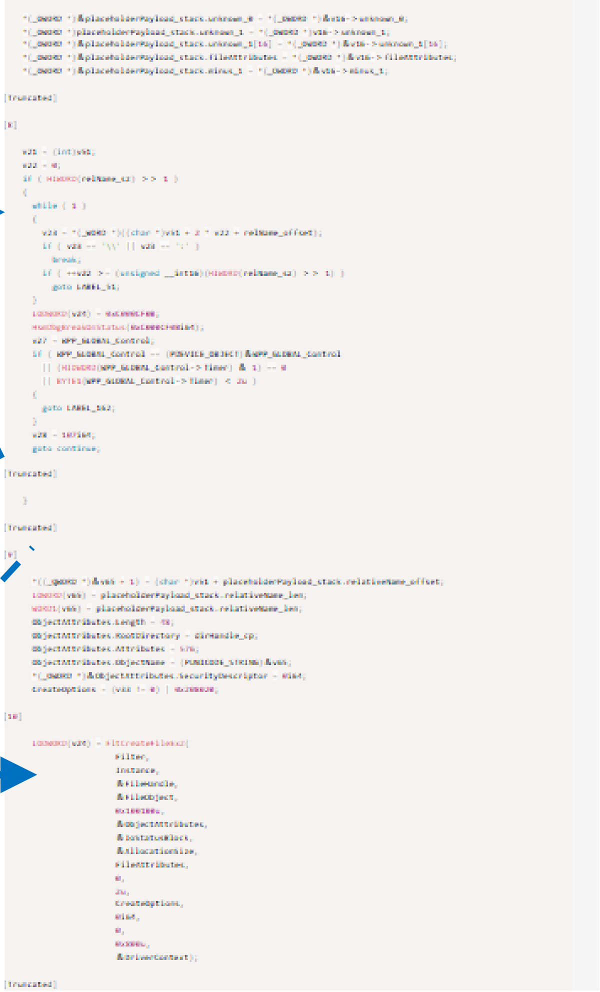
Mini Filter Driver: Complex…

- ❑ If first checks that the **specified filename** does not contain characters \ or :



