Where do security bugs come from?

MIT 6.858 (Computer Systems Security), September 23, 2015

Paul Youn

- Director of Security, Airbnb
- MIT 18/6-3 ('03), M.Eng ('04)

airbnb

Agenda

- Always think about security like a bad guy
- Vulnerabilities that have ruined some days
- Who works on security?
- How to think about defense

What is a Security Bug?

- What is security?
- Class participation: Tacos, Salsa, and Avocados (TSA)

What is security?

“A system is secure if it behaves precisely in the manner intended – and does nothing more” – Ivan Arce

- Who knows exactly what a system is intended to do? Systems are getting more and more complex.
- What types of attacks are possible?

First steps in security: define your security model and your threat model

Threat modeling: T.S.A.

- Logan International Airport security goal #3: prevent banned substances from entering Logan
- Class Participation: What is the threat model?
  - What are possible avenues for getting a banned substance into Logan?
  - Where are the points of entry?
- Threat modeling is also critical, you have to know what you're up against (many engineers don't)

Engineering challenges

- People care about features, not security (until something goes wrong)
- Engineers typically only see a small piece of the puzzle
- "OMG PDF WTF" (Julia Wolf, 2010)
  - How many lines of code in Linux 2.6.32?
  - How many lines in Windows NT 4?
  - How many in Adobe Acrobat?
  - How many lines of code are touched by Google a week?

Engineering challenges

- People care about features, not security (until something goes wrong)
- Engineers typically only see a small piece of the puzzle
- "OMG PDF WTF" (Julia Wolf, 2010)
  - How many lines of code in Linux 2.6.32?
    - 8 – 12.6 million
  - How many lines in Windows NT 4?
    - 11-12 million
  - How many in Adobe Acrobat?
    - 15 million
  - How many lines of code are touched by Google a week?
    - 15 million

Bad Engineering Assumptions

Therac-25 (the engineer)

- Two modes of operation: image and radiation treatment
- Intended invariant: in radiation treatment mode, a protective focusing shield must be in place

Therac-25

Shield code was something like:

```
//global persistent variable, single byte value
ub1  protectiveShield; //zero if shield isn't needed
…
//do we need a shield?
if(treatmentMode) then
{
        protectiveShield++;
} else {
        protectiveShield = 0;
}
…
if(protectiveShield) {
        putShieldInPlace();
} else {
        removeShield();
}
```

Therac-25

- Flawed assumption: protectiveShield would always be non-zero in treatment mode
- Impact: people actually died

Therac-25

- Flawed assumption: protectiveShield would always be non-zero in treatment mode
- Impact: people actually died
- My classmate's conclusion: "I learned to never write medical software"

Bad Assumptions

HOW APPLE AND AMAZON SECURITY FLAWS LED TO MY EPIC HACKING

Bad Assumptions

- Amazon allows you to add a credit card or email address with name, email address, physical address
- Amazon allows you to send a password reset to a registered email address
- Amazon lets you see the last four digits of registered credit card numbers
- Apple grants account access with the last four digits of a registered credit card (D'oh!)
- Gmail reset to Apple account

Bad Assumptions

- **Amazon allows you to add a credit card or email address with name, email address, physical address**
- Amazon allows you to send a password reset to a registered email address
- Amazon lets you see the last four digits of registered credit card numbers
- **Apple grants account access with the last four digits of a registered credit card (D'oh!)**
- Gmail reset to Apple account

Bad Assumptions

Conclusion: components that affect your system are often beyond your control (Facebook, Amazon, Apple). Consider the full threat model.

Bad Assumptions

Conclusion: components that affect your system are often beyond your control (Facebook, Amazon, Apple). Consider the full threat model.

Question: is your personal email account password stronger or weaker than your online banking passwords?

Designing Systems

Think like a security researcher:

- What assumptions are being made?
- Which assumptions are wrong?
- What can you break if the assumption is wrong?

The Confused Deputy

- Tricking an authority into letting you do something you shouldn't be able to do
- Most security problems could fall under this broad definition

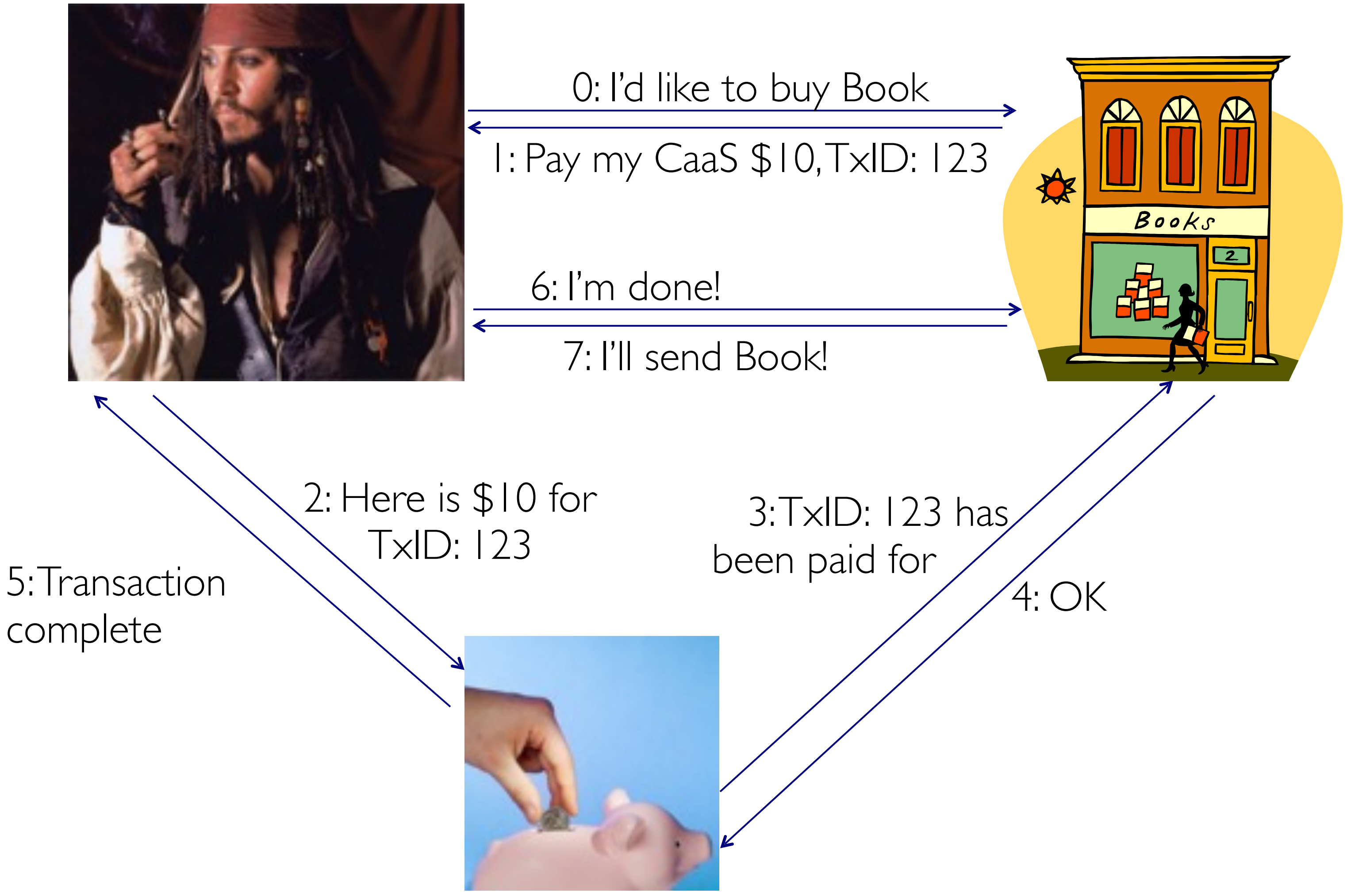
The Confused Deputy

"How to Shop for Free Online"* (security researcher and academic)

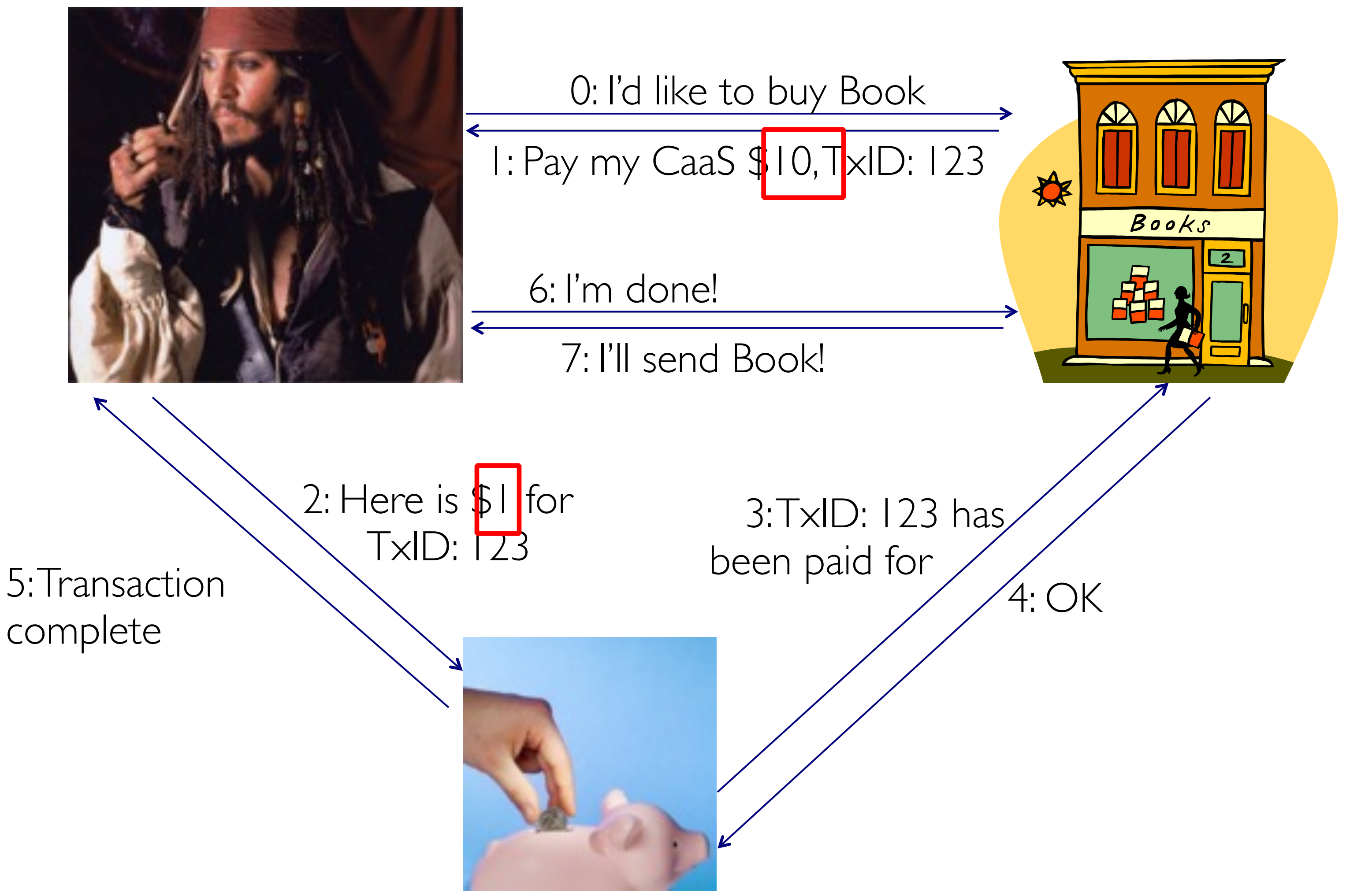
- Three-party payment systems (Cashier as a Service):
  - Merchant (seller)
  - Payment provider
  - ~~Cheater~~ User
- Communication between parties go through the user

* http://research.microsoft.com/pubs/145858/caas-oakland-final.pdf

The Confused Deputy


0: I'd like to buy Book
1: Pay my CaaS $10, TxID: 123
6: I'm done!
7: I'll send Book!
Books
2
2: Here is $10 for TxID: 123
3: TxID: 123 has been paid for
4: OK
5: Transaction complete

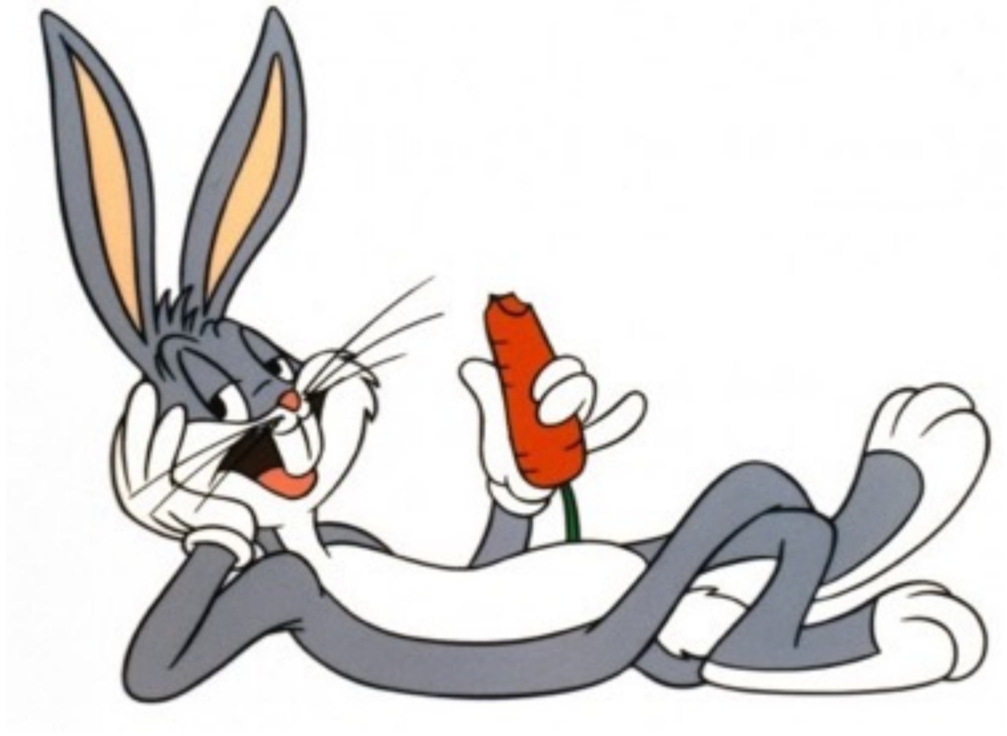
The Confused Deputy



The Confused Deputy

- The merchant thinks something ties the payment amount to the transaction
- Impact: shopping for free
- Solutions?
- Read the paper, lots of things can and do go wrong

