Shameful secrets of proprietary protocols

OWASP

The Open Web Application Security Project

Sławomir Jasek

Jakub Kałużny

Who are we

OWASP
The Open Web Application Security Project

Sławomir Jasek

Jakub Kałużny

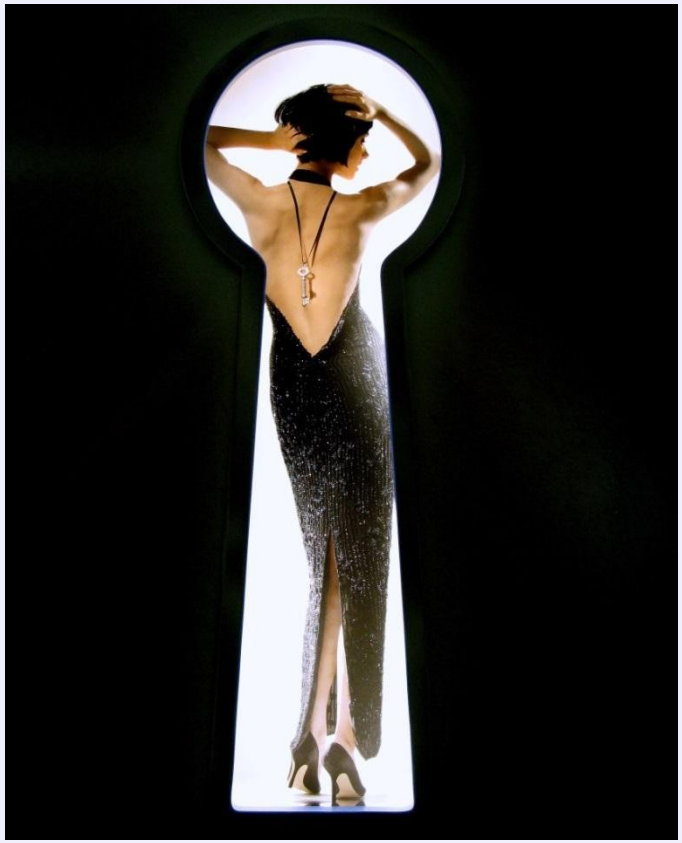
- Pentesters @ SecuRing
- Security assessments of applications, networks, systems...

OWASP
The Open Web Application Security Project

- Case studies – proprietary protocols
  - Home automation
  - Pull printing #1
  - Remote desktop
  - Pull printing #2
  - Trading
- Cheatsheet for architects & developers
- How to hack it

Proprietary network protocols

- A pentester will encounter one
- Don’t have the protocol specs nor tools to attack it
- How to hack it?
  - decompile the client?
  - search for some tools?
  - watch the raw packets?
- Let’s try!



https://www.flickr.com/photos/canonsnapper/2566562866

OWASP
The Open Web Application Security Project

- *„Plug the device, configure your router for port forwarding (and dynamic dns if necessary), set password.”*
- Proprietary TCP protocol, direct connection from Internet to device, password protected access

http://www.flickr.com/photos/99832244@N07/9436065073/

Protocol – a few packets

OWASP
The Open Web Application Security Project

CLIENT

```
ab 55 41 00 15 39 64 64  34 65 34 36 31 32 36    .UA..9dd 4e46126
02 01 00 00 a9 39 64 64  34 65 34 36 31 32 36    .....9dd 4e46126
aa 55 41 00 14 39 64 64  34 65 34 36 31 32 36    .UA..9dd 4e46126
```

```
aa 53 41 02 01 01 f0 f1  f1 f1 f1 00 be f1 f1 00  .SA..... ........
c4 00 e1 f1 f1 f1 f1 f1  f1 f1 f1 f1 f1 f1 f1 f1  ........ ........
f1 f1 f1 00 64 00 00 00  01 00 f0 f0 0a f1 00 02  ....d... ........
0f0fe7
```

```
ab 55 41 00 15 39 64 64  34 65 34 36 31 32 36    .UA..9dd 4e46126
0c 02 00 00 a4 39 64 64  34 65 34 36 31 32 36    .....9dd 4e46126
aa 55 41 00 14 39 64 64  34 65 34 36 31 32 36    .UA..9dd 4e46126
```

SERVER

And what if we change the password?

OWASP
The Open Web Application Security Project

Password 1:

```
00000000  aa 55 41 00 14 39 32 65  62 35 66 66 65 65 36    .UA..92e b5ffee6
    00000000  aa 53 41 02 01 01 f0 f1  f1 f1 f1 00 a1 f1 f1 00 .SA..... ........
    00000010  92 00 dd f1 f1 f1 f1 f1  f1 f1 f1 f1 f1 f1 f1 f1 ........ ........
    00000020  f1 f1 f1 00 78 00 02 00  00 00 f0 f0 07 a3 00 02 ....x... ........
    00000030  0f 0f d2                                       ...
```

Password 2

```
00000000  aa 55 41 00 14 34 61 38  61 30 38 66 30 39 64    .UA..4a8 a08f09d
    00000000  aa 53 41 02 01 01 f0 f1  f1 f1 f1 00 a1 f1 f1 00 .SA..... ........
    00000010  93 00 dd f1 f1 f1 f1 f1  f1 f1 f1 f1 f1 f1 f1 f1 ........ ........
    00000020  f1 f1 f1 00 78 00 02 00  00 00 f0 f0 07 a3 00 02 ....x... ........
    00000030  0f 0f d3                                       ...
```

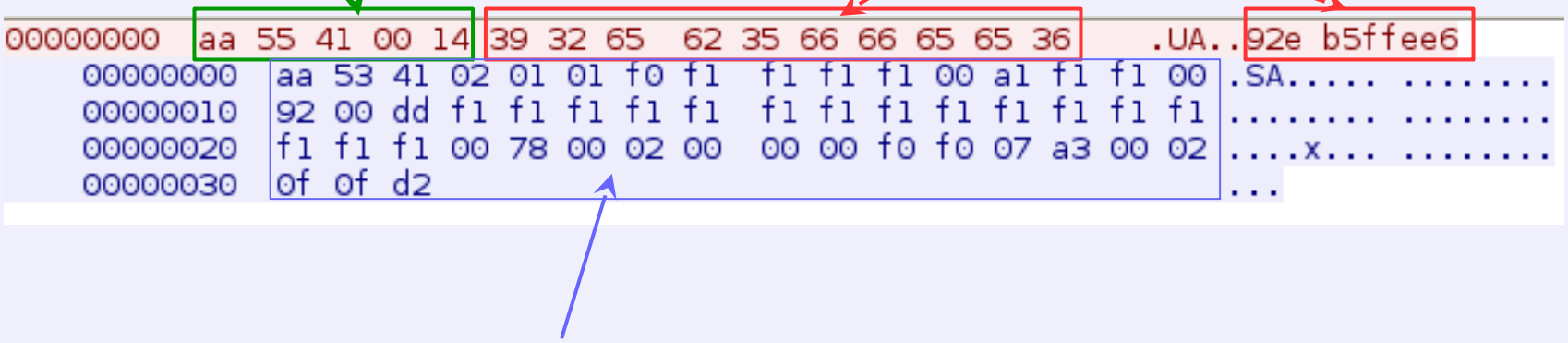
Password 3

```
00000000  aa 55 41 00 14 30 63 63  31 37 35 62 39 63 30    .UA..0cc 175b9c0
    00000000  aa 53 41 02 01 01 f0 f1  f1 f1 f1 00 a1 f1 f1 00 .SA..... ........
    00000010  92 00 dd f1 f1 f1 f1 f1  f1 f1 f1 f1 f1 f1 f1 f1 ........ ........
    00000020  f1 f1 f1 00 78 00 02 00  00 00 f0 f0 07 a3 00 02 ....x... ........
    00000030  0f 0f d2                                       ...
```