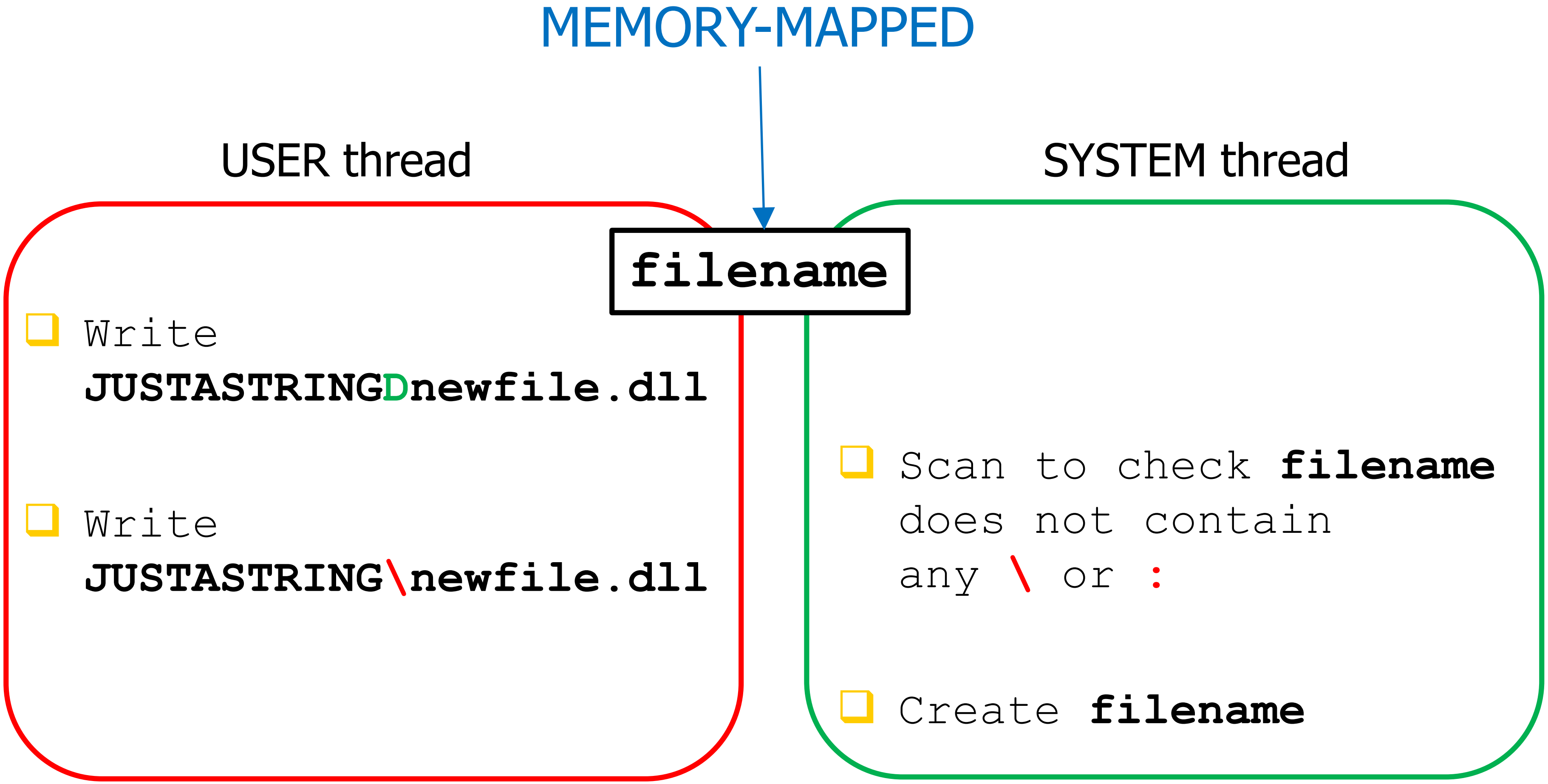
```
[Truncated]

    }

[Truncated]
```

- ❑ Creates a **placeholder file** of a **specified filename**



```
    *(_DWORD *)&placeholderPayload_stack.unknown_0 = *(_DWORD *)&v16->unknown_0;
    *(_QWORD *)placeholderPayload_stack.unknown_1 = *(_QWORD *)v16->unknown_1;
    *(_DWORD *)&placeholderPayload_stack.unknown_1[16] = *(_DWORD *)&v16->unknown_1[16];
    *(_DWORD *)&placeholderPayload_stack.fileAttributes = *(_DWORD *)&v16->fileAttributes;
    *(_DWORD *)&placeholderPayload_stack.minus_1 = *(_DWORD *)&v16->minus_1;

[Truncated]

[8]

    v21 = (int)v61;
    v22 = 0;
    if ( HIWORD(relName_s2) >> 1 )
    {
      while ( 1 )
      {
        v24 = *(_WORD *)((char *)v61 + 2 * v22 + relName_offset);
        if ( v24 == '\\' || v24 == ':' )
          break;
        if ( ++v22 >= (unsigned __int16)(HIWORD(relName_s2) >> 1) )
          goto LABEL_51;
      }
      LODWORD(v24) = 0xC000CF00;
      HsmDbgBreakOnStatus(0xC000CF00i64);
      v27 = WPP_GLOBAL_Control;
      if ( WPP_GLOBAL_Control == (PDEVICE_OBJECT)&WPP_GLOBAL_Control
        || (HIDWORD(WPP_GLOBAL_Control->Timer) & 1) == 0
        || BYTE1(WPP_GLOBAL_Control->Timer) < 2u )
      {
        goto LABEL_162;
      }
      v28 = 107i64;
      goto continue;

[Truncated]

    }

[Truncated]

[9]

      *((_QWORD *)&v65 + 1) = (char *)v61 + placeholderPayload_stack.relativeName_offset;
      LOWORD(v65) = placeholderPayload_stack.relativeName_len;
      WORD1(v65) = placeholderPayload_stack.relativeName_len;
      ObjectAttributes.Length = 48;
      ObjectAttributes.RootDirectory = dirHandle_cp;
      ObjectAttributes.Attributes = 576;
      ObjectAttributes.ObjectName = (PUNICODE_STRING)&v65;
      *(_OWORD *)&ObjectAttributes.SecurityDescriptor = 0i64;
      CreateOptions = (v13 != 0) | 0x200020;

[10]

      LODWORD(v24) = FltCreateFileEx2(
                       Filter,
                       Instance,
                       &FileHandle,
                       &FileObject,
                       0x100180u,
                       &ObjectAttributes,
                       &IoStatusBlock,
                       &AllocationSize,
                       FileAttributes,
                       0,
                       2u,
                       CreateOptions,
                       0i64,
                       0,
                       0x1000u,
                       &DriverContext);

[Truncated]
```

