The Future of Security and Exploitation

Modern Binary Exploitation

CSCI 4968 - Spring 2015

Markus Gaasedelen

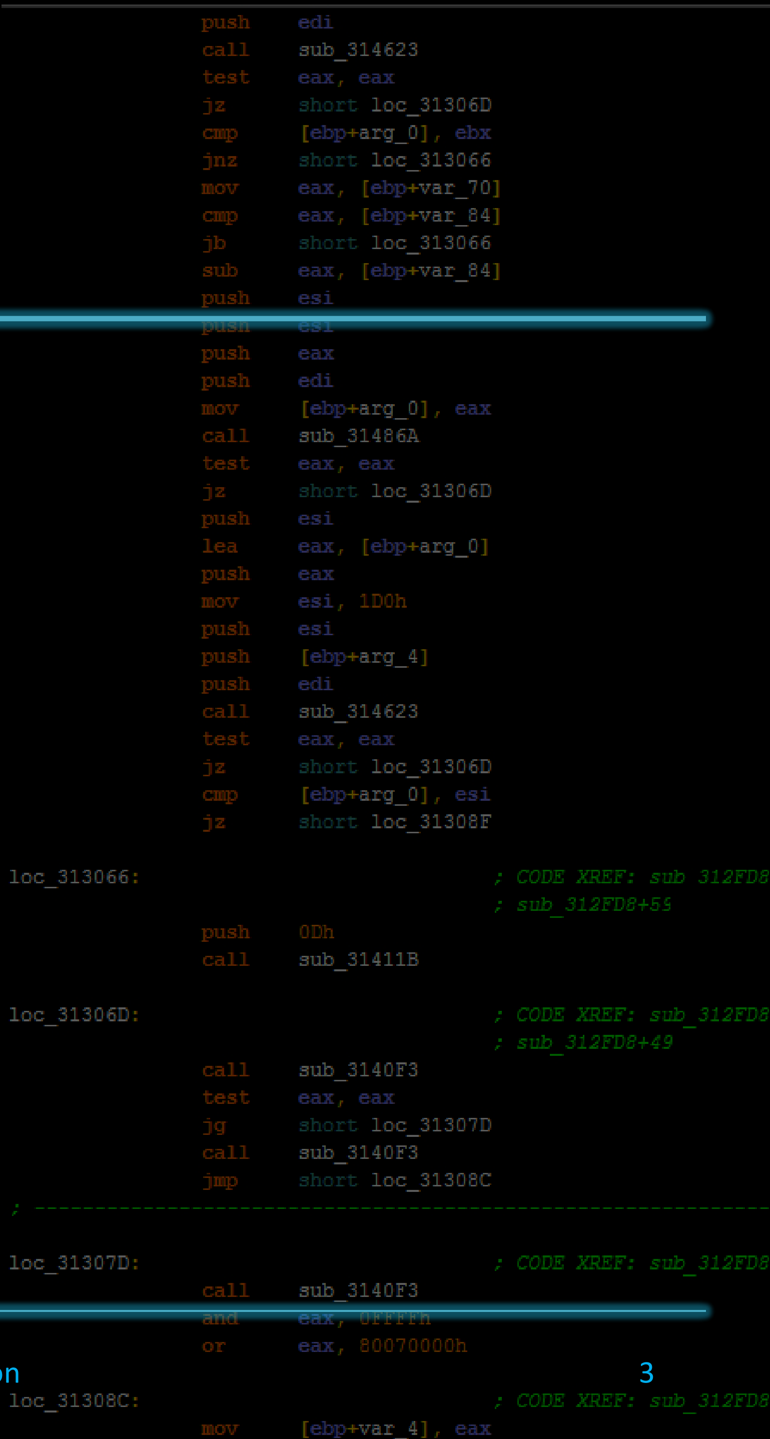
```
         push    edi
         call    sub_314623
         test    eax, eax
         jz      short loc_31306D
         cmp     [ebp+arg_0], ebx
         jnz     short loc_313066
         mov     eax, [ebp+var_70]
         cmp     eax, [ebp+var_84]
         jb      short loc_313066
         sub     eax, [ebp+var_84]
         push    esi
         push    esi
         push    eax
         push    edi
         mov     [ebp+arg_0], eax
         call    sub_31486A
         test    eax, eax
         jz      short loc_31306D
         push    esi
         lea     eax, [ebp+arg_0]
         push    eax
         mov     esi, 1D0h
         push    esi
         push    [ebp+arg_4]
         push    edi
         call    sub_314623
         test    eax, eax
         jz      short loc_31306D
         cmp     [ebp+arg_0], esi
         jz      short loc_31308F

loc_313066:                             ; CODE XREF: sub_312FD8
                                        ; sub_312FD8+55
         push    0Dh
         call    sub_31411B

loc_31306D:                             ; CODE XREF: sub_312FD8
                                        ; sub_312FD8+49
         call    sub_3140F3
         test    eax, eax
         jg      short loc_31307D
         call    sub_3140F3
         jmp     short loc_31308C
; ---------------------------------------------------------------

loc_31307D:                             ; CODE XREF: sub_312FD8
         call    sub_3140F3
         and     eax, 0FFFFh
         or      eax, 80070000h

loc_31308C:                             ; CODE XREF: sub_312FD8
         mov     [ebp+var_4], eax
```