OWASP
The Open Web Application Security Project

Internal command (5 bytes)

MD5(password) – first 10 bytes



Status returned by the appliance (sensors, settings, etc).

OWASP
The Open Web Application Security Project

- Sniffing
- MITM
- Connect directly to the appliance - sniffed hash is enough
- Recommendation: SSL!

OWASP
The Open Web Application Security Project

- Vendor: OK, we have added SSL support!

```
sslcontext = SSLContext.getInstance("TLS");
atrustmanager = new TrustManager[1];
atrustmanager[0] = new EasyX509TrustManager(null);
sslcontext.init(null, atrustmanager, null);
```

- Empty TrustManager – accepts all certificates

Side effect

OWASP
The Open Web Application Security Project

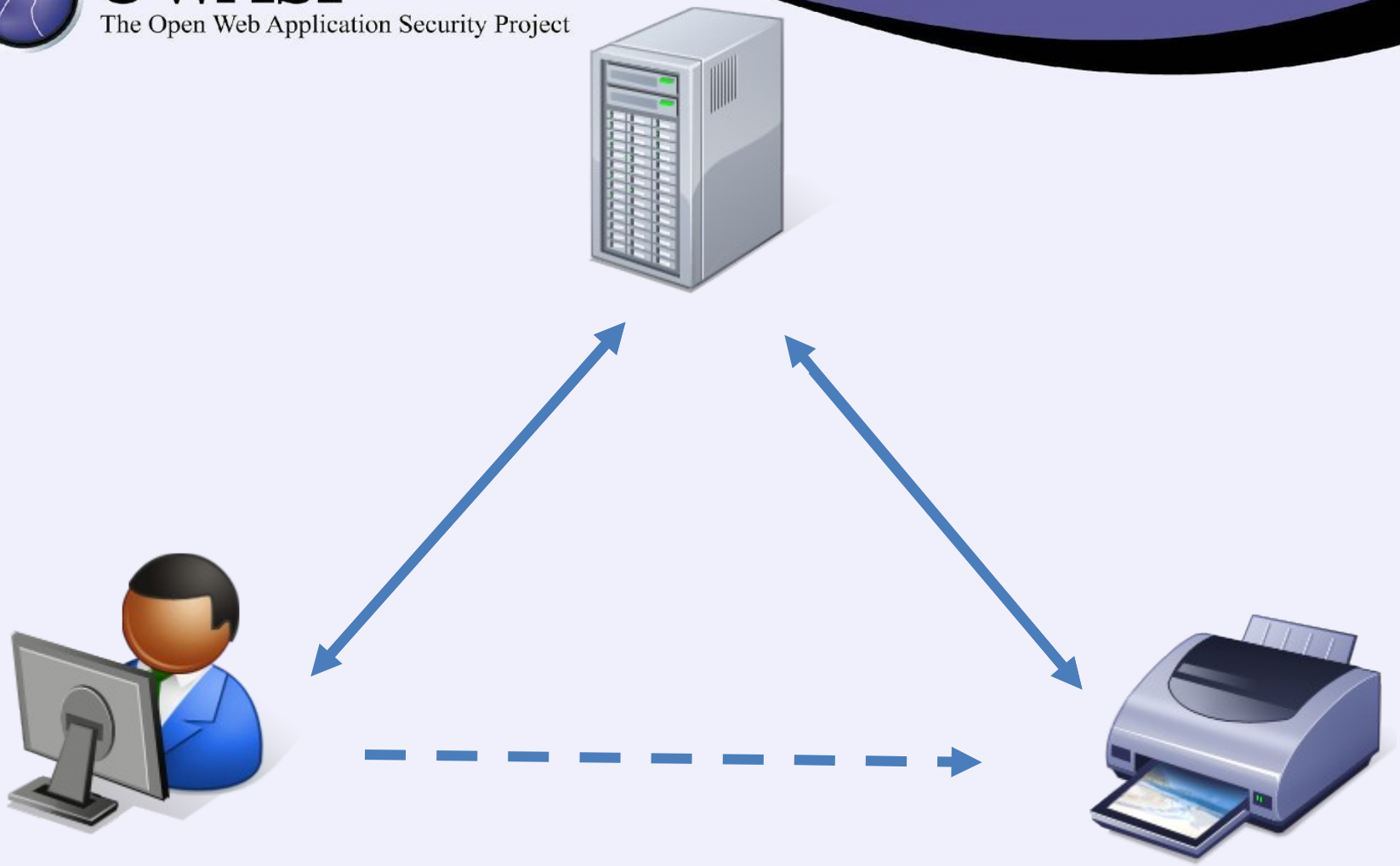
How to build your own appliance:

```
socat tcp4-listen:1234,fork,readbytes=5 /dev/ttyUSB0,vmin=51
```

And for the new version with SSL support:

```
socat openssl-listen:1234,key=s.key,cert=s.crt,verify=0,fork,readbytes=5 /dev/ttyUSB0,vmin=51
```