Oopsss…check skipped!



MEMORY-MAPPED

USER thread

SYSTEM thread

filename

- Write **JUSTASTRINGDnewfile.dll**
- Write **JUSTASTRING\newfile.dll**

- Scan to check **filename** does not contain any \ or :
- Create **filename**

Privilege Escalation!

- Write `INNOCENTDnewfile.dll`
- Write `INNOCENT\newfile.dll`

`filename`

- IF `INNOCENT` is a junction point to `SOMEDIR`
- THEN `newfile.dll` will be created in `SOMEDIR`

- Create `INNOCENT` as a junction to `C:\Windows\System32`
- Place `evilfile.dll` in that directory and have fun

Exploit: Basic Idea

- Create many threads
  - Many: Create placeholder file of specified filename
  - Many: Swap `D` and `\` in filename **continuously** with a **small delay**
  - One: Monitor that desired filename has been created

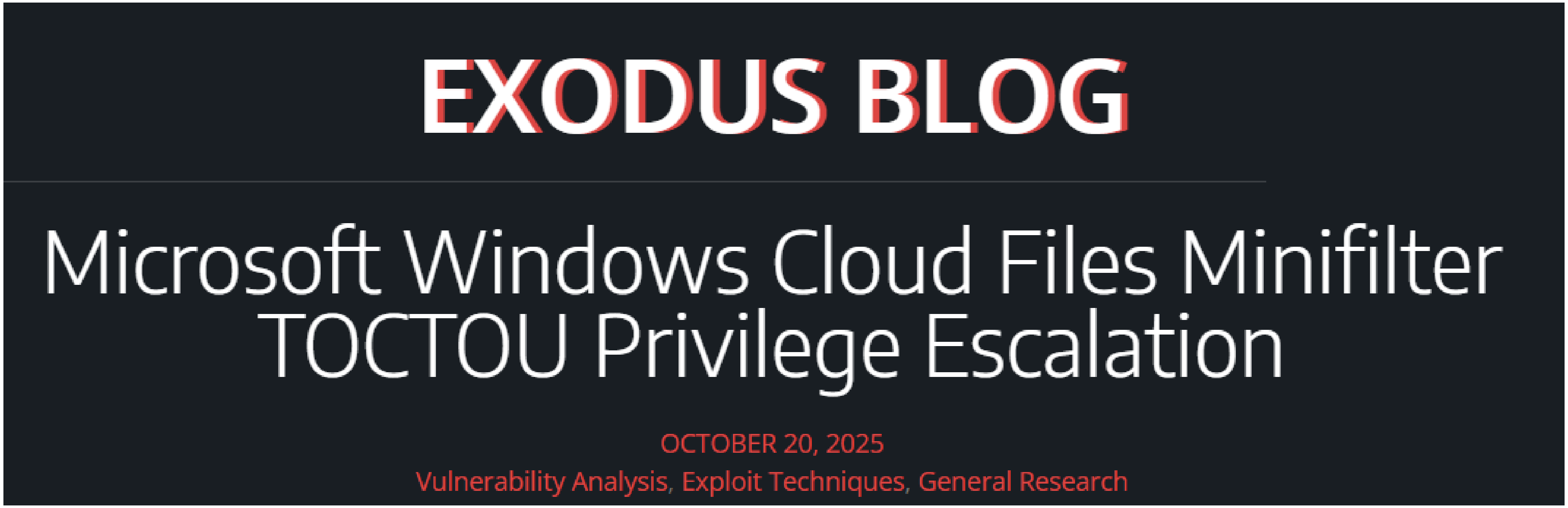
- The monitor thread has a handle to the newly created file and will fill it with malicious content