DEFCON Quals

- May 15/16/17
  - Starts 8pm Friday, May 15th
  - Sage 3101 Friday, Sage 4101 Saturday/Sunday
- Extra Credit
  - Letter grade raise on a Lab
  - OR +10% on the final project
- To get the extra credit
  - Solve one challenge (that’s not a sanity check)
  - OR Play 10 hours on Saturday and/or Sunday

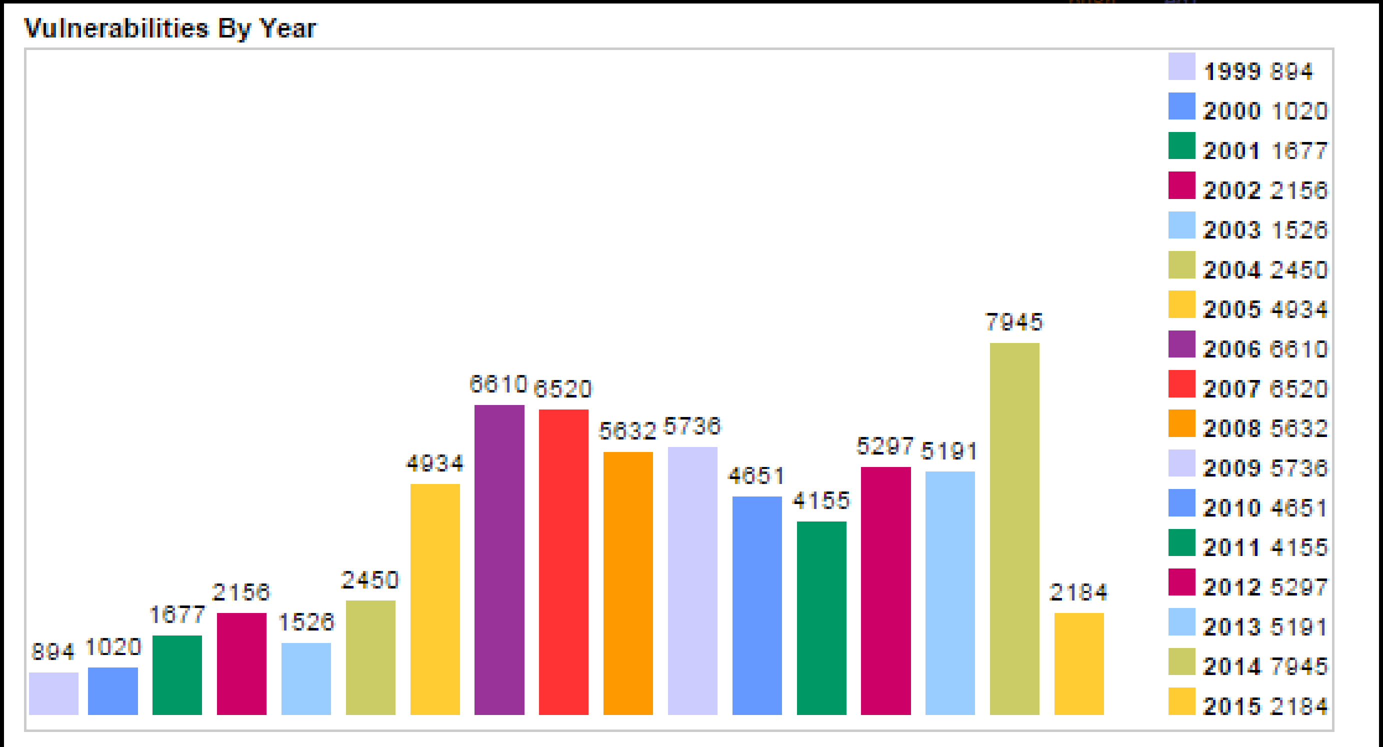
Lecture Overview

- Security
  - Security Today
  - Security Tomorrow
- Exploitation
  - Exploitation Today
  - Exploitation Tomorrow



```
                push    edi
                call    sub_314623
                test    eax, eax
                jz      short loc_31306D
                cmp     [ebp+arg_0], ebx
                jnz     short loc_313066
                mov     eax, [ebp+var_70]
                cmp     eax, [ebp+var_84]
                jb      short loc_313066
                sub     eax, [ebp+var_84]
                push    esi
                push    esi
                push    eax
                push    edi
                mov     [ebp+arg_0], eax
                call    sub_31486A
                test    eax, eax
                jz      short loc_31306D
                push    esi
                lea     eax, [ebp+arg_0]
                push    eax
                mov     esi, 1D0h
                push    esi
                push    [ebp+arg_4]
                push    edi
                call    sub_314623
                test    eax, eax
                jz      short loc_31306D
                cmp     [ebp+arg_0], esi
                jz      short loc_31308F

loc_313066:                             ; CODE XREF: sub_312FD8
                                        ; sub_312FD8+55
                push    0Dh
                call    sub_31411B

loc_31306D:                             ; CODE XREF: sub_312FD8
                                        ; sub_312FD8+49
                call    sub_3140F3
                test    eax, eax
                jg      short loc_31307D
                call    sub_3140F3
                jmp     short loc_31308C
; ---------------------------------------------------------------------------

loc_31307D:                             ; CODE XREF: sub_312FD8
                call    sub_3140F3
                and     eax, 0FFFFh
                or      eax, 80070000h

loc_31308C:                             ; CODE XREF: sub_312FD8
                mov     [ebp+var_4], eax
```