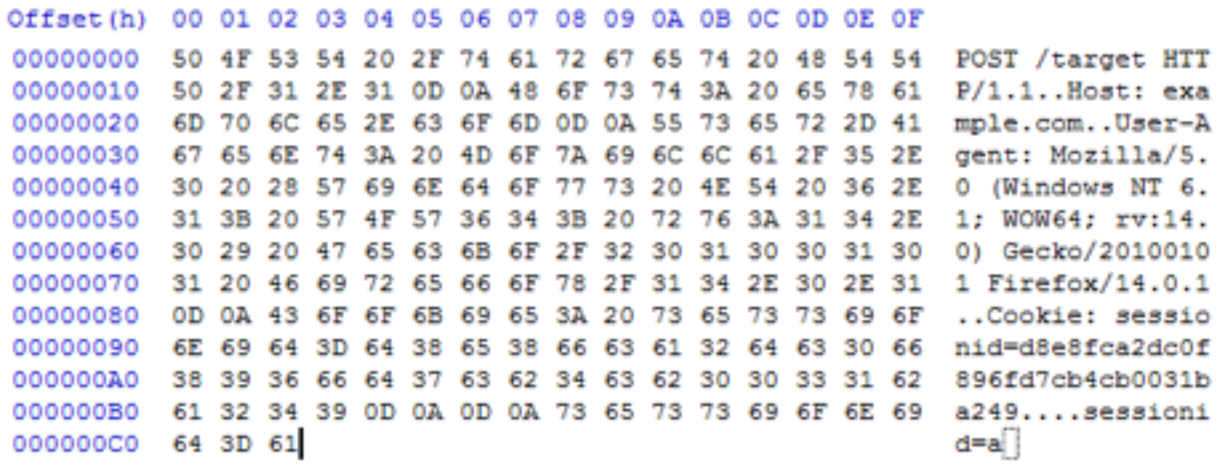
Bad assumptions == security bugs



CRIME

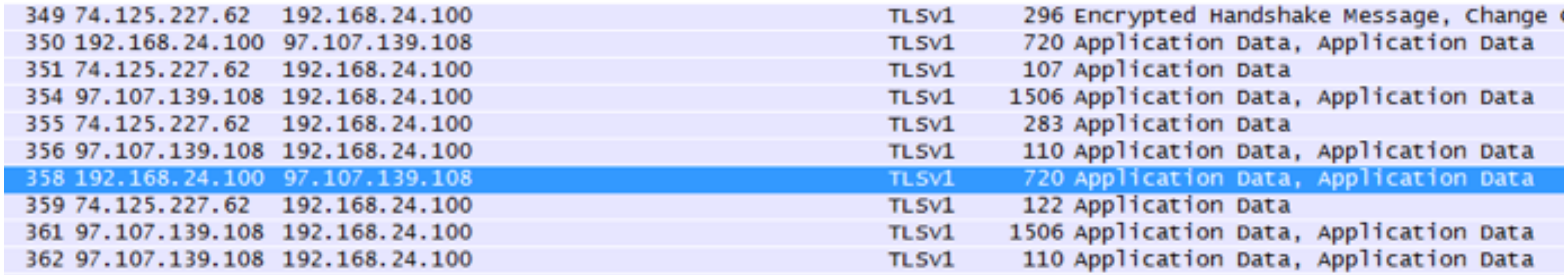
```
POST /target HTTP/1.1
Host: example.com
User-Agent: Mozilla/5.0 (Windows NT 6.1; WOW64; rv:
14.0) Gecko/20100101 Firefox/14.0.1
Cookie: sessionid=d8e8fca2dc0f896fd7cb4cb0031ba249

username=tom&password=hunter2
```

HTTP



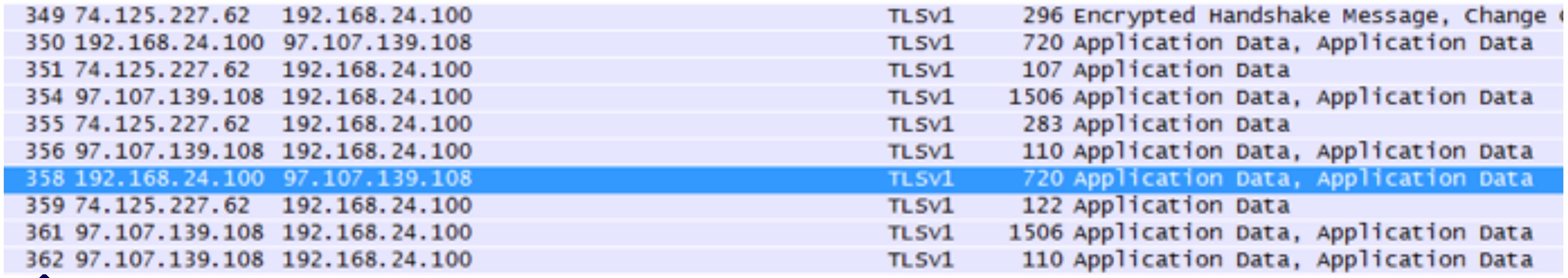
```
Offset(h) 00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F

00000000  50 4F 53 54 20 2F 74 61 72 67 65 74 20 48 54 54  POST /target HTT
00000010  50 2F 31 2E 31 0D 0A 48 6F 73 74 3A 20 65 78 61  P/1.1..Host: exa
00000020  6D 70 6C 65 2E 63 6F 6D 0D 0A 55 73 65 72 2D 41  mple.com..User-A
00000030  67 65 6E 74 3A 20 4D 6F 7A 69 6C 6C 61 2F 35 2E  gent: Mozilla/5.
00000040  30 20 28 57 69 6E 64 6F 77 73 20 4E 54 20 36 2E  0 (Windows NT 6.
00000050  31 3B 20 57 4F 57 36 34 3B 20 72 76 3A 31 34 2E  1; WOW64; rv:14.
00000060  30 29 20 47 65 63 6B 6F 2F 32 30 31 30 30 31 30  0) Gecko/2010010
00000070  31 20 46 69 72 65 66 6F 78 2F 31 34 2E 30 2E 31  1 Firefox/14.0.1
00000080  0D 0A 43 6F 6F 6B 69 65 3A 20 73 65 73 73 69 6F  ..Cookie: sessio
00000090  6E 69 64 3D 64 38 65 38 66 63 61 32 64 63 30 66  nid=d8e8fca2dc0f
000000A0  38 39 36 66 64 37 63 62 34 63 62 30 30 33 31 62  896fd7cb4cb0031b
000000B0  61 32 34 39 0D 0A 0D 0A 73 65 73 73 69 6F 6E 69  a249....sessioni
000000C0  64 3D 61                                         d=a
```

SSL



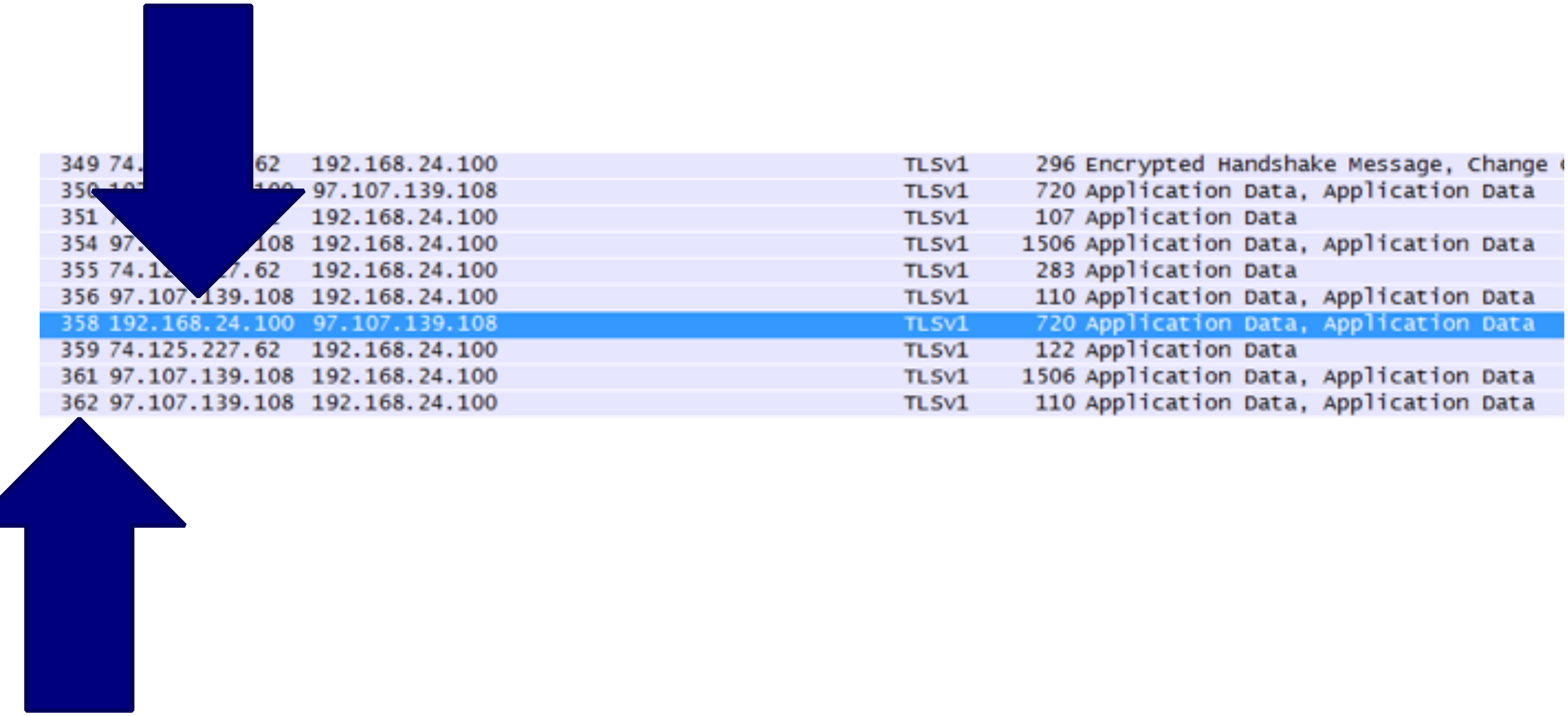
```
349 74.125.227.62   192.168.24.100                      TLSv1      296 Encrypted Handshake Message, Change
350 192.168.24.100  97.107.139.108                      TLSv1      720 Application Data, Application Data
351 74.125.227.62   192.168.24.100                      TLSv1      107 Application Data
354 97.107.139.108  192.168.24.100                      TLSv1     1506 Application Data, Application Data
355 74.125.227.62   192.168.24.100                      TLSv1      283 Application Data
356 97.107.139.108  192.168.24.100                      TLSv1      110 Application Data, Application Data
358 192.168.24.100  97.107.139.108                      TLSv1      720 Application Data, Application Data
359 74.125.227.62   192.168.24.100                      TLSv1      122 Application Data
361 97.107.139.108  192.168.24.100                      TLSv1     1506 Application Data, Application Data
362 97.107.139.108  192.168.24.100                      TLSv1      110 Application Data, Application Data
```

Time



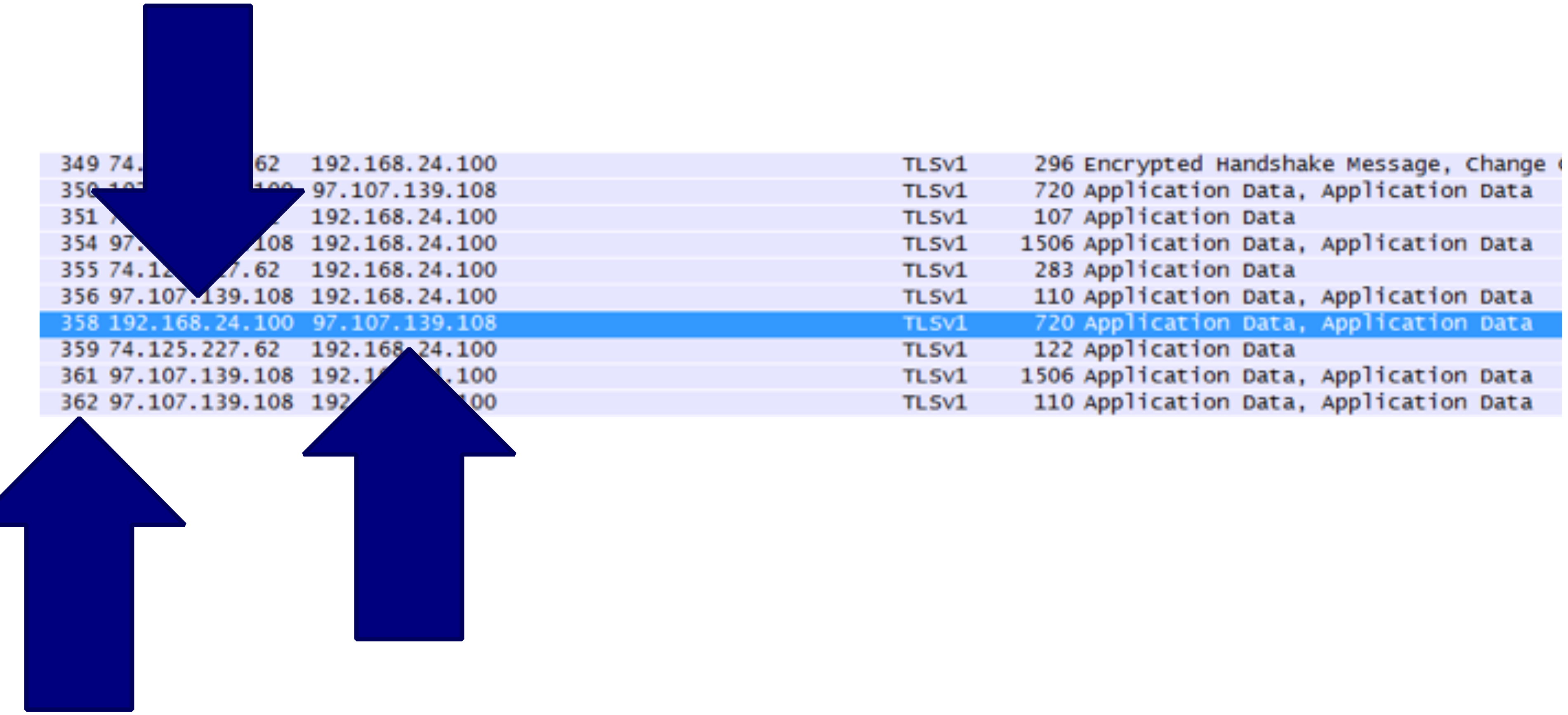
```
349 74.125.227.62   192.168.24.100                              TLSv1      296 Encrypted Handshake Message, Change
350 192.168.24.100  97.107.139.108                              TLSv1      720 Application Data, Application Data
351 74.125.227.62   192.168.24.100                              TLSv1      107 Application Data
354 97.107.139.108  192.168.24.100                              TLSv1     1506 Application Data, Application Data
355 74.125.227.62   192.168.24.100                              TLSv1      283 Application Data
356 97.107.139.108  192.168.24.100                              TLSv1      110 Application Data, Application Data
358 192.168.24.100  97.107.139.108                              TLSv1      720 Application Data, Application Data
359 74.125.227.62   192.168.24.100                              TLSv1      122 Application Data
361 97.107.139.108  192.168.24.100                              TLSv1     1506 Application Data, Application Data
362 97.107.139.108  192.168.24.100                              TLSv1      110 Application Data, Application Data
```