OWASP
The Open Web Application Security Project



OWASP
The Open Web Application Security Project

- Widely used
- Confidential data
- Getting popular

OWASP
The Open Web Application Security Project

sniffing

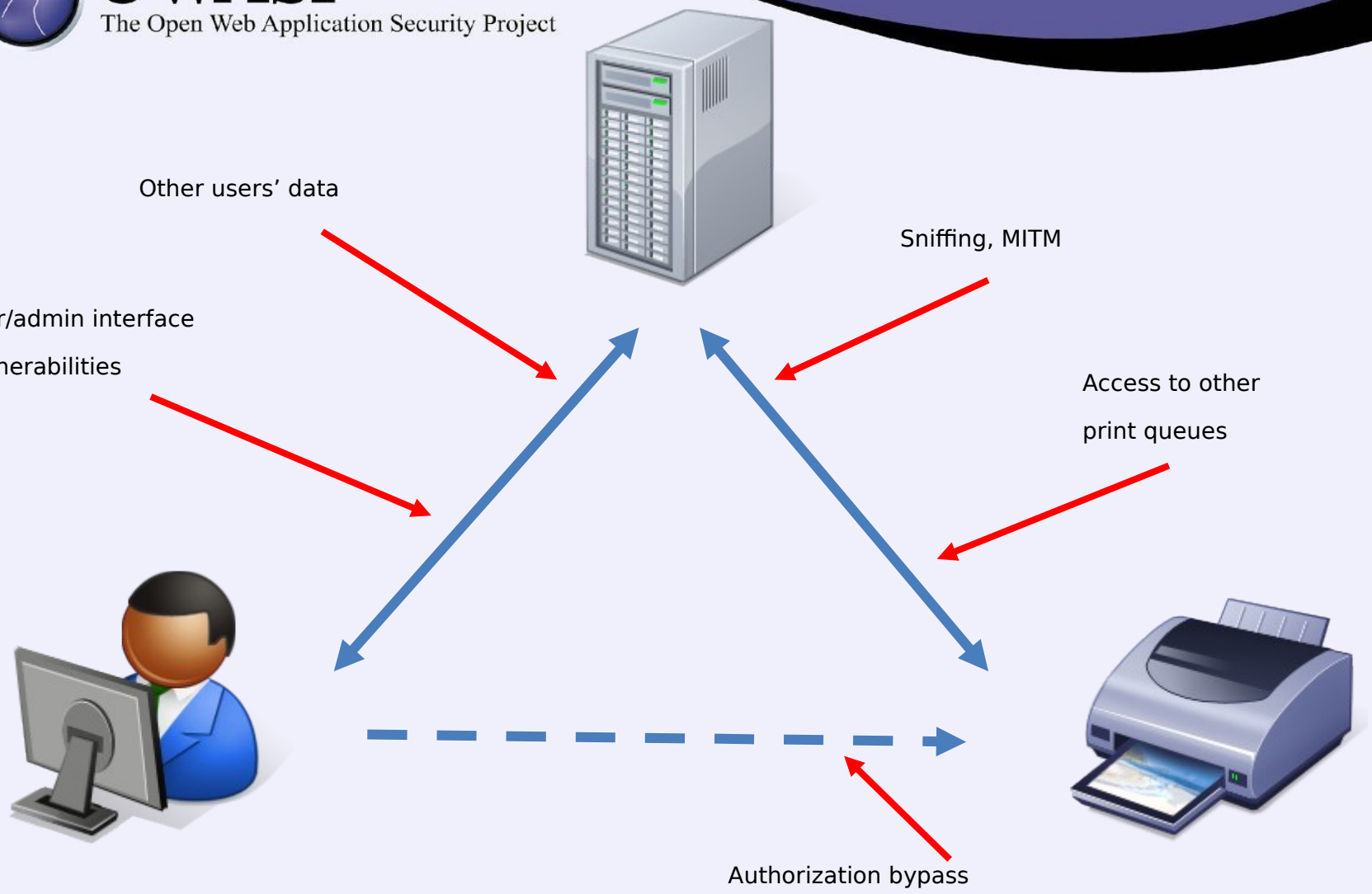
print queues

accountability

users’ data

Attack vectors

OWASP
The Open Web Application Security Project



OWASP
The Open Web Application Security Project

"**Secure print release** (…) can integrate card-swipe user authentication at devices (…) ensuring jobs are **only** printed when the collecting user is present."

Pull Printing #1 – binary protocol

Pull Printing #1 – closer look

OWASP
The Open Web Application Security Project

SERVER

PRINTER

Release print queue for user “guest-xyz”

Release my print queue

Charge user “guest-xyz” for copying 100 pages

I just used 100 pages

```
65 64 54 53 00 S..restrictedTS.
70 79 54 53 00 canColorCopyTS.
6c 69 65 72 44 .costMultiplierD
63 61 6e 43 68 ?....... S..canCh
72 6f 6d 4c 69 argeShar edFromLi
72 69 6e 74 4a stFS..he ldPrintJ
00 53 00 19 68 obCountI ....S..h
63 63 6f 75 6e asAdvanc edAccoun
              tOptions Fzz
59 63 65 41 c..m.#ex tDeviceA
53 65 54 72 PI.begin DeviceTr
5d 4e 39 42 ansactio nS..mN9B
75 65 73 74 KS..1004 S..guest
              -xyzS..z
75 73 53 00 ?..MS..s tatusS..
76 61 69 6c 61 SUCCESSS ..availa
ff d7 0a 3d 70 bleCredi tD?...=p
65 44 3f ff d7 ..S..bal anceD?..
74 75 73 4d 65 .=p..S.. statusMe
74 72 61 6e 73 ssageS.. S..trans
5a 70 44 35 30 actionId S..ZpD50
              zz
59 63 65 41 c..m.%ex tDeviceA
43 6f 70 69 PI.calcu lateCopi
00 05 6d 4e erPageCo stsS..mN
09 67 75 65 9BKS..10 04S..gue
34 46 46 7a st-xyzVV S..A4FFz
```

User permissions

beginDeviceTransaction

(...) guest-xyz

OWASP
The Open Web Application Security Project

sniffing

print queues

accountability

users’ data

OWASP
The Open Web Application Security Project

- Gave access to KB and support service
- And all versions of software
- Responded in few hours and patched in few days
- Was happy to be pentested

- X-win „on steroids” (encryption, compression, access control...)
- Mainframe access for critical business operations
- *„More than 100,000 users around the world”*
- *„Prevents unauthorized eavesdropping*