CVE Statistics – May 2015



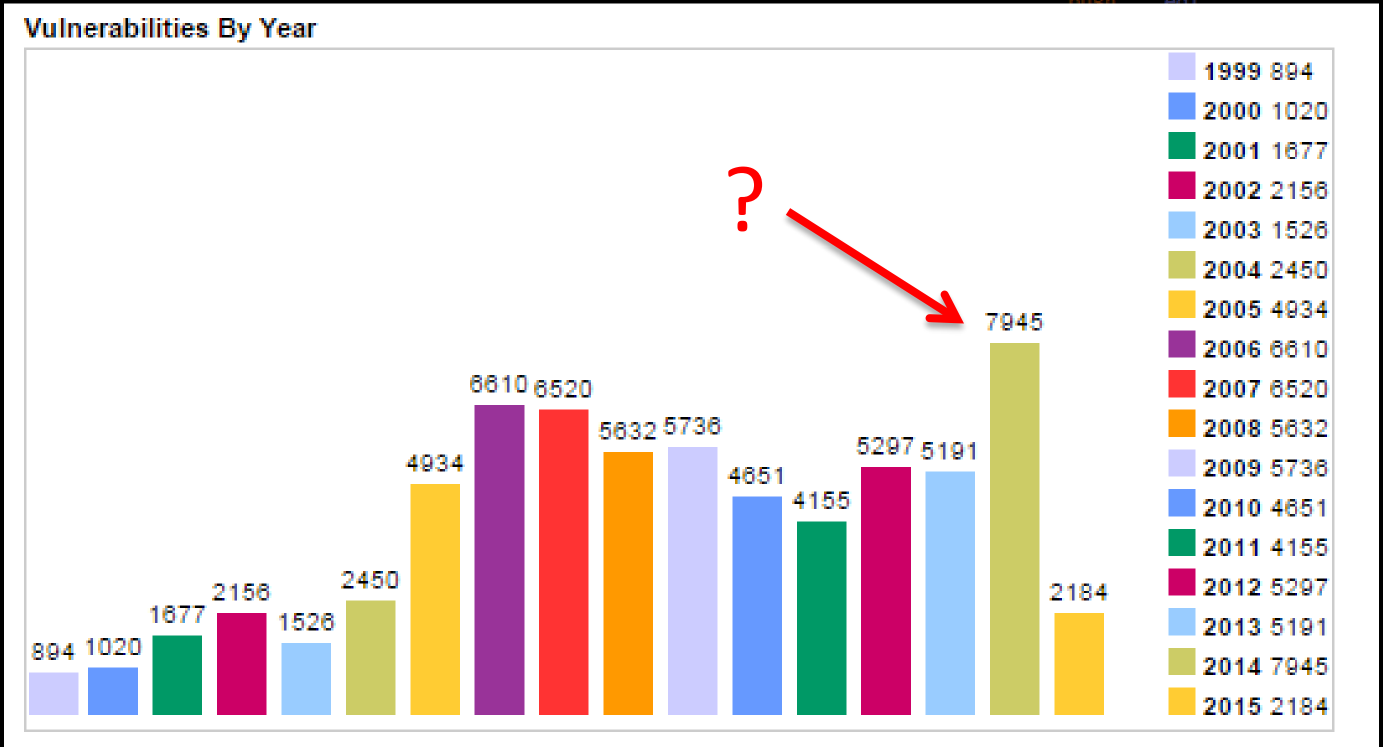
http://www.cvedetails.com/browse-by-date.php

Security Trends

- As you know, security and mitigation technologies are no doubt getting better
  – Why the spike in 2014?

```
                push    edi
                call    sub_314623
                test    eax, eax
                jz      short loc_31306D
                cmp     [ebp+arg_0], ebx
                jnz     short loc_313066
                mov     eax, [ebp+var_70]
                cmp     eax, [ebp+var_84]
                jb      short loc_313066
                sub     eax, [ebp+var_84]
                push    esi
                push    esi
                push    eax
                push    edi
                mov     [ebp+arg_0], eax
                call    sub_31486A
                test    eax, eax
                jz      short loc_31306D
                push    esi
                lea     eax, [ebp+arg_0]
                push    eax
                mov     esi, 1D0h
                push    esi
                push    [ebp+arg_4]
                push    edi
                call    sub_314623
                test    eax, eax
                jz      short loc_31306D
                cmp     [ebp+arg_0], esi
                jz      short loc_31308F

loc_313066:                             ; CODE XREF: sub_312FD8
                                        ; sub_312FD8+55