From



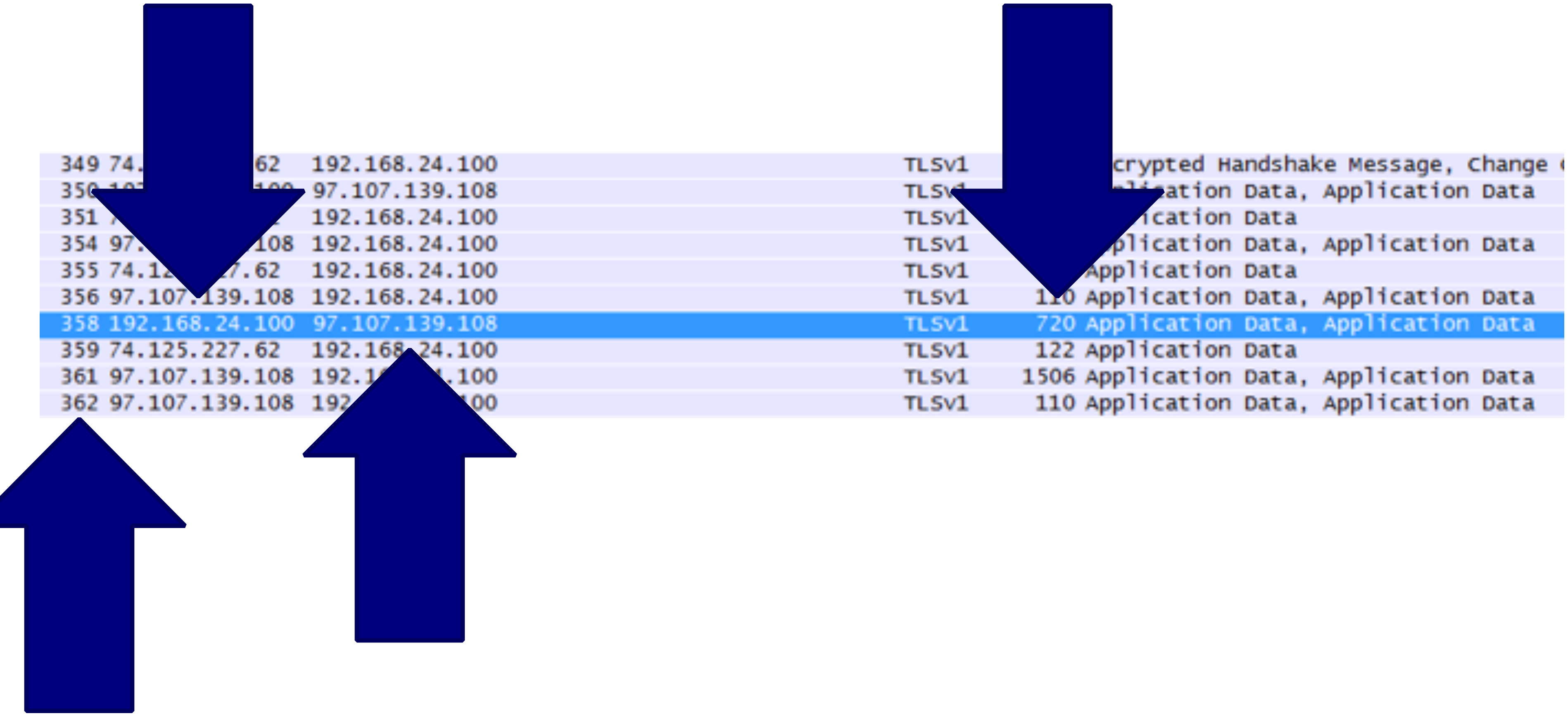
```
349 74.        62   192.168.24.100     TLSv1      296 Encrypted Handshake Message, Change
350             97.107.139.108     TLSv1      720 Application Data, Application Data
351 7              192.168.24.100     TLSv1      107 Application Data
354 97.        108  192.168.24.100     TLSv1     1506 Application Data, Application Data
355 74.12      7.62 192.168.24.100     TLSv1      283 Application Data
356 97.107.139.108 192.168.24.100     TLSv1      110 Application Data, Application Data
358 192.168.24.100 97.107.139.108     TLSv1      720 Application Data, Application Data
359 74.125.227.62  192.168.24.100     TLSv1      122 Application Data
361 97.107.139.108 192.168.24.100     TLSv1     1506 Application Data, Application Data
362 97.107.139.108 192.168.24.100     TLSv1      110 Application Data, Application Data
```

To



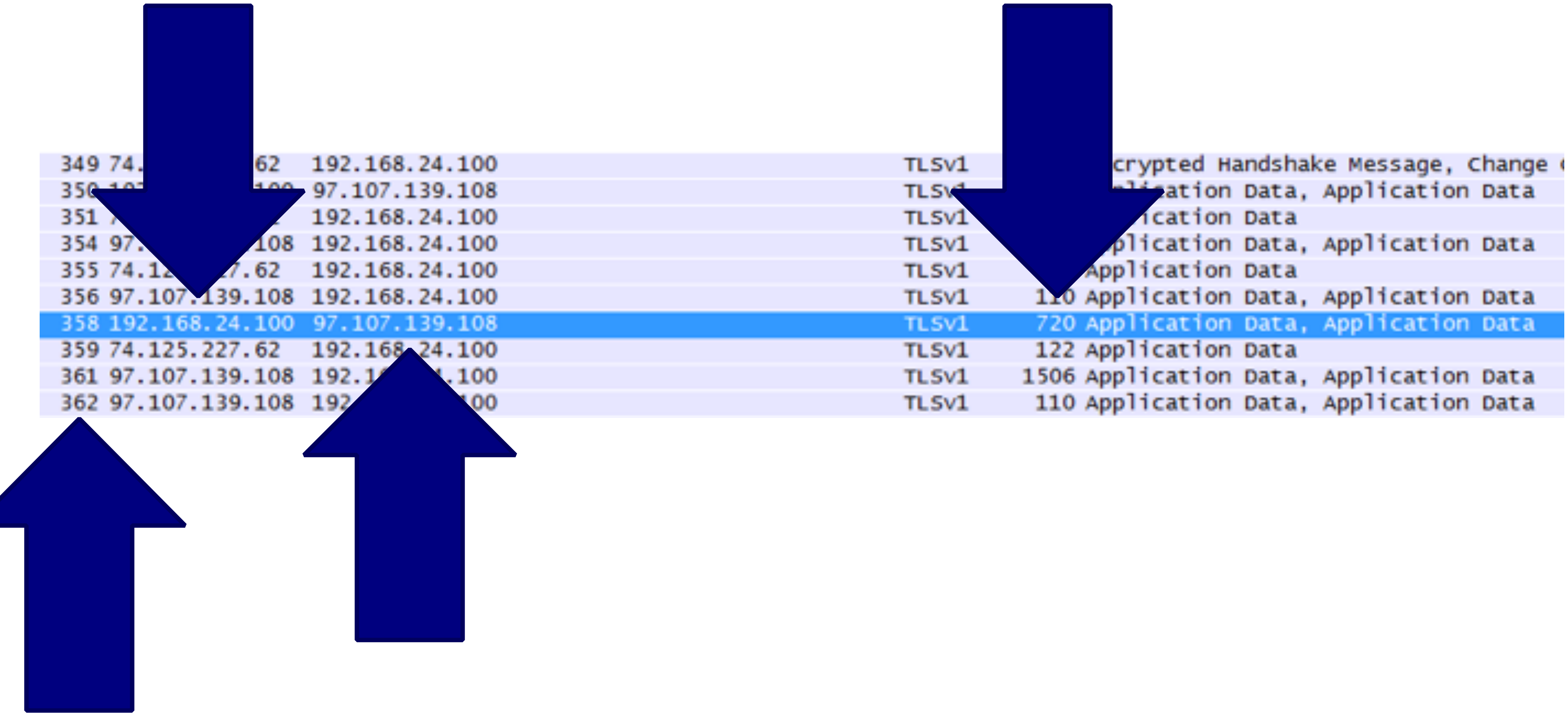
```
349 74.        62   192.168.24.100        TLSv1     296 Encrypted Handshake Message, Change
350                 97.107.139.108        TLSv1     720 Application Data, Application Data
351 7               192.168.24.100        TLSv1     107 Application Data
354 97.        108  192.168.24.100        TLSv1    1506 Application Data, Application Data
355 74.12      7.62 192.168.24.100        TLSv1     283 Application Data
356 97.107.139.108  192.168.24.100        TLSv1     110 Application Data, Application Data
358 192.168.24.100  97.107.139.108        TLSv1     720 Application Data, Application Data
359 74.125.227.62   192.168 24.100        TLSv1     122 Application Data
361 97.107.139.108  192.1     .100        TLSv1    1506 Application Data, Application Data
362 97.107.139.108  192         00        TLSv1     110 Application Data, Application Data
```

Length



```
349 74.       62   192.168.24.100        TLSv1        crypted Handshake Message, Change (
350            97.107.139.108        TLSv          ation Data, Application Data
351            192.168.24.100        TLSv1          ication Data
354 97.      108 192.168.24.100        TLSv1         plication Data, Application Data
355 74.12   7.62   192.168.24.100        TLSv1        Application Data
356 97.107.139.108 192.168.24.100        TLSv1     110 Application Data, Application Data
358 192.168.24.100 97.107.139.108        TLSv1     720 Application Data, Application Data
359 74.125.227.62  192.168 24.100        TLSv1     122 Application Data
361 97.107.139.108 192.1   .100          TLSv1    1506 Application Data, Application Data
362 97.107.139.108 192      00           TLSv1     110 Application Data, Application Data
```

Traffic Analysis. Huge Field



```
349 74.       62   192.168.24.100        TLSv1       crypted Handshake Message, Change
350           100  97.107.139.108        TLSv            ation Data, Application Data
351                192.168.24.100        TLSv1           ication Data
354 97.        108 192.168.24.100        TLSv1         plication Data, Application Data
355 74.12    7.62  192.168.24.100        TLSv1         Application Data
356 97.107.139.108 192.168.24.100        TLSv1     1 0 Application Data, Application Data
358 192.168.24.100 97.107.139.108        TLSv1     720 Application Data, Application Data
359 74.125.227.62  192.168 24.100        TLSv1     122 Application Data
361 97.107.139.108 192.1    .100         TLSv1    1506 Application Data, Application Data
362 97.107.139.108 192      00           TLSv1     110 Application Data, Application Data
```

HTTP

```
POST /target HTTP/1.1
Host: example.com
User-Agent: Mozilla/5.0 (Windows NT 6.1; WOW64; rv:
14.0) Gecko/20100101 Firefox/14.0.1
Cookie: sessionid=d8e8fca2dc0f896fd7cb4cb0031ba249

username=tom&password=hunter2
```

HTTP

```
POST /target HTTP/1.1
Host: example.com
User-Agent: Mozilla/5.0 (Windows NT 6.1; WOW64; rv:
14.0) Gecko/20100101 Firefox/14.0.1
Cookie: sessionid=d8e8fca2dc0f896fd7cb4cb0031ba249

username=tom&password=hunter
```

Attacker wants to know this

Attacker Can Control

```
POST /target HTTP/1.1
Host: example.com
User-Agent: Mozilla/5.0 (Windows NT 6.1; WOW64; rv:
14.0) Gecko/20100101 Firefox/14.0.1
Cookie: sessionid=d8e8fca2dc0f896fd7cb4cb0031ba249

username=tom&password=hunter2
```

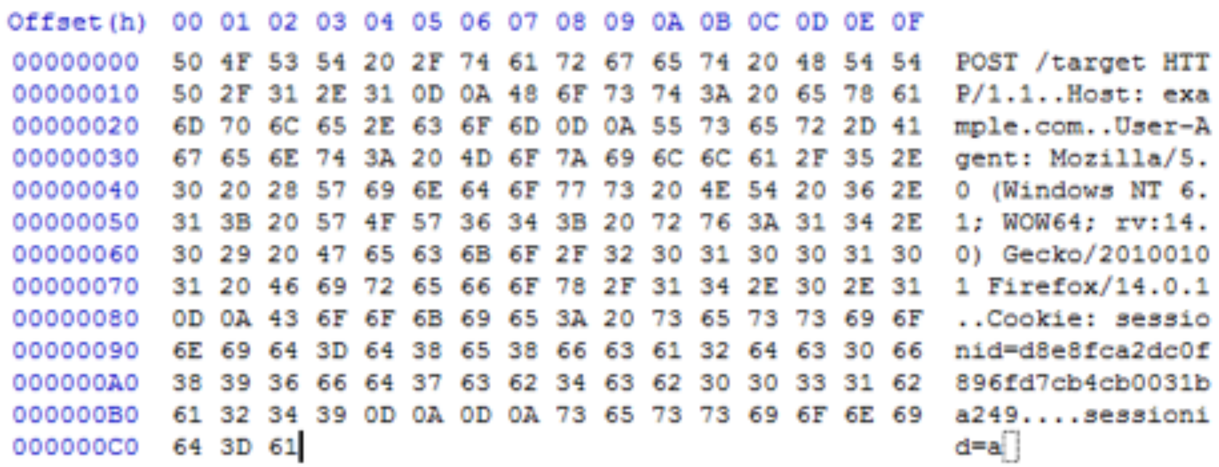
HTTP

```
POST /target HTTP/1.1
Host: example.com
User-Agent: Mozilla/5.0 (Windows NT 6.1; WOW64; rv:
14.0) Gecko/20100101 Firefox/14.0.1
Cookie: sessionid=d8e8fca2dc0f896fd7cb4cb0031ba249

username=tom&password=hunter2
```

HTTP

```
POST /target HTTP/1.1
Host: example.com
User-Agent: Mozilla/5.0 (Windows NT 6.1; WOW64; rv:
14.0) Gecko/20100101 Firefox/14.0.1
Cookie: sessionid=d8e8fca2dc0f896fd7cb4cb0031ba249

sessionid=a
```