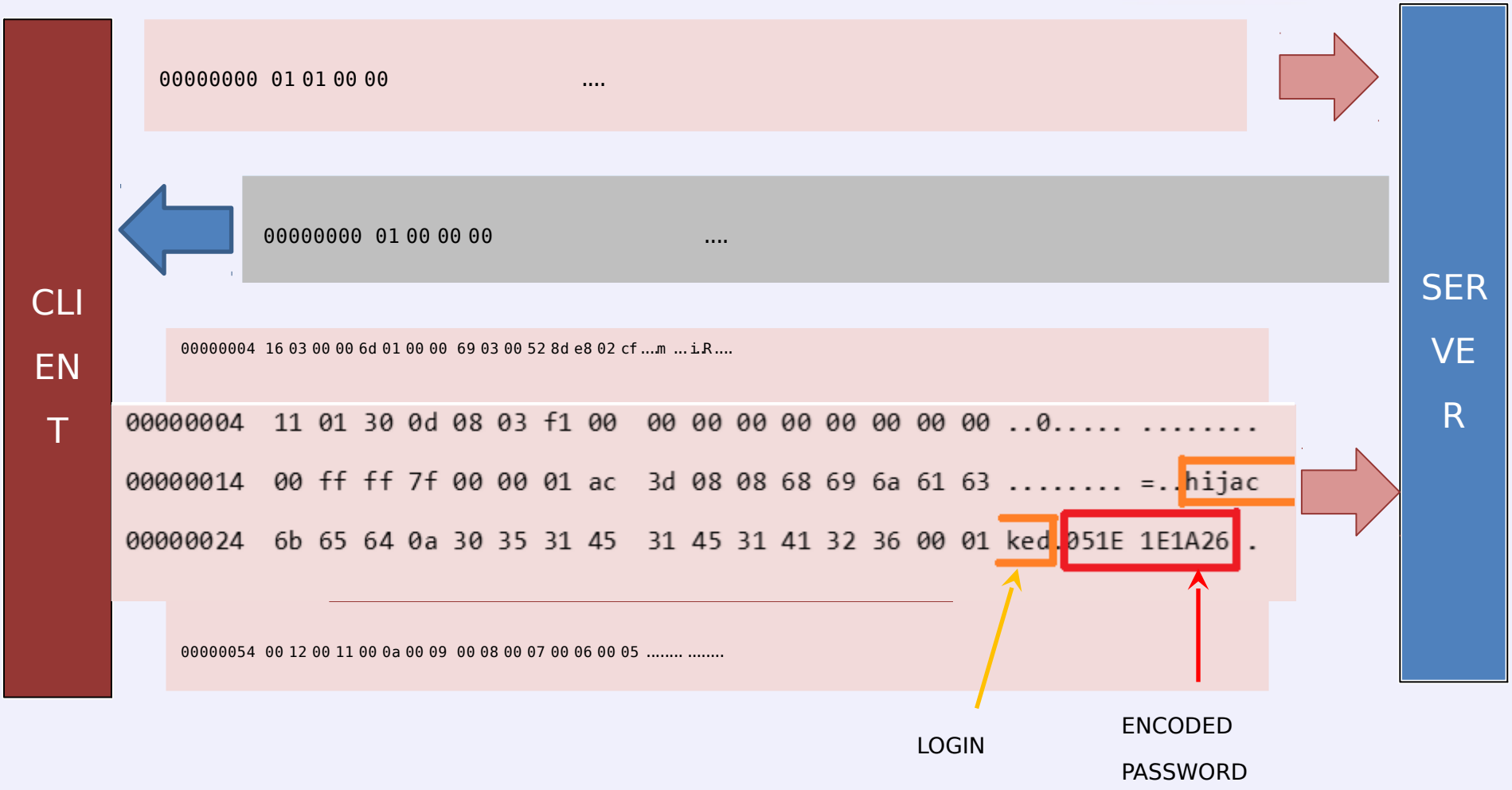
FIPS 140-2 Validated

End-to-end data encryption”

http://en.wikipedia.org/wiki/X_Window_System_protocols_and_architecture

Remote desktop protocol

OWASP
The Open Web Application Security Project



54657374696e6750617373776f72643132333454657374696e6750617373776f7264

TestingPassword1234TestingPassword

XOR

1c101e190000003208011757 2c1d095c475d5d3704071d060014702d1a1e1e1b1700

=

48756d6d696e6762697264 20436f6d6d756e69636174696f6e73204c696d69746564

[redacted] Communications Limited

OWASP
The Open Web Application Security Project

default configuration

CLI
EN
T

CLIENTHELLO!

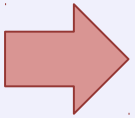
cipher suites:

SSL_DHE_RSA_WITH_AES_256_CBC_SHA

SSL_DHE_DSS_WITH_AES_256_CBC_SHA

SSL_RSA_WITH_AES_256_CBC_SHA

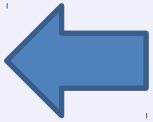
(...)



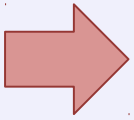
SERVERHELLO!

I don’t have any certificate!

cipherSuite: SSL_DH_anon_WITH_AES_256_CBC_SHA

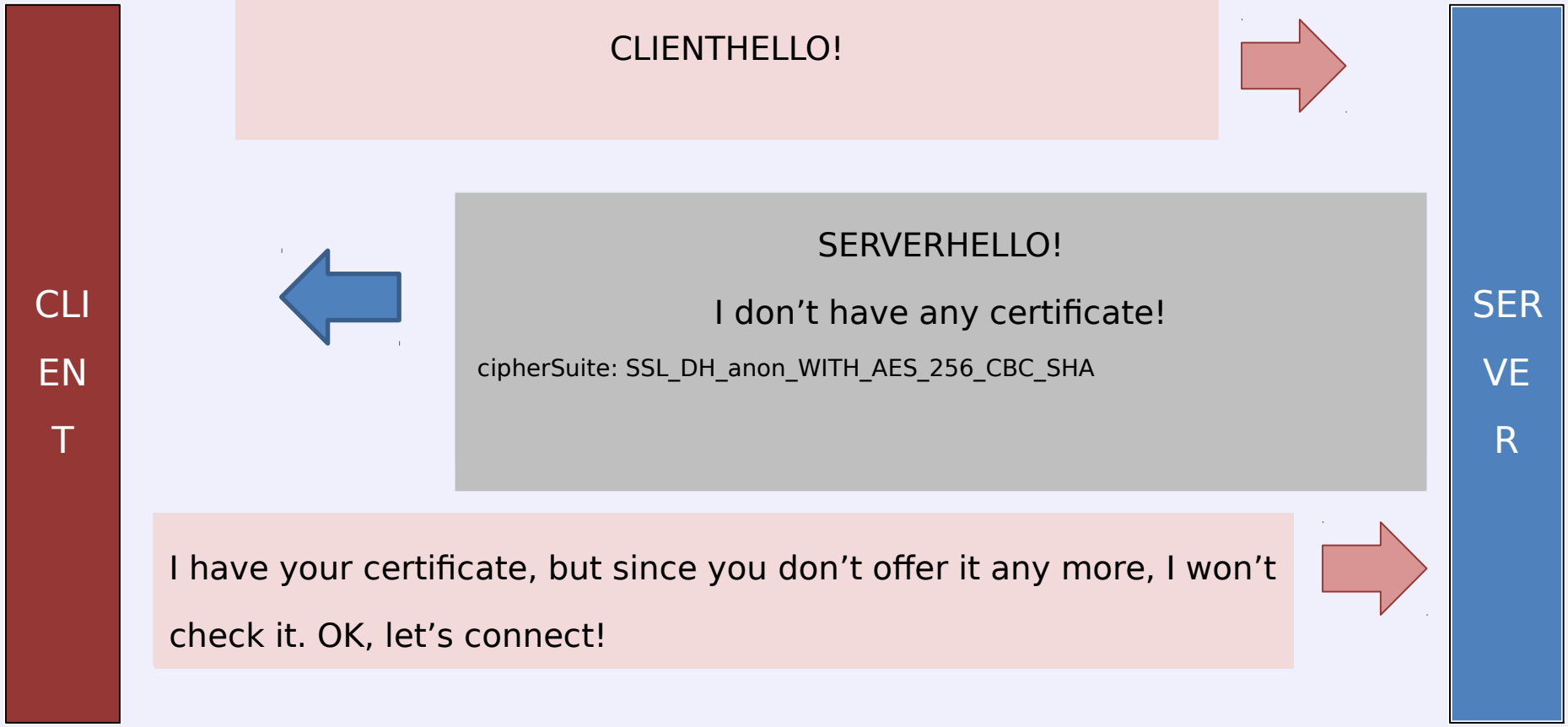


OK, no problem! You have to be the right server if you say so, don’t you?