                push    0Dh
                call    sub_31411B

loc_31306D:                             ; CODE XREF: sub_312FD8
                                        ; sub_312FD8+49
                call    sub_3140F3
                test    eax, eax
                jg      short loc_31307D
                call    sub_3140F3
                jmp     short loc_31308C
; ---------------------------------------------------------------------------

loc_31307D:                             ; CODE XREF: sub_312FD8
                call    sub_3140F3
                and     eax, 0FFFFh
                or      eax, 80070000h

loc_31308C:                             ; CODE XREF: sub_312FD8
                mov     [ebp+var_4], eax
```

CVE Statistics – May 2015



http://www.cvedetails.com/browse-by-date.php

June 2013

```
push    edi
call    sub_314623
test    eax, eax
jz      short loc_31306D
cmp     [ebp+arg_0], ebx
jnz     short loc_313066
mov     eax, [ebp+var_70]
cmp     eax, [ebp+var_84]
jb      short loc_313066
sub     eax, [ebp+var_84]
push    esi
push    esi
push    eax
push    edi
```

```
XREF: sub_312FD8
12FD8+55

XREF: sub_312FD8
12FD8+49

---------------------

loc_31307D:                             ; CODE XREF: sub_312FD8
        call    sub_3140F3
        and     eax, 0FFFFh
        or      eax, 80070000h

loc_31308C:                             ; CODE XREF: sub_312FD8
        mov     [ebp+var_4], eax
```