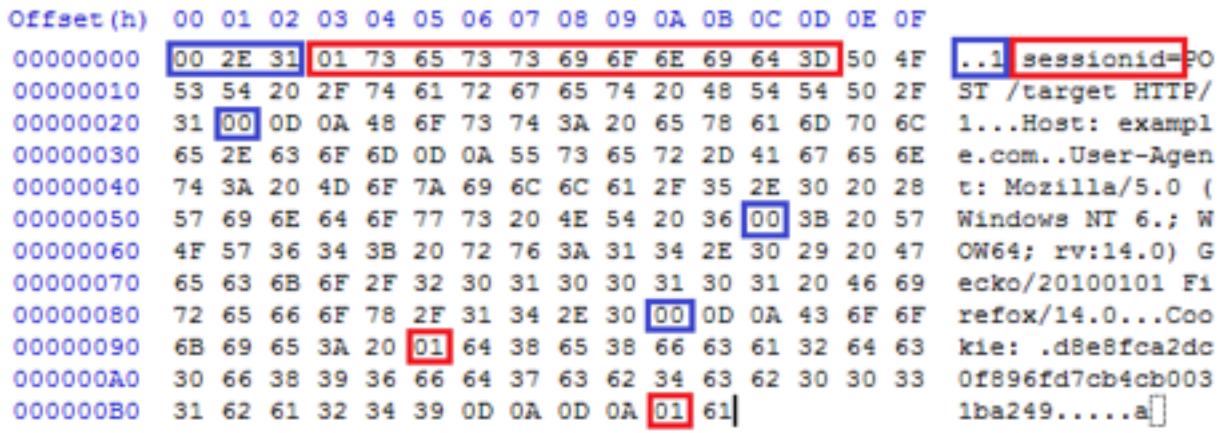
HTTP



```
Offset(h) 00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F

00000000  50 4F 53 54 20 2F 74 61 72 67 65 74 20 48 54 54  POST /target HTT
00000010  50 2F 31 2E 31 0D 0A 48 6F 73 74 3A 20 65 78 61  P/1.1..Host: exa
00000020  6D 70 6C 65 2E 63 6F 6D 0D 0A 55 73 65 72 2D 41  mple.com..User-A
00000030  67 65 6E 74 3A 20 4D 6F 7A 69 6C 6C 61 2F 35 2E  gent: Mozilla/5.
00000040  30 20 28 57 69 6E 64 6F 77 73 20 4E 54 20 36 2E  0 (Windows NT 6.
00000050  31 3B 20 57 4F 57 36 34 3B 20 72 76 3A 31 34 2E  1; WOW64; rv:14.
00000060  30 29 20 47 65 63 6B 6F 2F 32 30 31 30 30 31 30  0) Gecko/2010010
00000070  31 20 46 69 72 65 66 6F 78 2F 31 34 2E 30 2E 31  1 Firefox/14.0.1
00000080  0D 0A 43 6F 6F 6B 69 65 3A 20 73 65 73 73 69 6F  ..Cookie: sessio
00000090  6E 69 64 3D 64 38 65 38 66 63 61 32 64 63 30 66  nid=d8e8fca2dc0f
000000A0  38 39 36 66 64 37 63 62 34 63 62 30 30 33 31 62  896fd7cb4cb0031b
000000B0  61 32 34 39 0D 0A 0D 0A 73 65 73 73 69 6F 6E 69  a249....sessioni
000000C0  64 3D 61                                         d=a
```

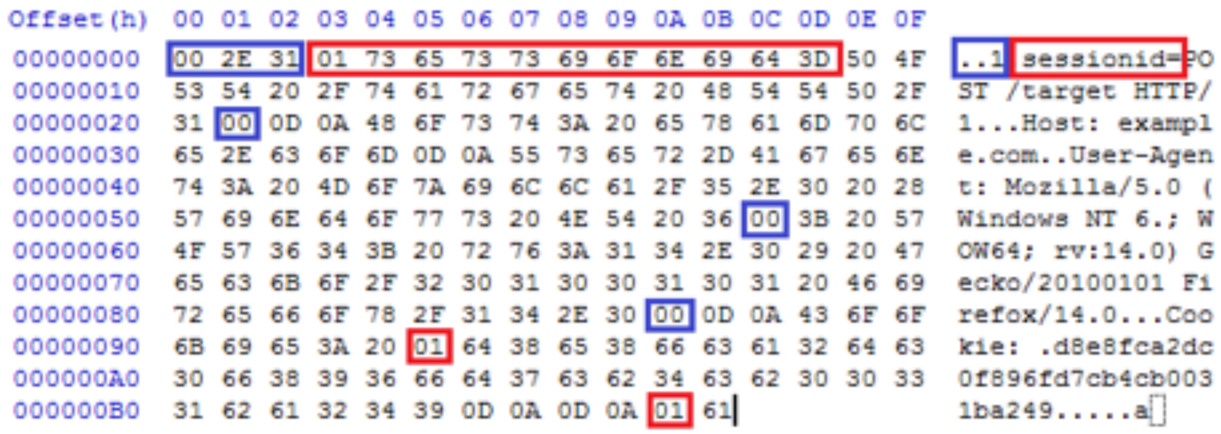
195 Bytes

HTTP



```
Offset(h) 00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F

00000000  00 2E 31 01 73 65 73 73 69 6F 6E 69 64 3D 50 4F  ..1 sessionid=PO
00000010  53 54 20 2F 74 61 72 67 65 74 20 48 54 54 50 2F  ST /target HTTP/
00000020  31 00 0D 0A 48 6F 73 74 3A 20 65 78 61 6D 70 6C  1...Host: exampl
00000030  65 2E 63 6F 6D 0D 0A 55 73 65 72 2D 41 67 65 6E  e.com..User-Agen
00000040  74 3A 20 4D 6F 7A 69 6C 6C 61 2F 35 2E 30 20 28  t: Mozilla/5.0 (
00000050  57 69 6E 64 6F 77 73 20 4E 54 20 36 00 3B 20 57  Windows NT 6.; W
00000060  4F 57 36 34 3B 20 72 76 3A 31 34 2E 30 29 20 47  OW64; rv:14.0) G
00000070  65 63 6B 6F 2F 32 30 31 30 30 31 30 31 20 46 69  ecko/20100101 Fi
00000080  72 65 66 6F 78 2F 31 34 2E 30 00 0D 0A 43 6F 6F  refox/14.0...Coo
00000090  6B 69 65 3A 20 01 64 38 65 38 66 63 61 32 64 63  kie: .d8e8fca2dc
000000A0  30 66 38 39 36 66 64 37 63 62 34 63 62 30 30 33  0f896fd7cb4cb003
000000B0  31 62 61 32 34 39 0D 0A 0D 0A 01 61              1ba249.....a
```

HTTP



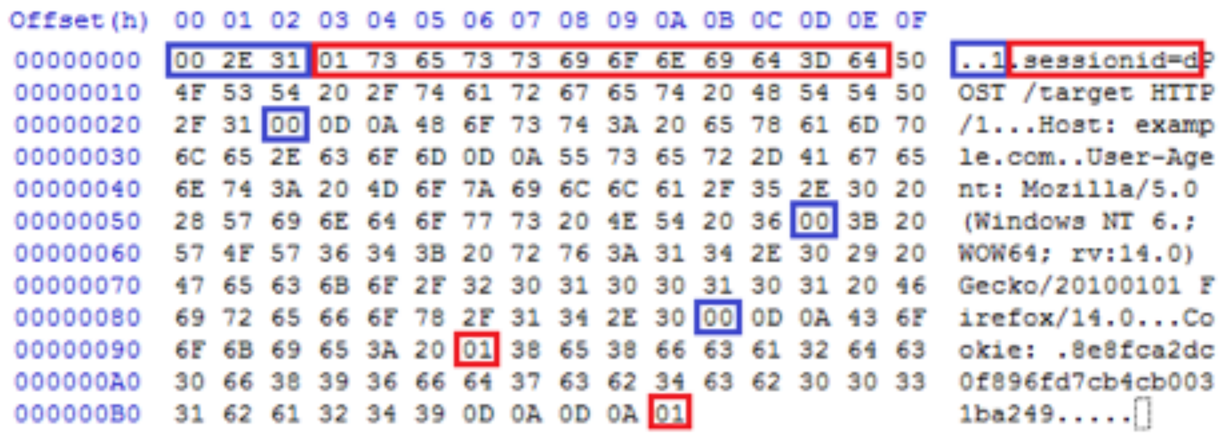
```
Offset(h) 00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F

00000000  00 2E 31 01 73 65 73 73 69 6F 6E 69 64 3D 50 4F  ..1 sessionid=PO
00000010  53 54 20 2F 74 61 72 67 65 74 20 48 54 54 50 2F  ST /target HTTP/
00000020  31 00 0D 0A 48 6F 73 74 3A 20 65 78 61 6D 70 6C  1...Host: exampl
00000030  65 2E 63 6F 6D 0D 0A 55 73 65 72 2D 41 67 65 6E  e.com..User-Agen
00000040  74 3A 20 4D 6F 7A 69 6C 6C 61 2F 35 2E 30 20 28  t: Mozilla/5.0 (
00000050  57 69 6E 64 6F 77 73 20 4E 54 20 36 00 3B 20 57  Windows NT 6.; W
00000060  4F 57 36 34 3B 20 72 76 3A 31 34 2E 30 29 20 47  OW64; rv:14.0) G
00000070  65 63 6B 6F 2F 32 30 31 30 30 31 30 31 20 46 69  ecko/20100101 Fi
00000080  72 65 66 6F 78 2F 31 34 2E 30 00 0D 0A 43 6F 6F  refox/14.0...Coo
00000090  6B 69 65 3A 20 01 64 38 65 38 66 63 61 32 64 63  kie: .d8e8fca2dc
000000A0  30 66 38 39 36 66 64 37 63 62 34 63 62 30 30 33  0f896fd7cb4cb003
000000B0  31 62 61 32 34 39 0D 0A 0D 0A 01 61              1ba249.....a
```

187 Bytes

HTTP

```
POST /target HTTP/1.1
Host: example.com
User-Agent: Mozilla/5.0 (Windows NT 6.1; WOW64; rv:
14.0) Gecko/20100101 Firefox/14.0.1
Cookie: sessionid=d8e8fca2dc0f896fd7cb4cb0031ba249

sessionid=d
```

HTTP



```
Offset(h) 00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F

00000000  00 2E 31 01 73 65 73 73 69 6F 6E 69 64 3D 64 50  ..1.sessionid=dP
00000010  4F 53 54 20 2F 74 61 72 67 65 74 20 48 54 54 50  OST /target HTTP
00000020  2F 31 00 0D 0A 48 6F 73 74 3A 20 65 78 61 6D 70  /1...Host: examp
00000030  6C 65 2E 63 6F 6D 0D 0A 55 73 65 72 2D 41 67 65  le.com..User-Age
00000040  6E 74 3A 20 4D 6F 7A 69 6C 6C 61 2F 35 2E 30 20  nt: Mozilla/5.0
00000050  28 57 69 6E 64 6F 77 73 20 4E 54 20 36 00 3B 20  (Windows NT 6.;
00000060  57 4F 57 36 34 3B 20 72 76 3A 31 34 2E 30 29 20  WOW64; rv:14.0)
00000070  47 65 63 6B 6F 2F 32 30 31 30 30 31 30 31 20 46  Gecko/20100101 F
00000080  69 72 65 66 6F 78 2F 31 34 2E 30 00 0D 0A 43 6F  irefox/14.0...Co
00000090  6F 6B 69 65 3A 20 01 38 65 38 66 63 61 32 64 63  okie: .8e8fca2dc
000000A0  30 66 38 39 36 66 64 37 63 62 34 63 62 30 30 33  0f896fd7cb4cb003
000000B0  31 62 61 32 34 39 0D 0A 0D 0A 01                 1ba249.....
```

186 Bytes

HTTP

```
POST /target HTTP/1.1
Host: example.com
User-Agent: Mozilla/5.0 (Windows NT 6.1; WOW64; rv:
14.0) Gecko/20100101 Firefox/14.0.1
Cookie: sessionid=d8e8fca2dc0f896fd7cb4cb0031ba249

sessionid=da
```

HTTP

```
POST /target HTTP/1.1
Host: example.com
User-Agent: Mozilla/5.0 (Windows NT 6.1; WOW64; rv:
14.0) Gecko/20100101 Firefox/14.0.1
Cookie: sessionid=d8e8fca2dc0f896fd7cb4cb0031ba249


sessionid=da
```

188 Bytes

HTTP

```
POST /target HTTP/1.1
Host: example.com
User-Agent: Mozilla/5.0 (Windows NT 6.1; WOW64; rv:
14.0) Gecko/20100101 Firefox/14.0.1
Cookie: sessionid=d8e8fca2dc0f896fd7cb4cb0031ba249

sessionid=d8
```

187 Bytes

Fighting CRIME

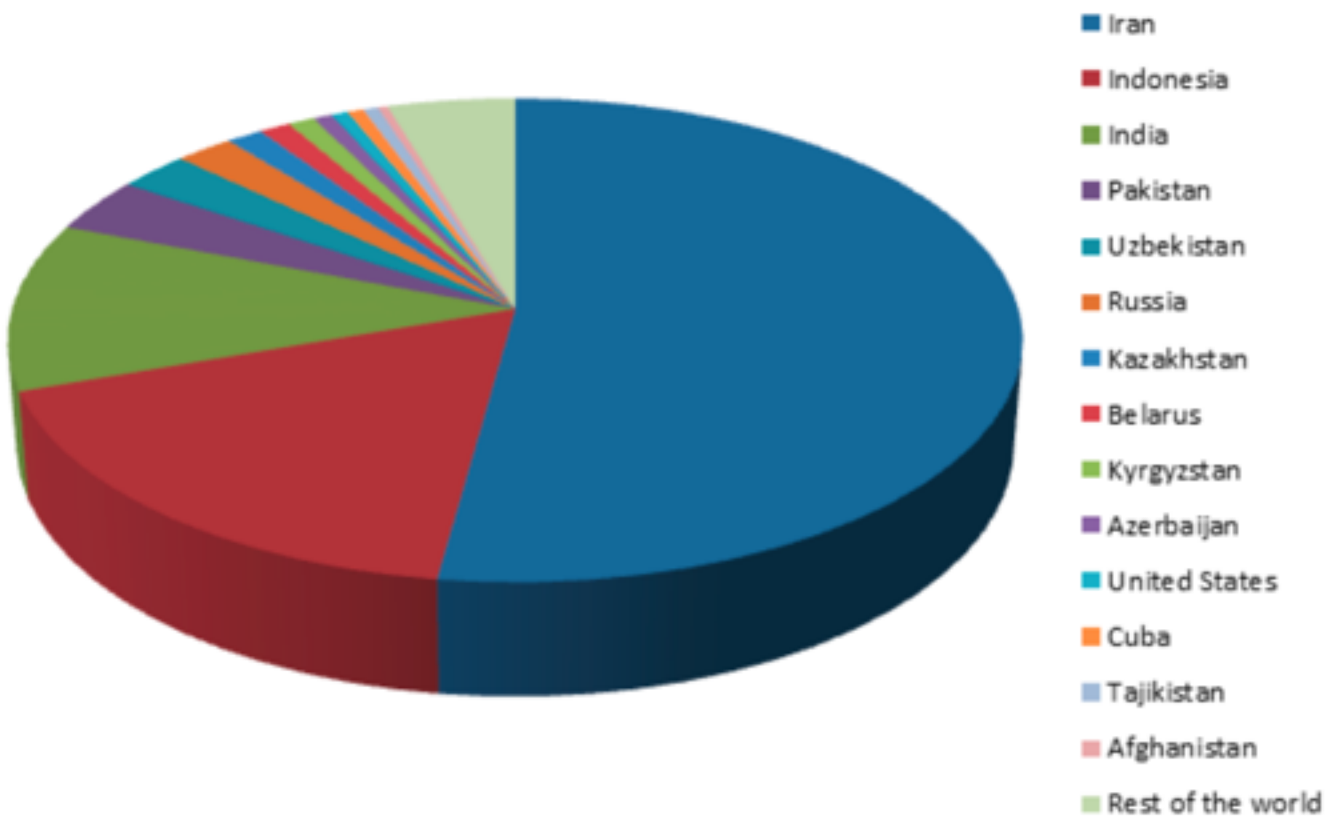
- Browsers disabled TLS compression
- SPDY revised so request secrets are compressed in a separate context

Stuxnet (gov't / security researcher)

Stuxnet (you brilliant bums)

- [worm [rootkit [rootkit [sabotage]]]]
- Five zero-day vulnerabilities
- Two stolen certificates
- Almost surgically targeted
- Eight propagation methods
- Partridge in a malware pear tree

Stuxnet



http://www.eset.com/resources/white-papers/Stuxnet_Under_the_Microscope.pdf

The Target

- Mixed MS Windows environment = *Redundant*
- Not exploiting memory corruption = *Reliable*
- Target: Iranian air-gapped networks operating centrifuges to enrich nuclear material (Natanz)
- How can you get a foot in the door? USB keys

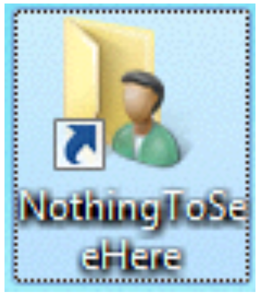
USB Vulnerability

Zero-Day* Vulnerabilities:

- **MS10-046 (Shell LNK / Shortcut)**
- MS10-061 (Print Spooler Service)
- MS10-073 (Win32K Keyboard Layout)
- MS08-067 (NetPathCanonicalize()), (Patched)
  http://www.phreedom.org/blog/2008/decompiling-ms08-067/
- MS10-092 (Task Scheduler)
- CVE-2010-2772 (Siemens SIMATIC Static Password)

MS10-046 (Shell LNK/Shortcut)

- You know, shortcuts and such
- Where does the icon come from?
- Loaded from a CPL (Control Panel File) specified by the user
- A CPL is just a DLL
- USB keys have attack DLL and a shortcut referencing the DLL
- Plugging in the USB stick leads to arbitrary code execution



MS10-046 (Shell LNK/Shortcut)

Flaw: we should run a user-specified DLL to display an icon for a shortcut?!