SER
VE
R

certificates configured



OWASP
The Open Web Application Security Project

- *„We don’t know PGP, use zip with our CEO’s name as password”*
- Do not plan to solve the issues (?)
- > /dev/null 2>&1
- Full disclosure!
- ... and a few weeks later the mysterious shut down of our beloved ;)

OWASP
The Open Web Application Security Project

"is a modern printing solution that **safeguards document confidentiality** and unauthorized access to print, scan, copy and e-mail functions. Its user-authentication **provides air-tight security** on your shared MFPs that function as personal printers."

OWASP
The Open Web Application Security Project

„Documents are delivered **only** into the right hands”
„Information is kept **confidential**. **No risk** of being left unattended at the printer”
„Document collection is **safe anytime and anywhere** — no “print and sprint”.”
„Integration with other enterprise applications and workflows **is kept secure** through single sign-on”

OWASP
The Open Web Application Security Project

First look on communication:

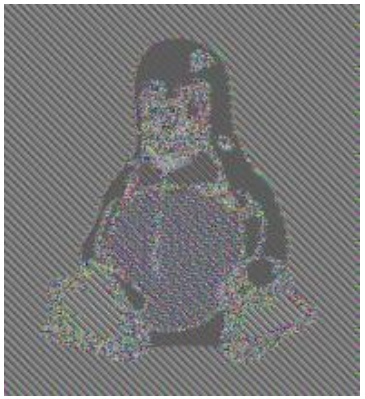
- TCP, 2 ports
- No cleartext, no SSL
- Seems to follow some scheme…

Ex1: Deeper sight on traffic

OWASP
The Open Web Application Security Project

```
..6..........l
..6..........l
..6..........l
..6..........l
..6..........l
..6..........l
..6..........l
..6..........l
..6..............:.c..<..~7%../}W...}....A1.......dh{hR9....
.d.._.]..y.{@...l
..6..........l
..6..........l
..6..........l
..6..........l
..6..........l
..6..........l
..6..........l
..6..........a.`OVT...9/.;.s.
....q...~+..@.9.'.o.]...b..o..
..6..........l
..6..........l
..6..........l
..6..........l
..6..........l
..6..........l
..6..........l
..6.............f...b...>..}U..4-{.K./o%#..;..l
..6..........l
..6..........l
..6.............l).......y...^Ql
..6..........r\...ix......0.n...;4.o.....p\..gSd............qg........
+).q8...z..F..b1.Fr...l
..6..........l
..6..........l
..6..........l
..6..........l
```

https://en.wikipedia.org/wiki/ECB_mode



OWASP
The Open Web Application Security Project

- Hardcoded RSA certificate in printer embedded software
- No trust store
- AES-128 ECB used for traffic encryption
- Same protocol in admin interface

OWASP
The Open Web Application Security Project

sniffing

print queues

accountability

users’ data

OWASP
The Open Web Application Security Project

"(…) system has been deployed at many high security customers and **has passed internal audits**."

Trading protocol

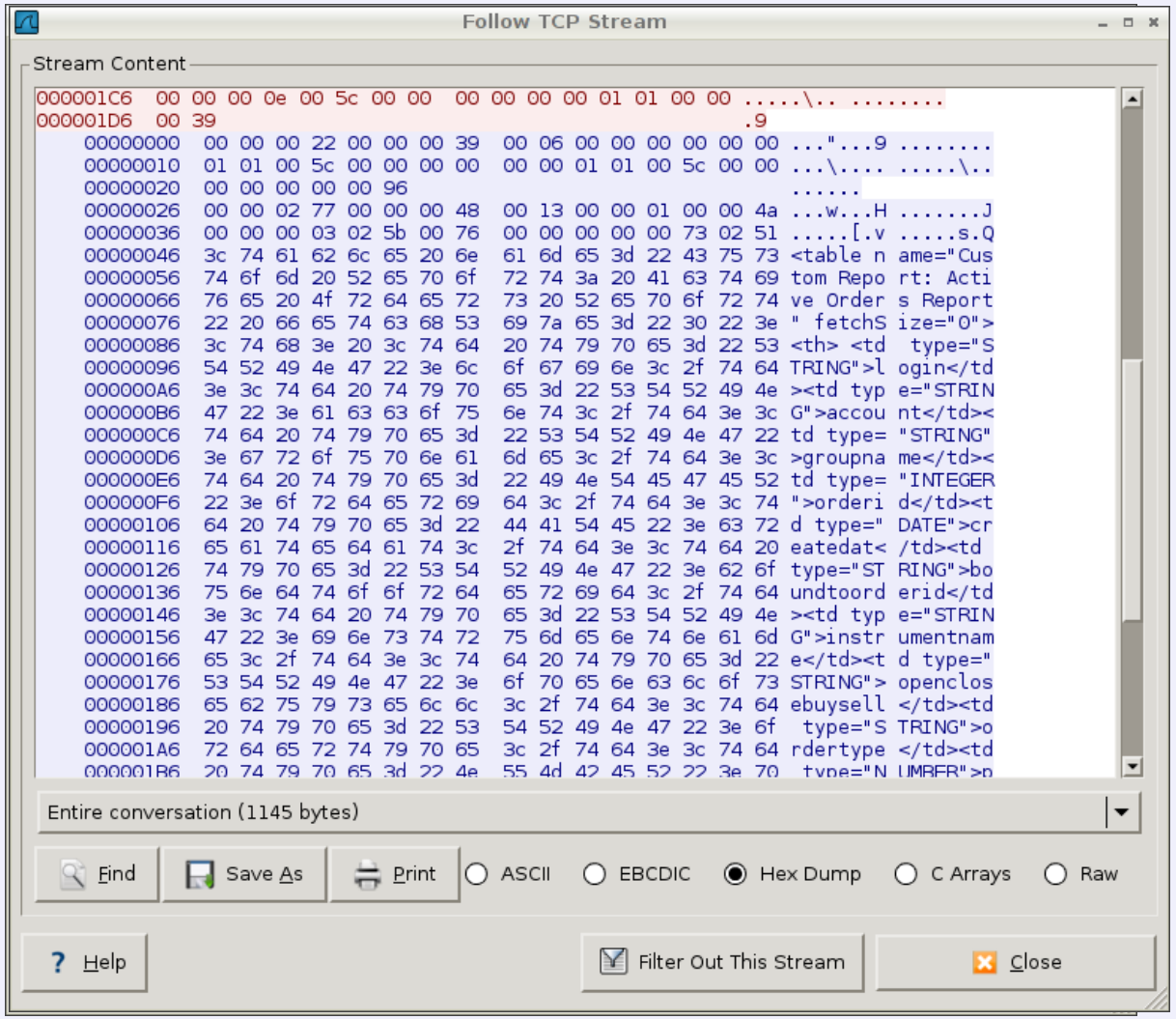
OWASP
The Open Web Application Security Project