Remarks

- **Local** injection
- Could have been detected with **Testing**?
- Bad practice:
  - File creation function invoked in Kernel mode without any programmatic access check

    Principle of Least privilege strikes again

Strange name: TOCTOU



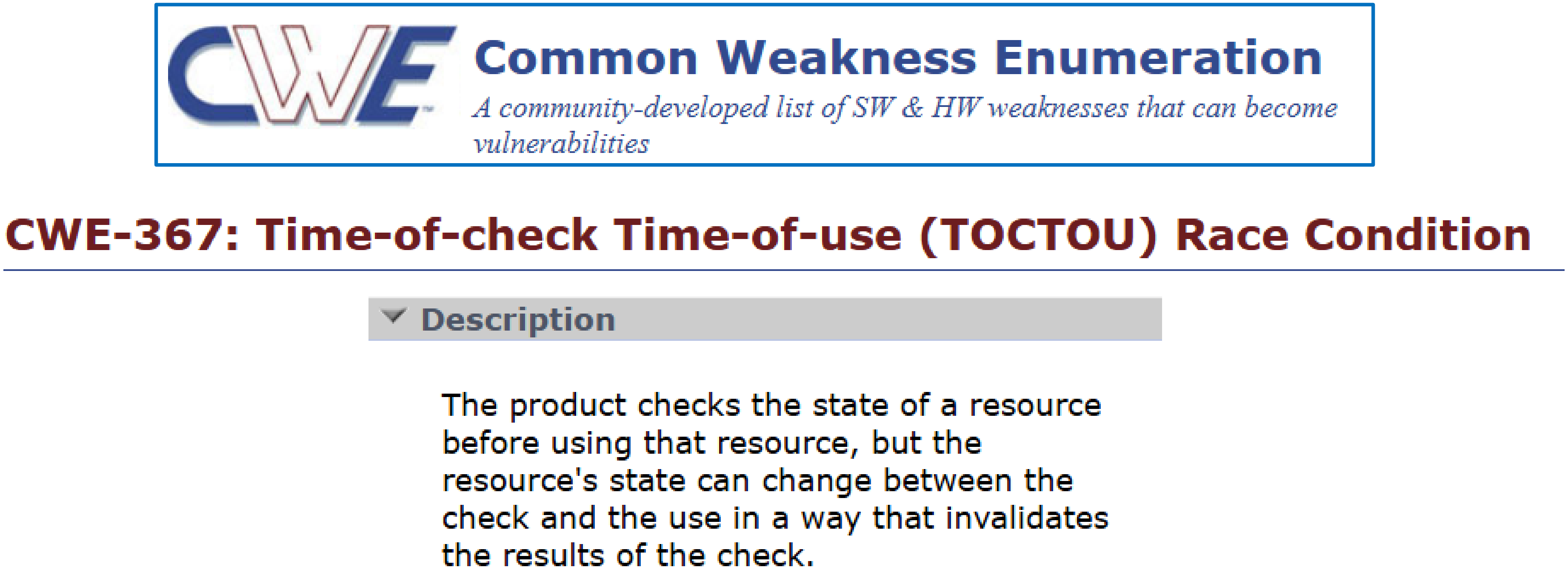
EXODUS BLOG

Microsoft Windows Cloud Files Minifilter TOCTOU Privilege Escalation

OCTOBER 20, 2025

Vulnerability Analysis, Exploit Techniques, General Research

CWE Example



CWE

Common Weakness Enumeration

A community-developed list of SW & HW weaknesses that can become vulnerabilities

CWE-367: Time-of-check Time-of-use (TOCTOU) Race Condition

▼ **Description**

The product checks the state of a resource before using that resource, but the resource's state can change between the check and the use in a way that invalidates the results of the check.

Check and **Access** should be an **atomic** operation

(have a look at the Mitigations)