But I’m not Admin!

Zero-Day* Vulnerabilities:

- MS10-046 (Shell LNK / Shortcut)
- MS10-061 (Print Spooler Service)
- **MS10-073 (Win32K Keyboard Layout)**
- MS08-067 (NetPathCanonicalize()), (Patched)
  http://www.phreedom.org/blog/2008/decompiling-ms08-067/
- MS10-092 (Task Scheduler)
- CVE-2010-2772 (Siemens SIMATIC Static Password)

MS10-073 (Win32K Keyboard Layout)

- Keyboard layouts can be loaded into Windows
- In XP, anyone can load a keyboard layout (later version only allow admins)
- Integer in the layout file indexes a global array of function pointers without proper bound checking
- Call any function, but I want to call *my* function…

MS10-073 (Win32K Keyboard Layout)

- How do we call attack code?
- Find the pointer to the global function array
- Find a pointer into user-land (modifiable by your program)
- Inject your attack code there
- Call the modified function (runs as SYSTEM)

MS10-073 (Win32K Keyboard Layout)

Flaws: improper bound checking on the keyboard layout function index and allowing standard users to specify layouts

But I'm not an Admin!

Zero-Day* Vulnerabilities:

- MS10-046 (Shell LNK / Shortcut)
- MS10-061 (Print Spooler Service)
- MS10-073 (Win32K Keyboard Layout)
- MS08-067 (NetPathCanonicalize()), (Patched)

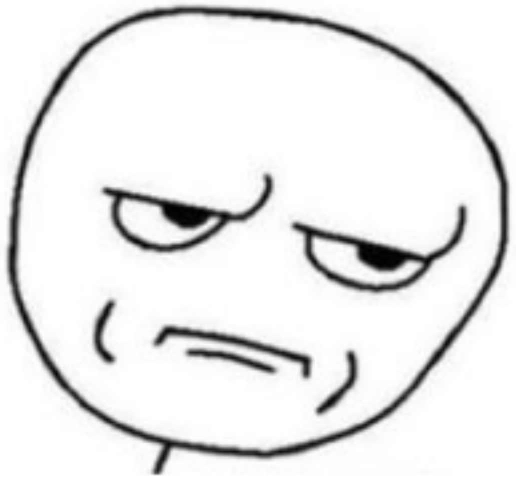
  http://www.phreedom.org/blog/2008/decompiling-ms08-067/
- **MS10-092 (Task Scheduler)**
- CVE-2010-2772 (Siemens SIMATIC Static Password)

MS10-092 (Task Scheduler)

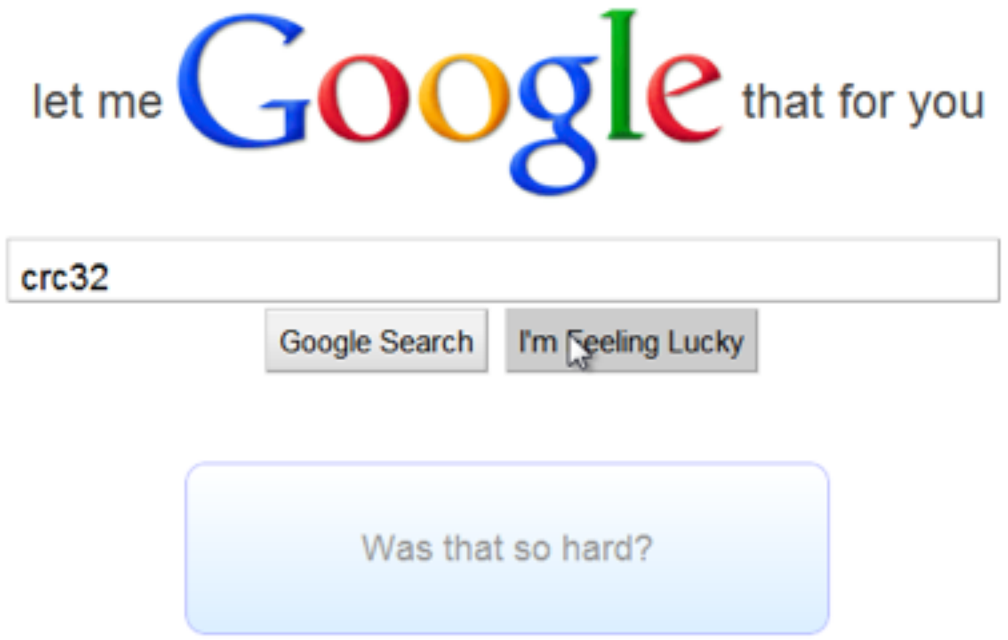
- Standard users can create and edit scheduled tasks (XML)
- After a task is created, a CRC32 checksum is generated to prevent tampering
- … CRC32 …

MS10-092 (Task Scheduler)

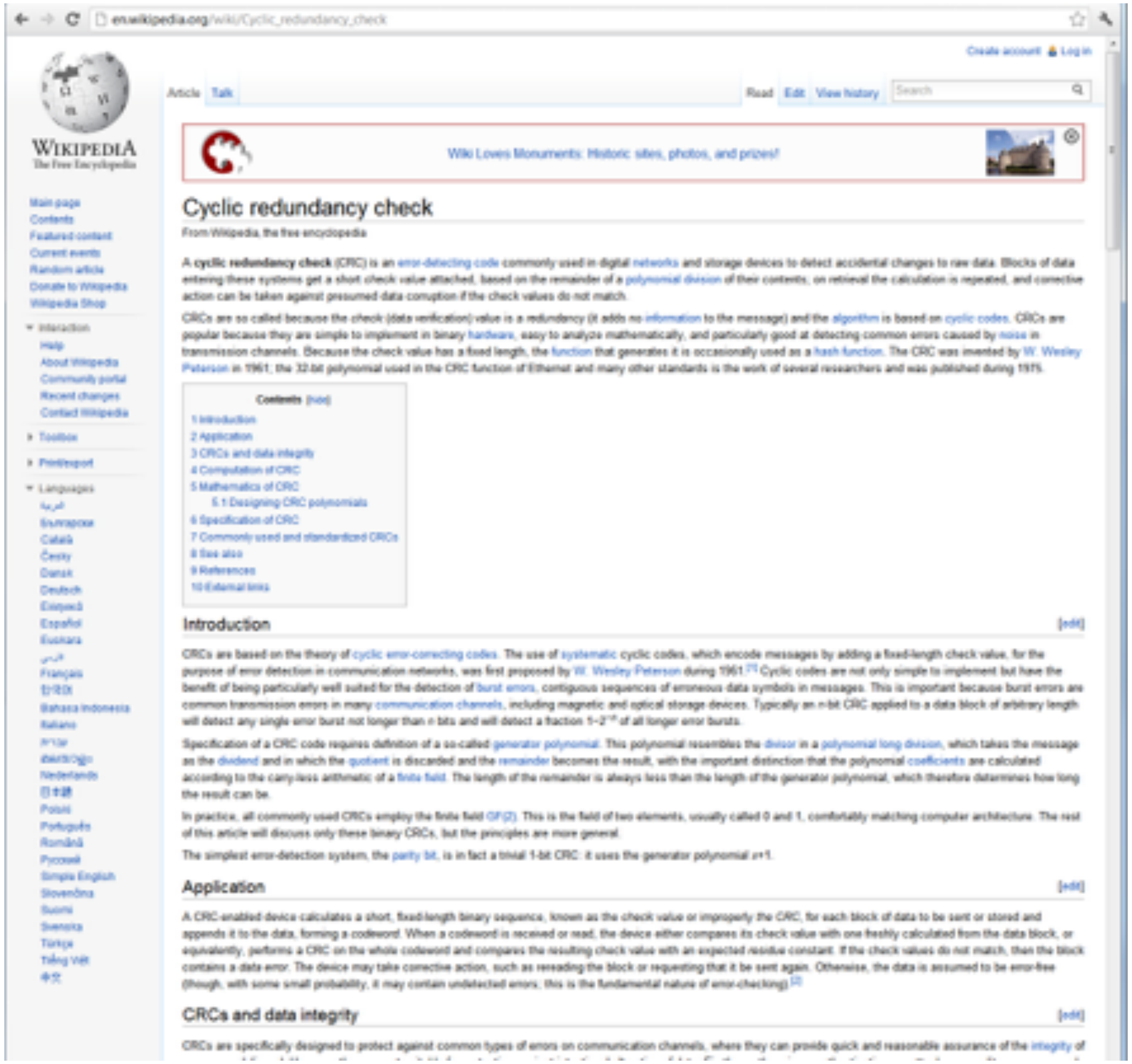
- Standard users can create and edit scheduled tasks (XML)
- After a task is created, a CRC32 checksum is generated to prevent tampering
- … CRC32 …



CRC32


let me Google that for you
crc32
Google Search
I'm Feeling Lucky
Was that so hard?

CRC32



Cyclic redundancy check

From Wikipedia, the free encyclopedia

A **cyclic redundancy check** (CRC) is an error-detecting code commonly used in digital networks and storage devices to detect accidental changes to raw data. Blocks of data entering these systems get a short check value attached, based on the remainder of a polynomial division of their contents; on retrieval the calculation is repeated, and corrective action can be taken against presumed data corruption if the check values do not match.

CRCs are so called because the check (data verification) value is a redundancy (it adds no information to the message) and the algorithm is based on cyclic codes. CRCs are popular because they are simple to implement in binary hardware, easy to analyze mathematically, and particularly good at detecting common errors caused by noise in transmission channels. Because the check value has a fixed length, the function that generates it is occasionally used as a hash function. The CRC was invented by W. Wesley Peterson in 1961; the 32-bit polynomial used in the CRC function of Ethernet and many other standards is the work of several researchers and was published during 1975.

Contents [hide]

Introduction [edit]

CRCs are based on the theory of cyclic error-correcting codes. The use of systematic cyclic codes, which encode messages by adding a fixed-length check value, for the purpose of error detection in communication networks, was first proposed by W. Wesley Peterson during 1961.[1] Cyclic codes are not only simple to implement but have the benefit of being particularly well suited for the detection of burst errors, contiguous sequences of erroneous data symbols in messages. This is important because burst errors are common transmission errors in many communication channels, including magnetic and optical storage devices. Typically an n-bit CRC applied to a data block of arbitrary length will detect any single error burst not longer than n bits and will detect a fraction $1-2^{-n}$ of all longer error bursts.

Specification of a CRC code requires definition of a so-called generator polynomial. This polynomial resembles the divisor in a polynomial long division, which takes the message as the dividend and in which the quotient is discarded and the remainder becomes the result, with the important distinction that the polynomial coefficients are calculated according to the carry-less arithmetic of a finite field. The length of the remainder is always less than the length of the generator polynomial, which therefore determines how long the result can be.

In practice, all commonly used CRCs employ the finite field GF(2). This is the field of two elements, usually called 0 and 1, comfortably matching computer architecture. The rest of this article will discuss only these binary CRCs, but the principles are more general.

The simplest error-detection system, the parity bit, is in fact a trivial 1-bit CRC: it uses the generator polynomial $x+1$.

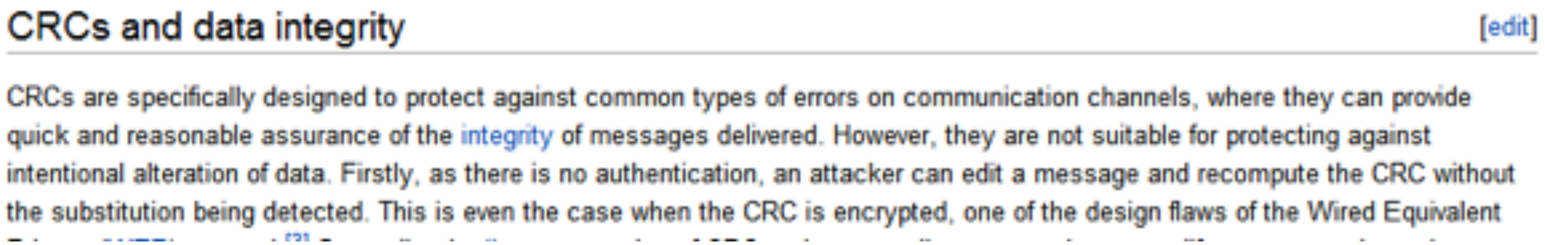
Application [edit]

A CRC-enabled device calculates a short, fixed-length binary sequence, known as the check value or improperly the CRC, for each block of data to be sent or stored and appends it to the data, forming a codeword. When a codeword is received or read, the device either compares its check value with one freshly calculated from the data block, or equivalently, performs a CRC on the whole codeword and compares the resulting check value with an expected residue constant. If the check values do not match, then the block contains a data error. The device may take corrective action, such as rereading the block or requesting that it be sent again. Otherwise, the data is assumed to be error-free (though, with some small probability, it may contain undetected errors; this is the fundamental nature of error-checking).[2]

CRCs and data integrity [edit]

CRCs are specifically designed to protect against common types of errors on communication channels, where they can provide quick and reasonable assurance of the integrity of

Enhance!