- An online application for instant financial operations
- A proprietary, binary protocol, designed in order to minimise delays
- TCP in SSL tunnel

https://www.flickr.com/photos/tradingrichmom/5571144428/

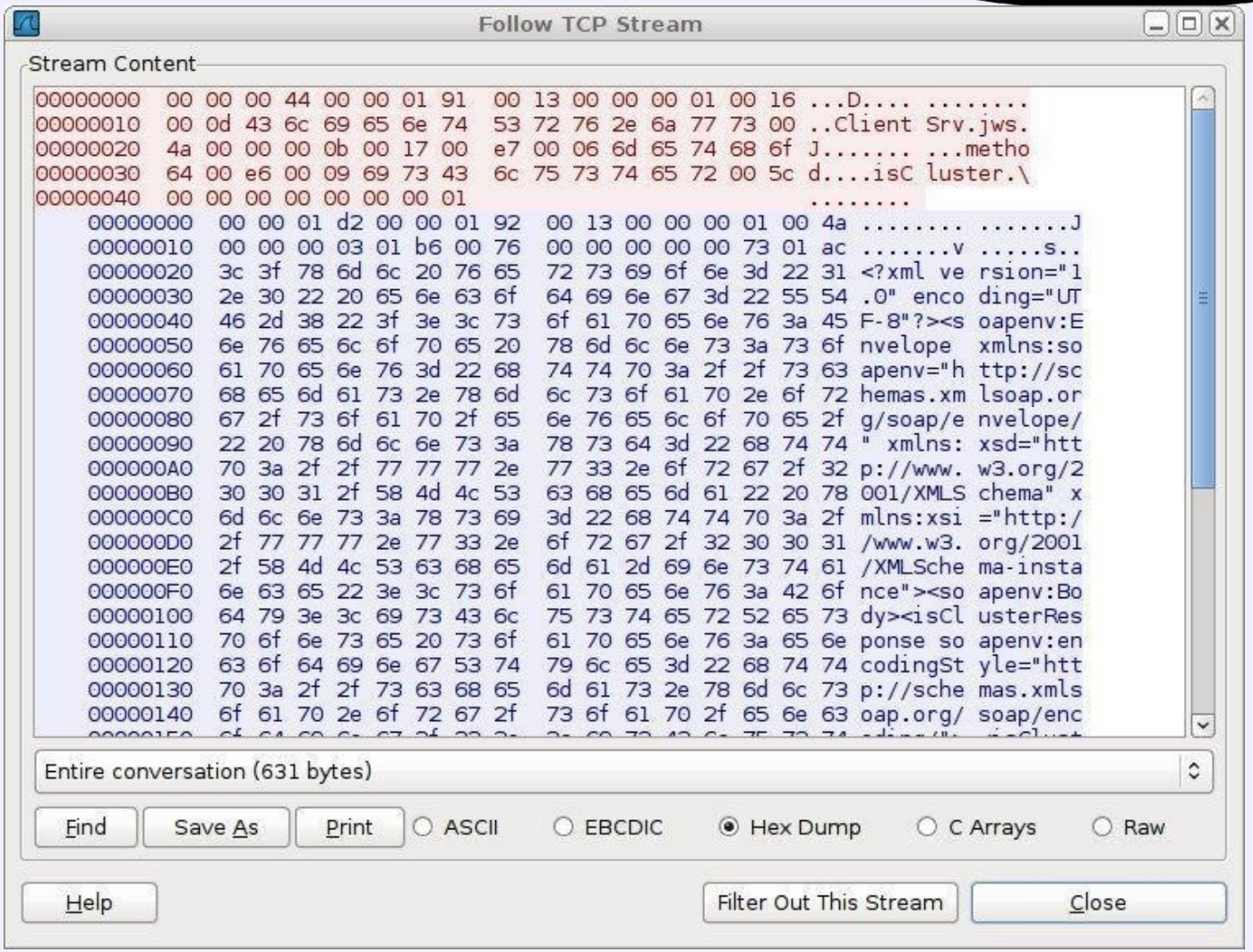
Trading protocol – a few packets

OWASP
The Open Web Application Security Project



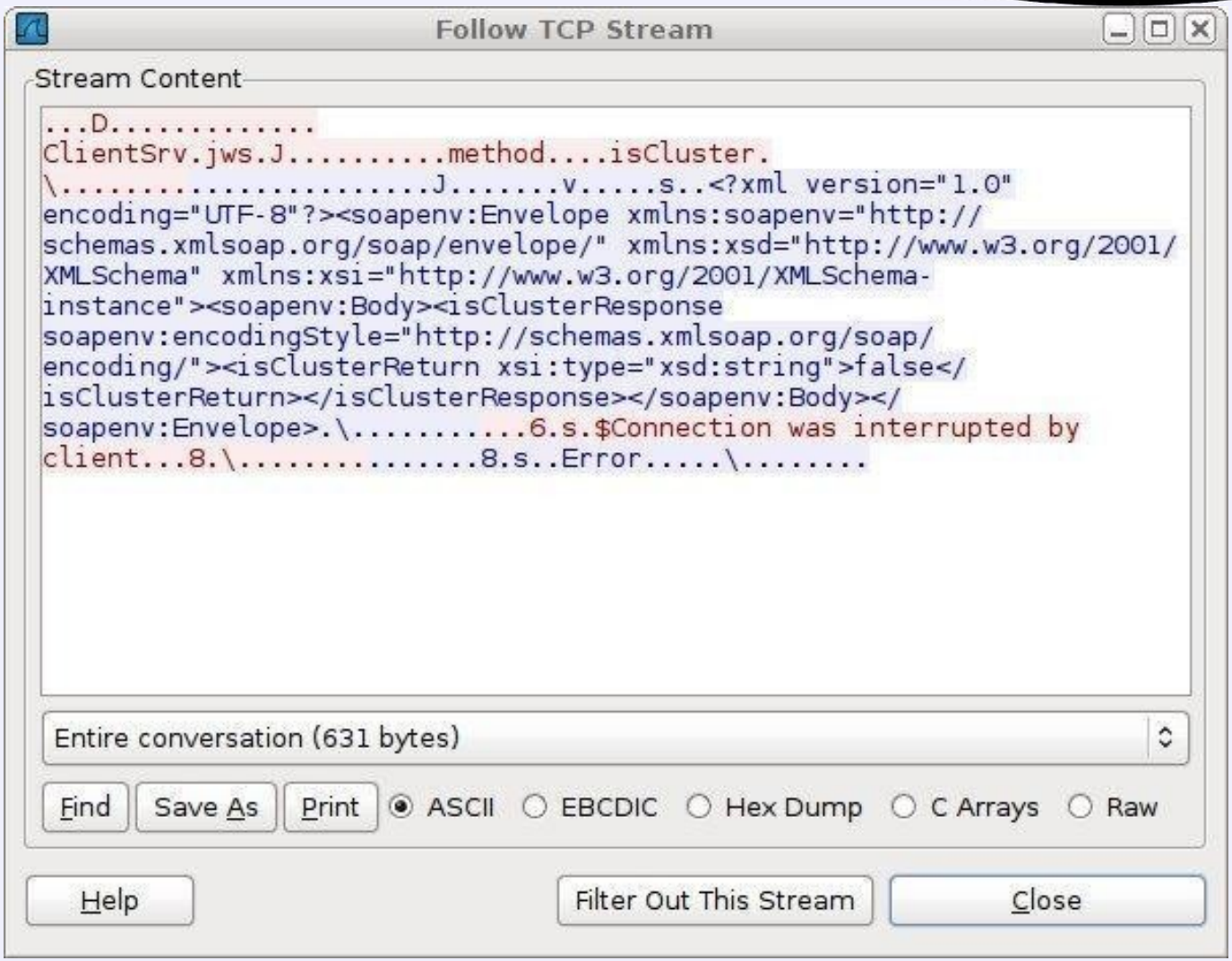
That's interesting!

OWASP
The Open Web Application Security Project



That's interesting!

OWASP
The Open Web Application Security Project



And what if we...

OWASP
The Open Web Application Security Project

Follow TCP Stream

Stream Content

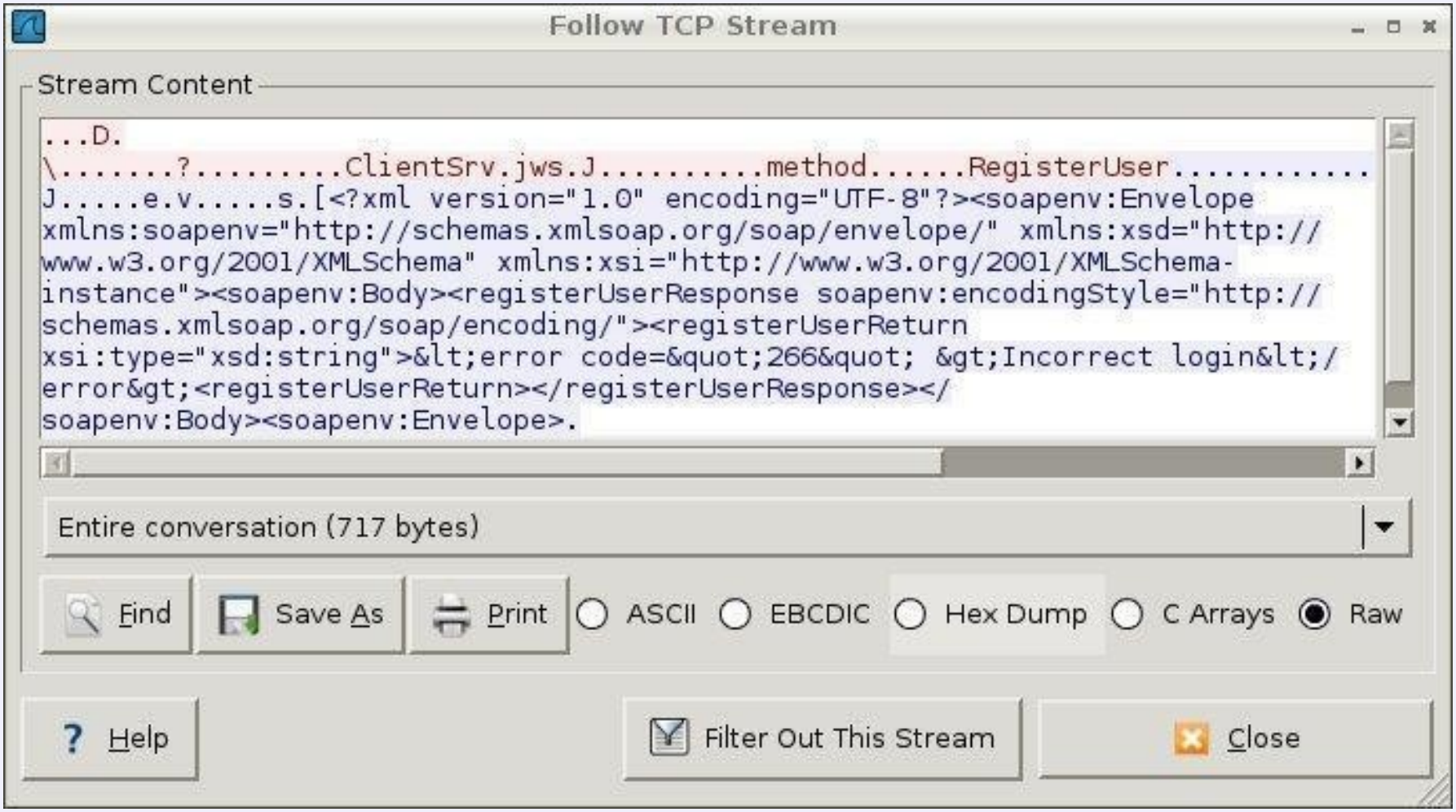
```
...W.\.......?.......
AdminService..J..........method....AdminService.J..........aaa....bbb...@..........J.....
$.v.....s..<?xml version="1.0" encoding="UTF-8"?><soapenv:Envelope xmlns:soapenv="http://
schemas.xmlsoap.org/soap/envelope/" xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:xsi="http://
www.w3.org/2001/XMLSchema-
instance"><soapenv:Body><soapenv:Fault><faultcode>soapenv:Server.userException</
faultcode><faultstring>java.lang.Exception: Unable to process the message -was it a valid WSDD
descriptor?</faultstring><detail><ns1:stackTrace xmlns:ns1="http://xml.apache.org/
axis/">java.lang.Exception: Unable to process the message -was it a valid WSDD descriptor?
.at org.apache.axis.utils.Admin.process(Admin.java:163)
.at org.apache.axis.utils.Admin.AdminService(Admin.java:65)
.at sun.reflect.NativeMethodAccessorImpl.invoke0(Native Method)
.at sun.reflect.NativeMethodAccessorImpl.invoke(NativeMethodAccessorImpl.java:39)
.at sun.reflect.DelegatingMethodAccessorImpl.invoke(DelegatingMethodAccessorImpl.java:25)
.at java.lang.reflect.Method.invoke(Method.java:597)
.at org.apache.axis.providers.java.MsgProvider.processMessage(MsgProvider.java:126)
```