CRCs and data integrity [edit]

CRCs are specifically designed to protect against common types of errors on communication channels, where they can provide quick and reasonable assurance of the integrity of messages delivered. However, they are not suitable for protecting against intentional alteration of data. Firstly, as there is no authentication, an attacker can edit a message and recompute the CRC without the substitution being detected. This is even the case when the CRC is encrypted, one of the design flaws of the Wired Equivalent

"However, [CRCs] are not suitable for protecting against intentional alteration of data." – Wikipedia (Cyclic redundancy check)

MS10-092 (Task Scheduler)

- Created task as normal user, record CRC32 value
- Modified user definition in the task to LocalSystem
- Take CRC32 of the task XML, pad until the CRC32 matches original

MS10-092 (Task Scheduler)

- Created task as normal user, record CRC32 value
- Modified user definition in the task to LocalSystem
- Take CRC32 of the task XML, pad until the CRC32 matches original
- ?????
- Profit!

MS10-092 (Task Scheduler)

Flaw:

We should use CRC32 to-
NEVER USE CRC32 FOR ANYTHING!
quickmeme.com

MS10-092 (Task Scheduler)

"Our job is to read one more sentence in the man page than the developer did." –Chris Palmer (former iSECer)

- Be really curious
- Think about how components interact with each other

Let's Spread!

Zero-Day* Vulnerabilities:

- MS10-046 (Shell LNK / Shortcut)
- **MS10-061 (Print Spooler Service)**
- MS10-073 (Win32K Keyboard Layout)
- MS08-067 (NetPathCanonicalize()), (Patched)
  http://www.phreedom.org/blog/2008/decompiling-ms08-067/
- MS10-092 (Task Scheduler)
- CVE-2010-2772 (Siemens SIMATIC Static Password)

MS10-061 (Print Spooler Service)

- Enumerates printer shares
- Connects to printer and asks to print two files to SYSTEM32
- Should fail?! Printer should connect as Guest, which shouldn't have privilege to create files in SYSTEM32

MS10-061 (Print Spooler Service)

- `“//We run as system because in XP the guest account doesn’t have enough privilege to do X/Y/Z”`
- Stuxnet payload is dropped

MS10-061 (Print Spooler Service)

- How do we execute? Enter the MOF
- MOF files are basically script files
- A process monitors the following directory for new files and executes them: Windows\System32\wbem\mof\
- MOF file executes the Stuxnet payload

MS10-061 (Print Spooler Service)

Flaws:

- Printer spooler runs as SYSTEM (highest privilege) and allows arbitrary files to be written to arbitrary places
- File creation leads to arbitrary code execution

Let's Spread!

Zero-Day* Vulnerabilities:

- MS10-046 (Shell LNK / Shortcut)
- MS10-061 (Print Spooler Service)
- MS10-073 (Win32K Keyboard Layout)
- **MS08-067 (NetPathCanonicalize()), (Patched)**
  http://www.phreedom.org/blog/2008/decompiling-ms08-067/
- MS10-092 (Task Scheduler)
- CVE-2010-2772 (Siemens SIMATIC Static Password)

MS08-067 (NetPathCanonicalize())

- Known, patched (recent) vulnerability that allowed you to drop a payload and schedule it for execution

Flaws:

- Unpatched systems
- RPC flaw that allows unauthorized remote users to schedule tasks

Rootkits

- Goal: maintain control in secret
- Two stolen certificates:
  - Signs MrxCls.sys: launches Stuxnet on boot
  - Signs MRxNet.sys: hides Stuxnet filesystem objects and hooks new filesystem objects

PWN ALL THE THINGS

Hammer Time

Zero-Day* Vulnerabilities:

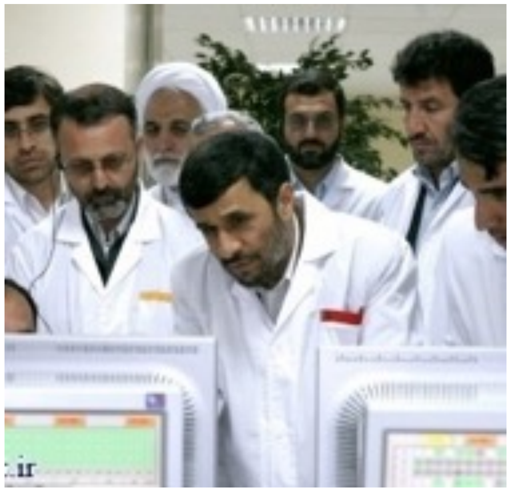
- MS10-046 (Shell LNK / Shortcut)
- MS10-061 (Print Spooler Service)
- MS10-073 (Win32K Keyboard Layout)
- MS08-067 (NetPathCanonicalize()), (Patched)
  http://www.phreedom.org/blog/2008/decompiling-ms08-067/
- MS10-092 (Task Scheduler)
- **CVE-2010-2772 (Siemens SIMATIC Static Password)**

When and Where?

- Stuxnet is targeted for the Natanz Nuclear Facility
  - Targets a configuration with six centrifuge cascades in a very specific configuration
  - Attacks specific controllers/hardware used at Natanz
  - Certainly had a test environment
- Where did the intelligence come from?

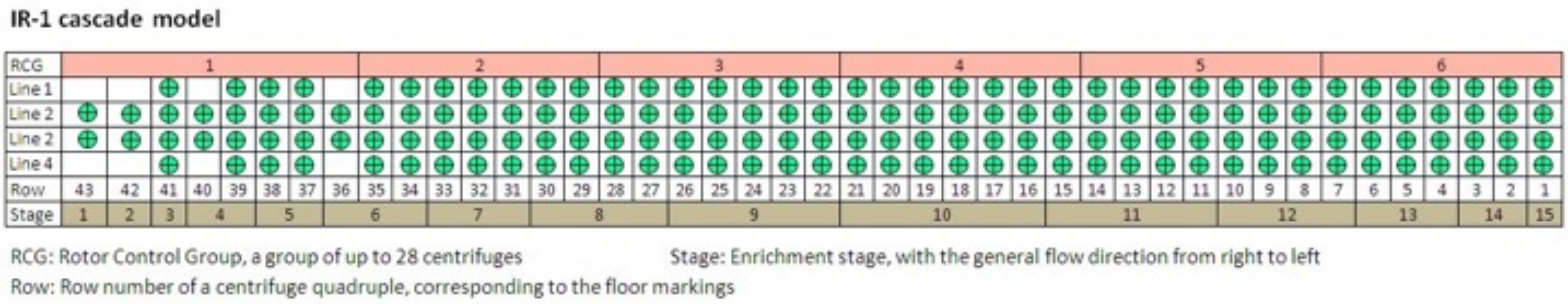
When and Where?

President Ahmadinejad's homepage! Here he is at Natanz. Wait, what's that on the screen?



When and Where?

Full resolution photos?? ENHANCE!



IR-1 cascade model

RCG	1								2							3							4							5							6						
Line 1			⊕		⊕	⊕	⊕		⊕	⊕	⊕	⊕	⊕	⊕	⊕	⊕	⊕	⊕	⊕	⊕	⊕	⊕	⊕	⊕	⊕	⊕	⊕	⊕	⊕	⊕	⊕	⊕	⊕	⊕	⊕	⊕	⊕	⊕	⊕	⊕	⊕	⊕	⊕	⊕
Line 2	⊕	⊕	⊕	⊕	⊕	⊕	⊕	⊕	⊕	⊕	⊕	⊕	⊕	⊕	⊕	⊕	⊕	⊕	⊕	⊕	⊕	⊕	⊕	⊕	⊕	⊕	⊕	⊕	⊕	⊕	⊕	⊕	⊕	⊕	⊕	⊕	⊕	⊕	⊕	⊕	⊕	⊕	⊕	⊕
Line 2	⊕	⊕	⊕	⊕	⊕	⊕	⊕	⊕	⊕	⊕	⊕	⊕	⊕	⊕	⊕	⊕	⊕	⊕	⊕	⊕	⊕	⊕	⊕	⊕	⊕	⊕	⊕	⊕	⊕	⊕	⊕	⊕	⊕	⊕	⊕	⊕	⊕	⊕	⊕	⊕	⊕	⊕	⊕	⊕
Line 4			⊕		⊕	⊕	⊕		⊕	⊕	⊕	⊕	⊕	⊕	⊕	⊕	⊕	⊕	⊕	⊕	⊕	⊕	⊕	⊕	⊕	⊕	⊕	⊕	⊕	⊕	⊕	⊕	⊕	⊕	⊕	⊕	⊕	⊕	⊕	⊕	⊕	⊕	⊕	⊕
Row	43	42	41	40	39	38	37	36	35	34	33	32	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1
Stage	1	2	3	4		5		6			7			8				9					10						11					12				13			14		15

RCG: Rotor Control Group, a group of up to 28 centrifuges

Stage: Enrichment stage, with the general flow direction from right to left

Row: Row number of a centrifuge quadruple, corresponding to the floor markings

When and Where?

Don’t get too ‘Merica on me, we do it too…

CVE-2010-2772 (Static Password)

- Siemens' controllers for centrifuges run WinCC
- WinCC SQL database servers
  - Connect using a hardcoded password
  - Loads Stuxnet as binary into a table
  - Executes binary as a stored procedure

CVE-2010-2772 (Static Password)

- Step7 DLL is renamed and replaced with an attack DLL
- If the PLC matches the desired profile, it's infected
- Breaks centrifuges by spinning them in weird ways while reporting everything is fine

Stuxnet: Fun Facts

- Black Market value of these vulns… probably millions
- Probably set back Iran's nuclear program by years
- Stolen code signing certificates actually signed the virus to make it look legitimate
- Virus phoned command and control centers to gather data, update, and presumably limit the scope of infection
- Whodunit?
- Learn more:
  - http://www.youtube.com/watch?v=rOwMW6agpTI
  - http://go.eset.com/us/resources/white-papers/Stuxnet_Under_the_Microscope.pdf
  - http://www.symantec.com/content/en/us/enterprise/media/security_response/whitepapers/w32_stuxnet_dossier.pdf
  - http://www.digitalbond.com/2012/01/31/langners-stuxnet-deep-dive-s4-video/
  - https://www.youtube.com/watch?v=rsXe2Gr2e3Q

You're too young to get this reference

Flame (Stuxnet's Cousin)

- Spyware
- Does crazy things like:
  - Get all the GPS tags from all your photos
  - Get your contact list from any Bluetooth attached phone
  - Screenshots, keystroke logging, audio recording

MD5 is Broken (an Interlude)

- MD5 is broken because you can find collisions
- Specifically, chosen-prefix collision
- Demonstrated to be feasible in 2008 to generate a rogue CA (http://marc-stevens.nl/research/papers/CR09-SSALMOdW.pdf)
- Attack required 3 days running on 215 PS3s to find a collision
- Everyone panics, CAs stop using MD5 entirely

Flame (Stuxnet's Cousin)

- Microsoft forgot about one Microsoft Terminal Server still issuing MD5 certificates
- Attackers devised a new way to find MD5 collisions
- Harder challenges, 1 ms time window to get the right timestamp
- Created an arbitrary MS root certificate for signing anything

Flame (Stuxnet's Cousin)

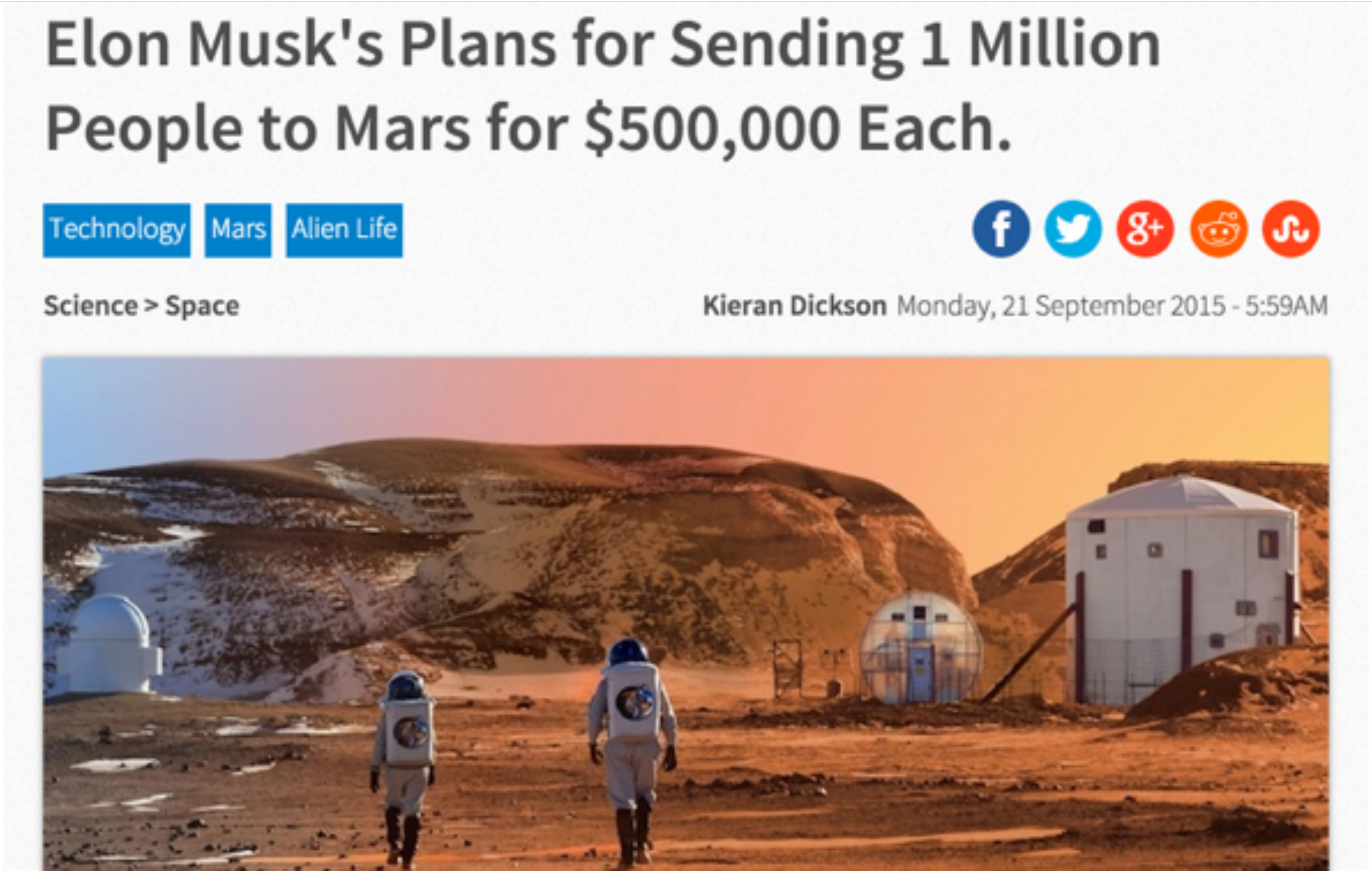
- Microsoft forgot about one Microsoft Terminal Server still issuing MD5 certificates
- Attackers devised a new way to find MD5 collisions
- Harder challenges, 1 ms time window to get the right timestamp
- Created an arbitrary MS root certificate for signing anything
- …. Like Windows Updates

Flame (Stuxnet's Cousin)

- "Oh Hai! I'm a Windows Update server!"
- "Oh Hello, I need an update."
- "Here, have delicious delicious Flame!"
- "You silly goose, this is signed by MS! I'll install it!"

airbnb

Recent history


Elon Musk's Plans for Sending 1 Million People to Mars for $500,000 Each.

Technology Mars Alien Life

Science > Space

Kieran Dickson Monday, 21 September 2015 - 5:59AM

US Government Data Hack

Former NSA Director Michael Hayden says much the same thing:

> If Hayden had had the ability to get the equivalent Chinese records when running CIA or NSA, he says, "I would not have thought twice. I would not have asked permission. I'd have launched the star fleet. And we'd have brought those suckers home at the speed of light." The episode, he says, "is not shame on China. This is shame on us for not protecting that kind of information." The episode is "a tremendously big deal, and my deepest emotion is embarrassment."

(Excerpt from Bruce Schneier's CryptoGram)

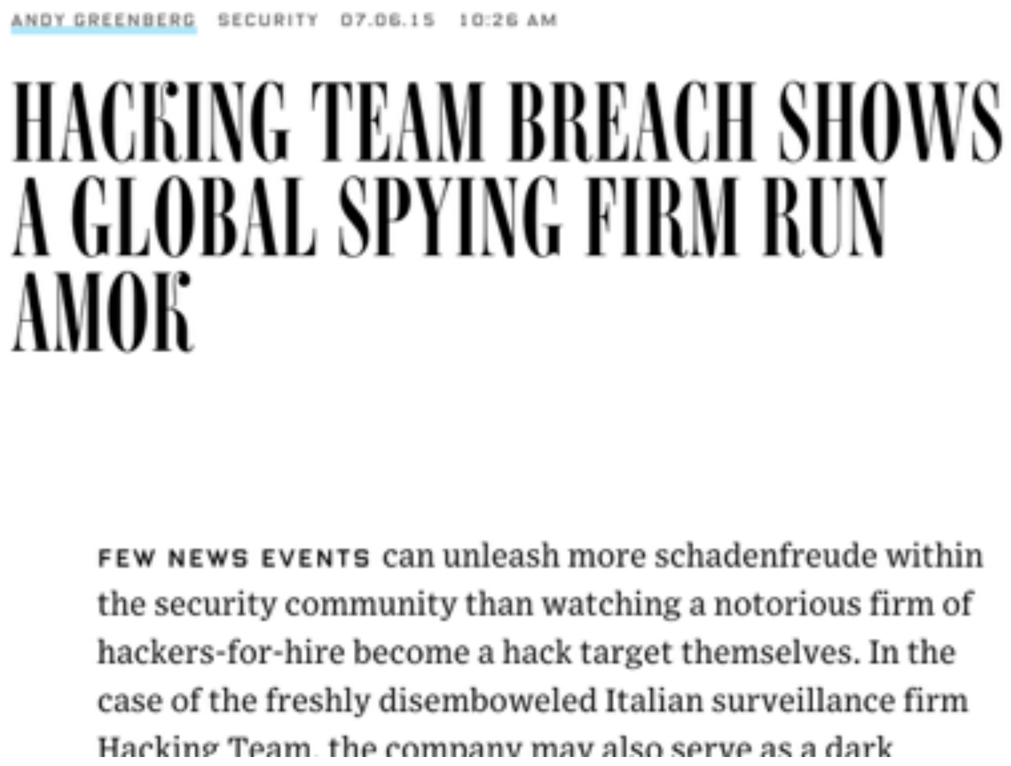
TECH SECURITY

SIM Card Company Says the NSA Probably Hacked It

Dan Kedmey | Feb. 25, 2015

But it denies the NSA got access to billions of people's mobile communications

One of the world's largest manufacturers of SIM cards has acknowledged evidence of security agency attacks on the company's internal networks, but it's denying that American and British intelligence agents



ANDY GREENBERG SECURITY 07.06.15 10:26 AM

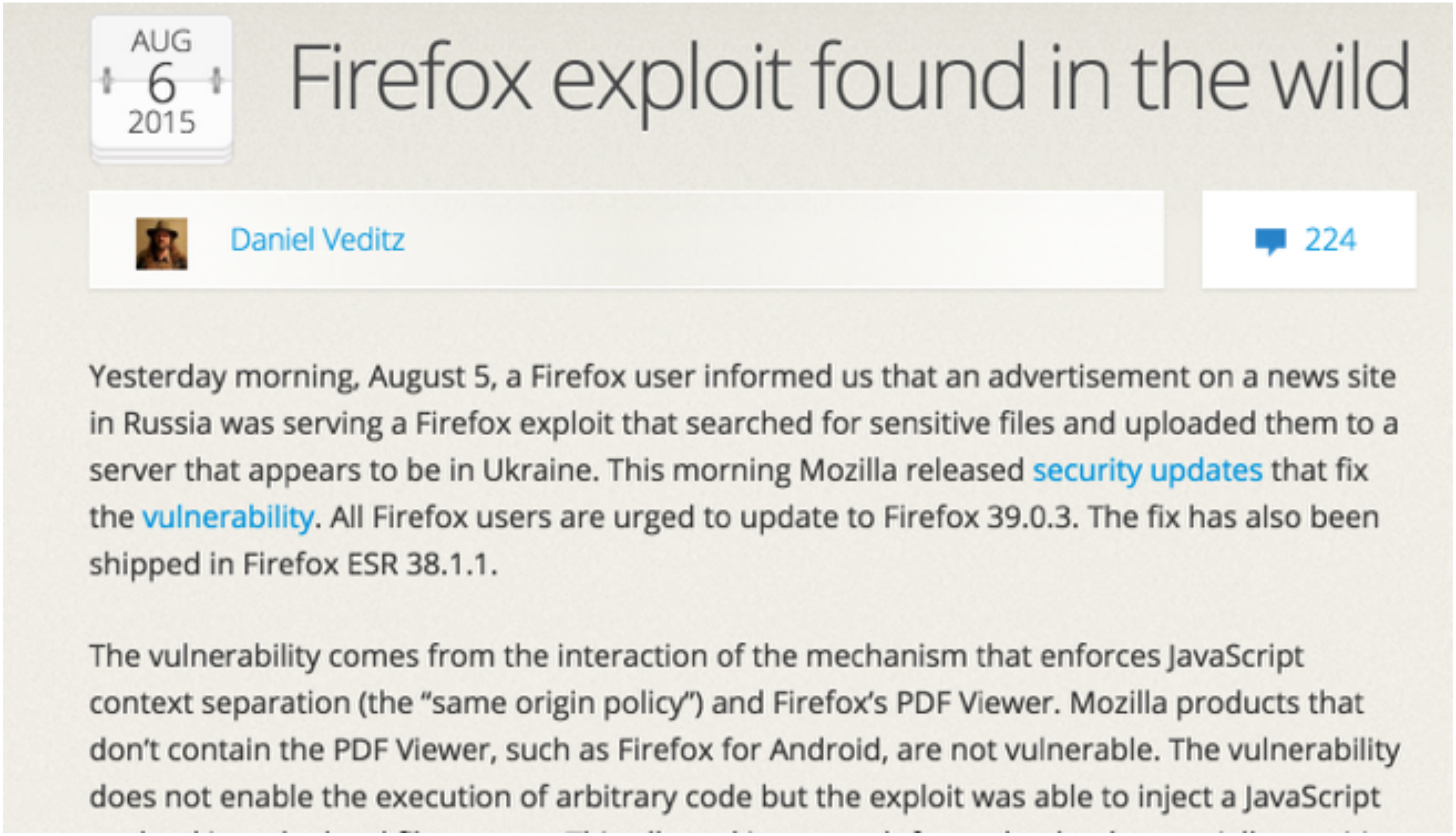
HACKING TEAM BREACH SHOWS A GLOBAL SPYING FIRM RUN AMOK

FEW NEWS EVENTS can unleash more schadenfreude within the security community than watching a notorious firm of hackers-for-hire become a hack target themselves. In the case of the freshly disemboweled Italian surveillance firm Hacking Team, the company may also serve as a dark

Fun facts about Hacking Team

- Sold vulnerabilities to people with money (sometimes oppressive regimes)
- They were hacked
- ~half-dozen flash 0-days found and other vulns

Firefox



AUG
6
2015

Firefox exploit found in the wild

Daniel Veditz

224

Yesterday morning, August 5, a Firefox user informed us that an advertisement on a news site in Russia was serving a Firefox exploit that searched for sensitive files and uploaded them to a server that appears to be in Ukraine. This morning Mozilla released security updates that fix the vulnerability. All Firefox users are urged to update to Firefox 39.0.3. The fix has also been shipped in Firefox ESR 38.1.1.

The vulnerability comes from the interaction of the mechanism that enforces JavaScript context separation (the "same origin policy") and Firefox's PDF Viewer. Mozilla products that don't contain the PDF Viewer, such as Firefox for Android, are not vulnerable. The vulnerability does not enable the execution of arbitrary code but the exploit was able to inject a JavaScript

Firefox

- Nomnoms, gimme your SSH keys

Stagefright (July/August)

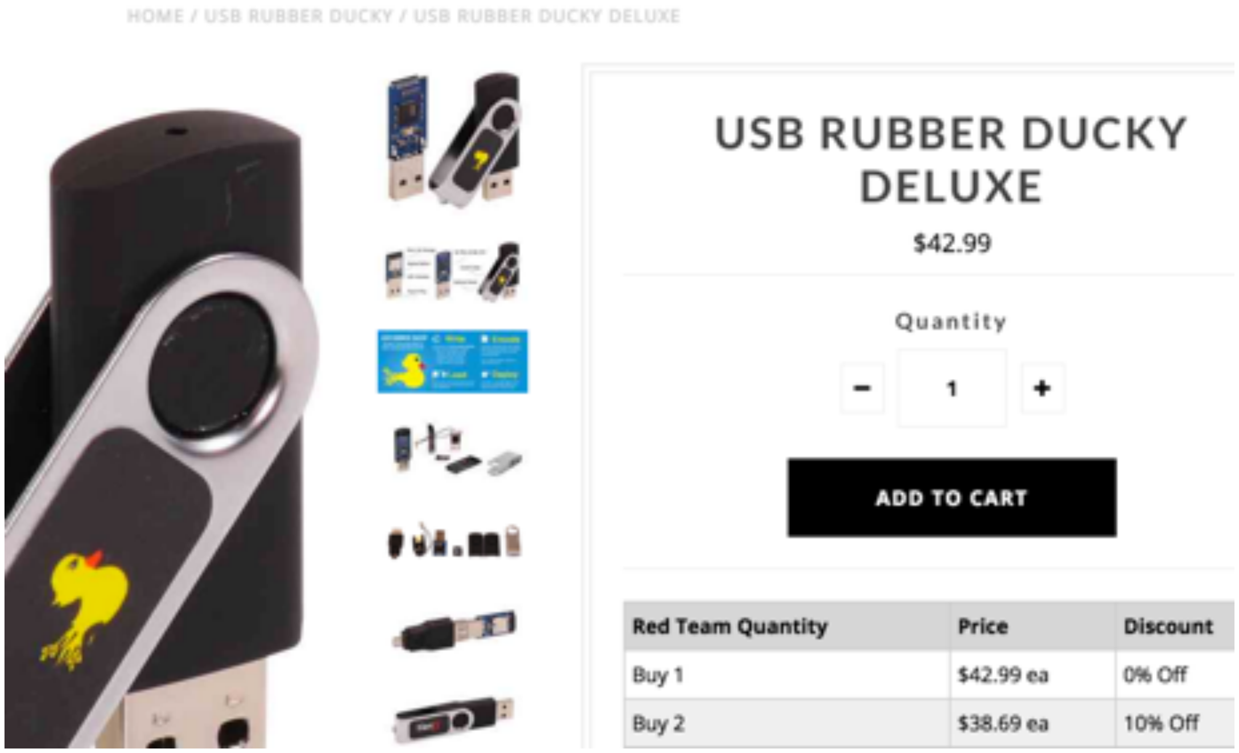
- Vulnerability in Android
- Multi-media library used by everything
- Fuzzing doesn’t look good
- Full RCE (on old versions of Android)

Stagefright (July/August)

- Vulnerability in Android
- Multi-media library used by everything
- Fuzzing doesn’t look good
- Full RCE (on old versions of Android)
- Mitigated (partially?) by ASLR

USB attacks (timeless classic)

- **RD:** "What up! I'm a keyboard"
- **Your laptop:** "Hi keyboard, I'm a computer, let me know what human wants to do."



HOME / USB RUBBER DUCKY / USB RUBBER DUCKY DELUXE

USB RUBBER DUCKY DELUXE

$42.99

Quantity

– 1 +

ADD TO CART

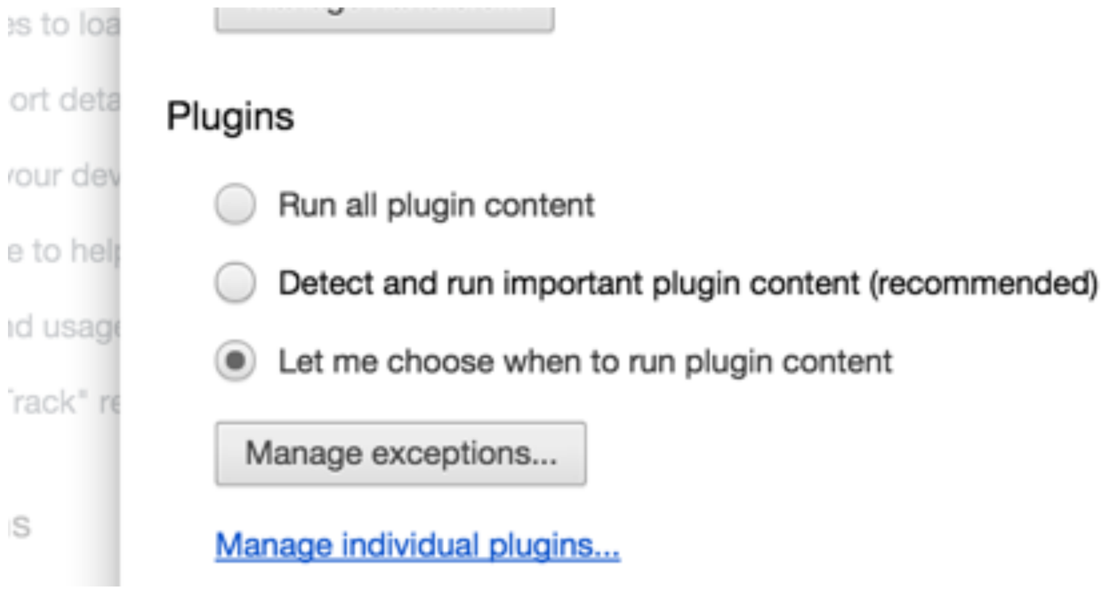
Red Team Quantity	Price	Discount
Buy 1	$42.99 ea	0% Off
Buy 2	$38.69 ea	10% Off

USB attacks (timeless classic)

- **RD:** "What up! I'm a keyboard"
- **Your laptop:** "Hi keyboard, I'm a computer, let me know what human wants to do."
- **RD:** "Human seems to be installing some malware."
- **Your laptop:** "Who am I to question human?!"