Entire conversation (2975 bytes)

Find | Save As | Print | (•) ASCII | ( ) EBCDIC | ( ) Hex Dump | ( ) C Arrays | ( ) Raw

Help | Filter Out This Stream | Close

OWASP
The Open Web Application Security Project



OWASP
The Open Web Application Security Project

```
<soapenv:Body>
        <registerUserResponse soapenv:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/">
        <registerUserReturn xsi:type="xsd:string">
        &lt;error code=&quot;266&quot; &gt;Incorrect login&lt;/error&gt;
        </registerUserReturn>
        </registerUserResponse>
</soapenv:Body>
```

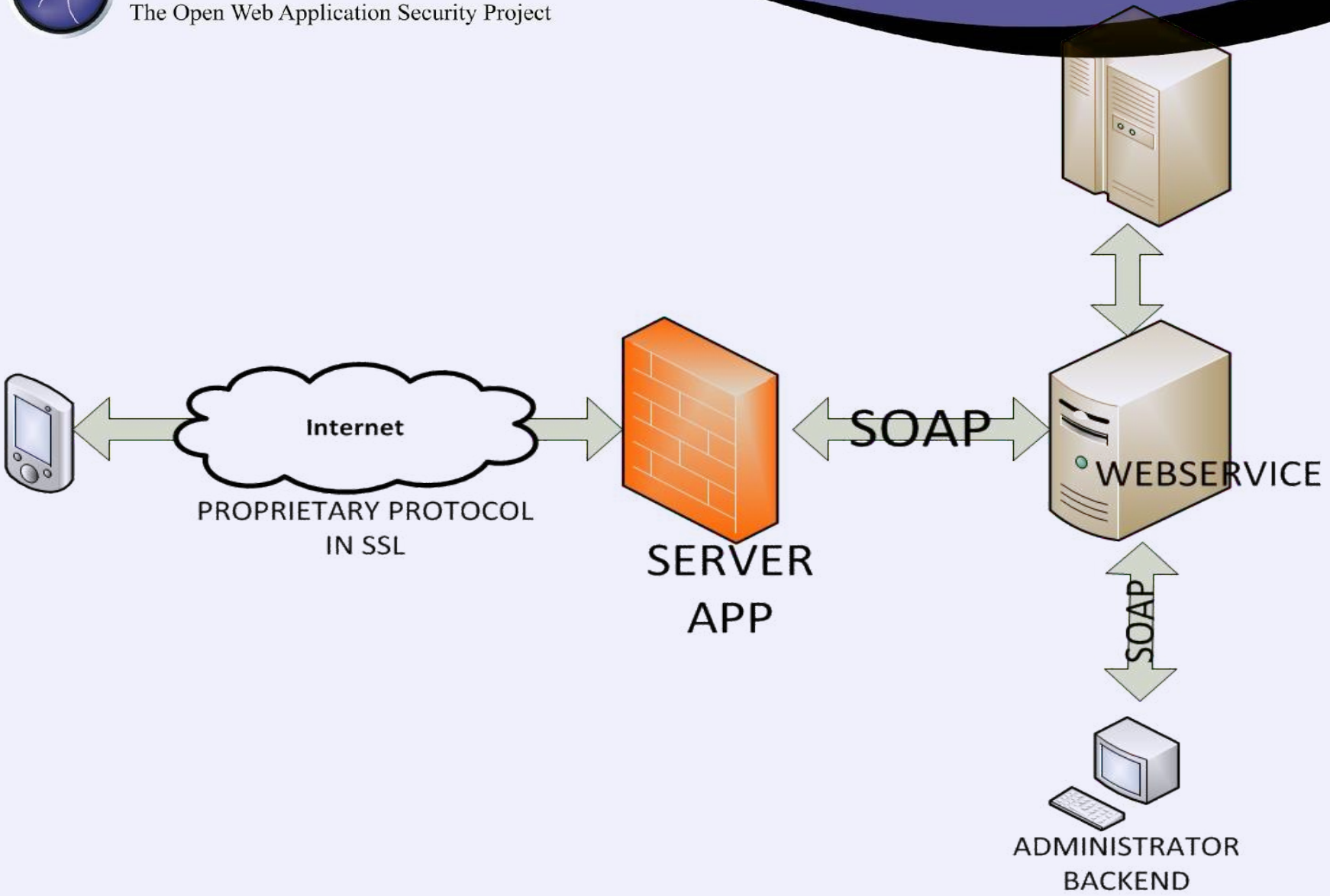
- Incorrect password
- Incorrect first name
- Group with name null doesn't exist
- Group with name admin doesn't exist
- Group with name Administrator doesn't exist
- And how about „root”?

OWASP
The Open Web Application Security Project

```
< soapenv:Body>
 < registerUserResponse soapenv:encodingStyle= "http://schemas.xmlsoap.org/soap/encoding/">
< registerUserReturn xsi:type= "xsd:string">
User was registered sucessfully with id= 5392745
< /registerUserReturn>
< /registerUserResponse>
< /soapenv:Body>
```

So now we can manage all the other accounts and spend their money!

OWASP
The Open Web Application Security Project



OWASP
The Open Web Application Security Project

While deploying a proprietary solution:

- Get it pentested
- Verify vendor claims
- Ask the vendor for secure development lifecycle, procedures of addressing vulnerabilities, previous bugs

OWASP
The Open Web Application Security Project

- Protocol is NOT secure by its secrecy
- Proper encryption. Use known standards, implement them with care.
- Input validation, access control, many layers of security, least privilege principle...
- Beware backwards compatibility

How to hack protocols?

OWASP
The Open Web Application Security Project

Decompile client?

Inject code?

Search for the specs?

Use some tools?

Watch the packets?

https://www.flickr.com/photos/nkphillips/2865781749

OWASP
The Open Web Application Security Project

- There may be unofficial client, or e.g. wireshark plugin
- Ask for the docs ☺
- Search for them
  - Yes, we have found internal protocol specification by google hacking!

OWASP
The Open Web Application Security Project

- Sometimes easy – e.g. not obfuscated Android application:

```
 byte abyte3[] = pass.getBytes();
 byte abyte4[] = MessageDigest.getInstance("MD5").digest(abyte3);
String s1 = "";
for(int j= 0;j< 10;j++)
s1 = (new  StringBuilder()).append(s1).append(toHexString(abyte4).charAt(j)).toString();
System .arraycopy(s1.getBytes(),0,abyte1,5,10);
```

- Sometimes really hard & time consuming.
- May be fun, but often leads astray

OWASP
The Open Web Application Security Project

- Various tools to analyze proprietary protocols
  - time consuming, usually do not work
- Raw, just try to spot some scheme
  - of course with a little help of your friends: wireshark, tcpdump, ssldump etc.
- Your favourite scripting language

OWASP

The Open Web Application Security Project

Q&A?

slawomir.jasek@securing.pl

jakub.kaluzny@securing.pl

www.securing.pl