Chrome extensions

- Chrome is dope. Use Chrome and enable "click to play"
- You are now safer than 95%
- Use a password manager, you're now safer than 99.9%



Plugins

( ) Run all plugin content

( ) Detect and run important plugin content (recommended)

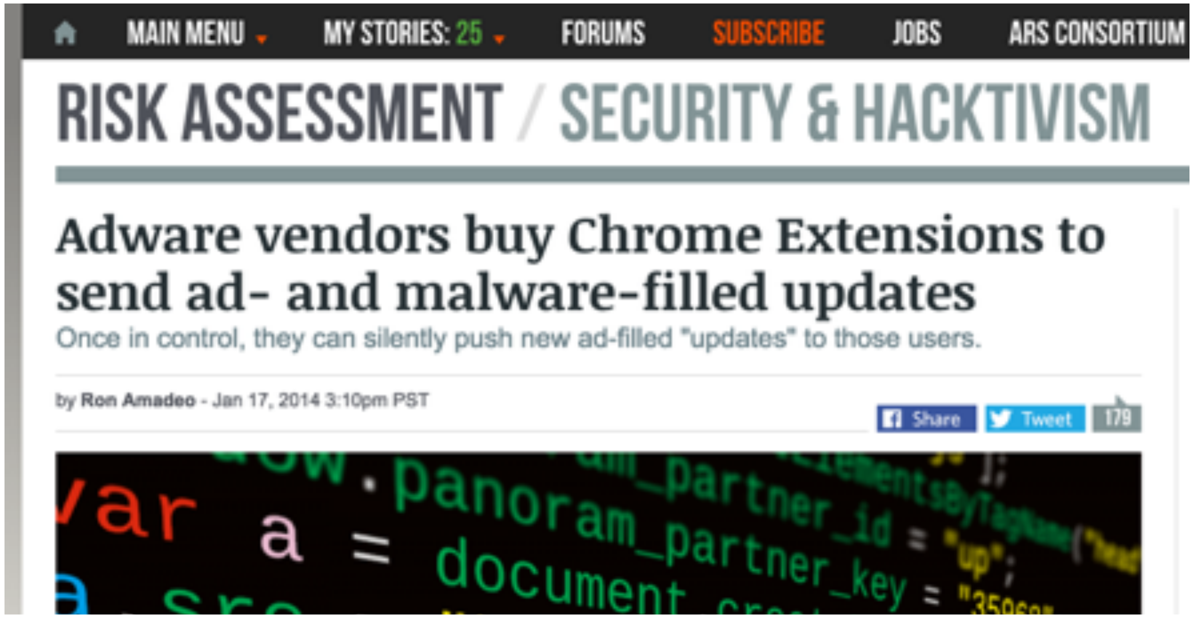
(●) Let me choose when to run plugin content

Manage exceptions...

Manage individual plugins...

Chrome extensions

- Chrome is dope. Use Chrome and enable "click to play"
- You are now safer than 95%
- Use a password manager, you're now safer than 99.9%
- Installing random extensions ruins all of the above



RISK ASSESSMENT / SECURITY & HACKTIVISM

Adware vendors buy Chrome Extensions to send ad- and malware-filled updates

Once in control, they can silently push new ad-filled "updates" to those users.

by Ron Amadeo - Jan 17, 2014 3:10pm PST

Xcode Ghost

airbnb

Apple scrambles after 40 malicious “XcodeGhost” apps haunt App Store

Outbreak may have caused hundreds of millions of people to download malicious apps.

by **Dan Goodin** - Sep 21, 2015 7:40am PDT

Share Tweet 234

Xcode Ghost

- Malicious version of Xcode
- Automatically injects malware into builds
- Used by real dev of real popular apps
- Lots of deployment in China (why?)
- Phishing capability and collects basic device data

I Love Security, What's Next?

- Ethics in security
- Possible Careers

6'6"
6'0"
5'6"
5'0"
4'6"
4'0"
3'6"
3'0"
arts-wallpapers.com
- Not for Sale
do not for sale this desktop work image, only be used for your personal use only. desktop work (c) arts-wallpapers.com

Ethics in Security

- Big ethical debates used to be:

Responsible vs Full Disclosure

Ethics in Security

- Big ethical debates used to be:

  Responsible vs Full Disclosure

- Debate has shifted to:

  Disclosure vs Selling Weapons

Careers in Security

- Shape your job around your ethical standpoint, not vice versa

Careers in Security

- Shape your job around your ethical standpoint, not vice versa
- Write security relevant software

Careers in Security

- Shape your job around your ethical standpoint, not vice versa
- Write security relevant software
- Write (more) secure software

Careers in Security

- Shape your job around your ethical standpoint, not vice versa
- Write security relevant software
- Write (more) secure software
- Be a criminal

Careers in Security

- Shape your job around your ethical standpoint, not vice versa
- Write security relevant software
- Write (more) secure software
- Be a criminal
- Academia

Careers in Security

- Shape your job around your ethical standpoint, not vice versa
- Write security relevant software
- Write (more) secure software
- Be a criminal
- Academia
- Independent researcher

Careers in Security

- Shape your job around your ethical standpoint, not vice versa
- Write security relevant software
- Write (more) secure software
- Be a criminal
- Academia
- Independent researcher
- Pen testing

Careers in Security

- Shape your job around your ethical standpoint, not vice versa
- Write security relevant software
- Write (more) secure software
- Be a criminal
- Academia
- Independent researcher
- Pen testing
- Help me fight bad guys

Who looks for security bugs?

- Criminals
- Security Researchers
- Pen Testers
- Governments
- Hacktivists
- Academics
- Defenders

Criminals

- Goals:
  - Money (botnets, CC#s, blackmail)
  - Stay out of jail
- Thoroughness:
  - Reliable exploits
  - Don't need 0-days (but they sure are nice)
- Access:
  - Money
  - Blackbox testing

Security Researchers

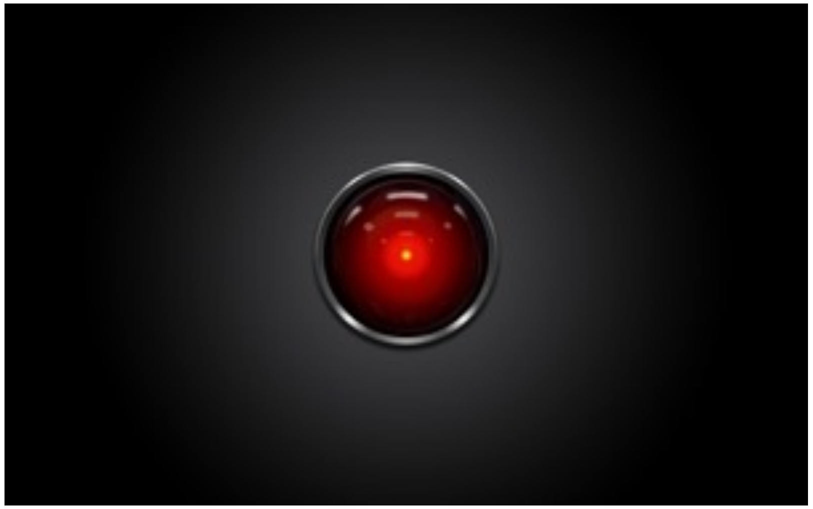
- Goals:
  - Column inches from press, props from friends
  - Preferably in a trendy platform
- Thoroughness:
  - Don’t need to be perfect, don’t want to be embarrassed
- Access:
  - Casual access to engineers
  - Source == Lawyers

Pen Testers

- Goals:
  - Making clients and users safer
  - Finding vulns criminals would use
- Thoroughness:
  - Need coverage
  - Find low-hanging fruit
  - Find high impact vulnerabilities
  - Don't fix or fully exploit
- Access:
  - Access to Engineers
  - Access to Source
  - Permission

Governments

- Goals:
  - Attack/espionage
  - Defend
- Thoroughness:
  - Reliable exploits
- Access:
  - Money
  - Talent
  - Time



Hacktivists

- Goals:
  - Doing something "good"
  - Stay out of jail
- Thoroughness:
  - Reliable exploits
  - Don't need 0-days
- Access:
  - Talent
  - Plentiful targets

Academics

- Goals:
  - Finding common flaws and other general problems
  - Developing new crypto
  - Make something cool and useful
  - Make everyone safer
- Thoroughness:
  - Depth in area of research
- Access:
  - Creating new things
  - Blackbox

Techniques

- With access:
  - Source code review
  - Engineer interviews
  - Testing in a controlled environment
- Without access:
  - Blackbox testing
  - Fuzzing (give weird inputs, see what happens)
  - Reverse Engineering
  - Social Engineering

Ethics in Security

- A single iOS 0-day sold for a purported 500k
- Most profitable way to be a hacker is likely to sell exploits
- Be afraid, be very afraid (tin foil available up front)
- But remember, there are many ways to make money by being unethical, you still shouldn't do it

airbnb

Defense wins championships

Landscape

- People are selling 0-day vulnerabilities all over the place
- Companies are relying on unaudited open source software incredibly ubiquitously
- Bad things happen all the time
- Users are prone to error and phishing
- People love clicking on everything, or plugging in USBs

Landscape

- Security must work within this landscape

Modern security: Perimeter

- Spend some energy here:
  - Patch your stuff fast when security issues come up
  - Sprinkle 2FA on everything you care about
  - Don’t get taken down by DDoS ransom artists, be ready.
  - Train your employees
  - Basic controls: review, static analysis, run your own checks of your security like an attacker would
  - Push Chrome usage/password managers/click to play

Modern security: small incidents

- Make incidents small:
  - Did I mention 2FA on everything? Yeah, everything
  - Encrypt your sensitive data and separate the key from the data (transparent data encryption isn't good)
  - Limit access to a minimal set
  - Audit and monitor access for anomalies
  - Backup your stuff (coder space)
  - Segment your stuff

Modern security: detect/stop incidents

- Detect and stop
  - Know what's normal.
  - Instrument every host as well as your network
  - Invest in tuning signal so you know what looks weird
  - Have a plan
  - Look for indicators of compromise and stop them:
    - Persistence
    - Command/Control
    - Lateral movement

Roles in defense

- Security engineers: know a lot about security. Do pen testing, design review, and bring security to the rest of the employees.

Roles in defense

- Security engineers: know a lot about security. Do pen testing, design review, and bring security to the rest of the employees.
- Software engineers: write security software. Work on encrypting our sensitive data more security, improving access controls/auditing, making it easier to write secure software.

Roles in defense

- Security engineers: know a lot about security. Do pen testing, design review, and bring security to the rest of the employees.
- Software engineers: write security software. Work on encrypting our sensitive data more security, improving access controls/auditing, making it easier to write secure software.
- Intrusion Detection specialist: create signals that look for indicators of compromise (IOC) across all laptops, hosts, and internal equipment.

Roles in defense

- Security engineers: know a lot about security. Do pen testing, design review, and bring security to the rest of the employees.
- Software engineers: write security software. Work on encrypting our sensitive data more security, improving access controls/auditing, making it easier to write secure software.
- Intrusion Detection specialist: create signals that look for indicators of compromise (IOC) across all laptops, hosts, and internal equipment.
- Analysts: Front-lines of security, look at signal from IDS folks, tune rules, and investigate issues.

Roles in defense

- Incident Response team: the "oh sh*t" crew. Something looks pretty bad, don't know what's going on. Call these people.
- Tools: security is about customization, create custom security tooling and open source it.
- Network security engineers: design better network layout and segmentation.
- Security operations: manage users, operationalize security.
- Internal policy folks.
- Collaborate with everyone in the organization. Give security a try!

Thanks!

- paul.youn@airbnb.com
- Career fair booth! Be there.
- Go to a small company:
  - Startup like Airbnb
  - NCC Group pen testing
- Go Dubs!

References

- Help from NCC Group on previous versions of slides.
- References:

http://www.babylifestyles.com/images/blog/2009/05/stork.gif
http://cdn3.mixrmedia.com/wp-uploads/wirebot/blog/2010/01/jacked_in.jpg
http://www.dan-dare.org/FreeFun/Images/CartoonsMoviesTV/BugsLifeWallpaper800.jpg
http://cdn.tss.uproxx.com/TSS/wp-content/uploads/2008/03/ep60_mcnultybunk_506_03.jpg
http://desertpeace.files.wordpress.com/2010/11/spy-vs-spy.jpg
http://upload.wikimedia.org/wikipedia/commons/thumb/d/d7/Don_Knotts_Jim_Nabors_Andy_Griffith_Show_1964.JPG/220px-Don_Knotts_Jim_Nabors_Andy_Griffith_Show_1964.JPG
http://worldofstuart.excellentcontent.com/bruceworld/pics/depp-pirate.jpg
http://keetsa.com/blog/wp-content/uploads/2007/09/nuclear_explosion.jpg
http://www.asianbite.com/photos/psy-gangnam-style_27980.jpg
http://upload.wikimedia.org/wikipedia/commons/d/d3/Cbc_encryption.png
http://www.neatorama.com/wp-content/uploads/2010/11/bugs-bunnyreclining-499x367.jpg
http://www.langner.com/en/wp-content/uploads/2011/12/IR-1-cascade-model1.jpg
http://bdnpull.bangorpublishing.netdna-cdn.com/wp-content/uploads/2012/06/Natanz_Ahmadinejad-Visit_4-computers-250x241.jpg
http://www.politico.com/blogs/bensmith/0509/Secret_CIA_document_on_White_House_Flickr_feed.html
http://www.sirlin.net/storage/street_fighter/dhalsim_yoga_flame.gif?__SQUARESPACE_CACHEVERSION=1226558938179
http://www.cosmosmagazine.com/files/imagecache/feature/files/20080314_sherlock_holmes.jpg
http://www.inquisitr.com/wp-content/2012/08/original3-e1346095350417.jpg
http://www-bgr-com.vimg.net/wp-content/uploads/2011/06/lulzsec-hackers110624115314.jpg
http://img.timeinc.net/time/photoessays/2009/blame_25/blame_25_madoff.jpg
http://www.imgbase.info/images/safe-wallpapers/miscellaneous/1_other_wallpapers/16562_1_other_wallpapers_hal_9000.jpg
http://www.thecfpgroup.com/images/engineers.gif
http://www.moviefanatic.com/gallery/ryan-gosling-in-drive/
http://www.allmovieposter.org/poster/the-usual-suspects-poster-15.jpg
emisportscentral.com
http://letsgowarriors.com
http://www.waririorsworld.net
http://arstechnica.com/security/2015/09/apple-scrambles-after-40-malicious-xcodeghost-apps-haunt-app-store/
http://arstechnica.com/security/2014/01/malware-vendors-buy-chrome-extensions-to-send-adware-filled-updates/
http://hakshop.myshopify.com/products/usb-rubber-ducky-deluxe?variant=353378649
https://blog.mozilla.org/security/2015/08/06/firefox-exploit-found-in-the-wild/
http://www.wired.com/2015/07/hacking-team-breach-shows-global-spying-firm-run-amok/
http://time.com/3722150/nsa-sim-cards/
http://www.outerplaces.com/science/item/9914-elon-musk-s-plans-for-sending-1-million-people-to-mars-for-500-000-each