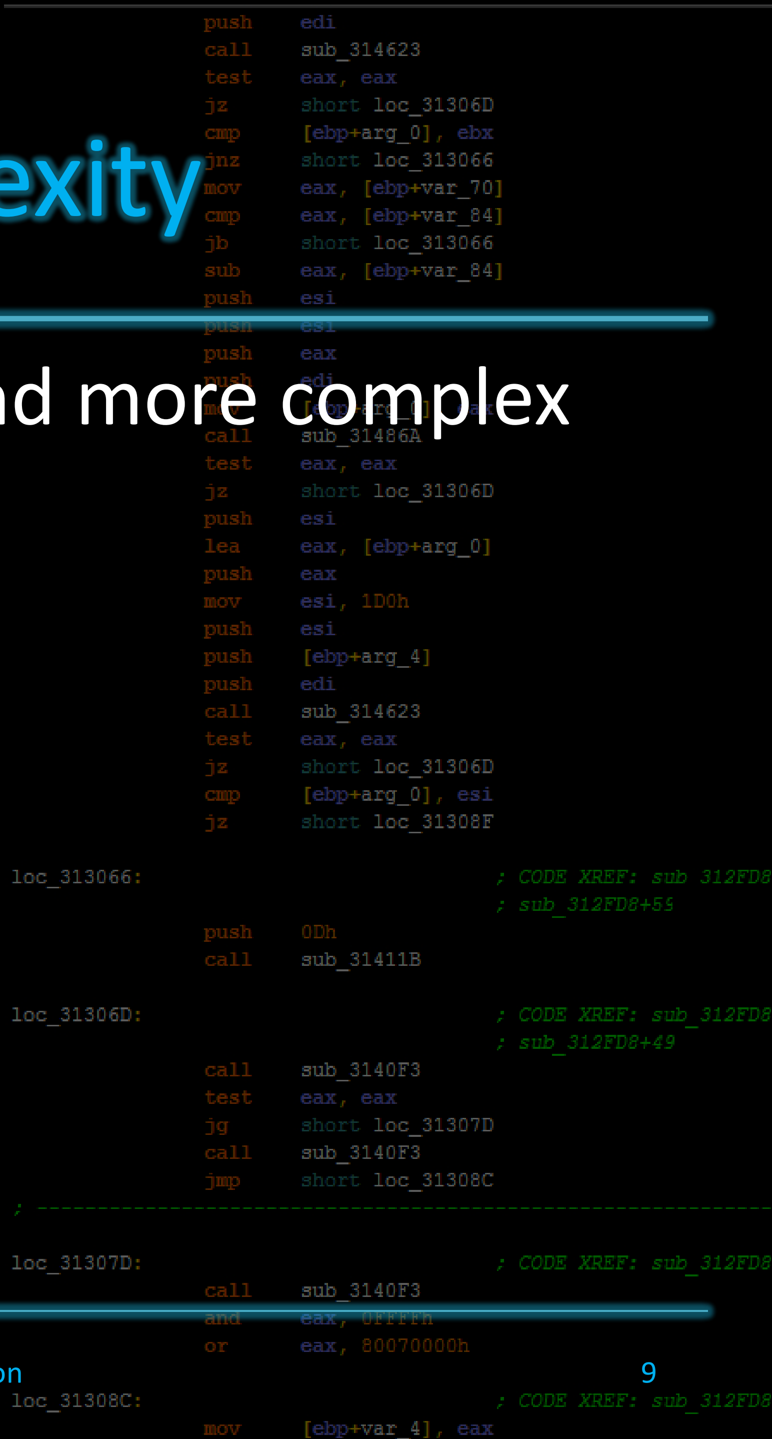
Security Trends

- As you know, security and mitigation technologies are no doubt getting better
  - Why the spike in 2014?

- Possibly a result of the Snowden revelations
  - The fallout raised global awareness and interest in security/privacy. ‘Cyber’ in the news ever since

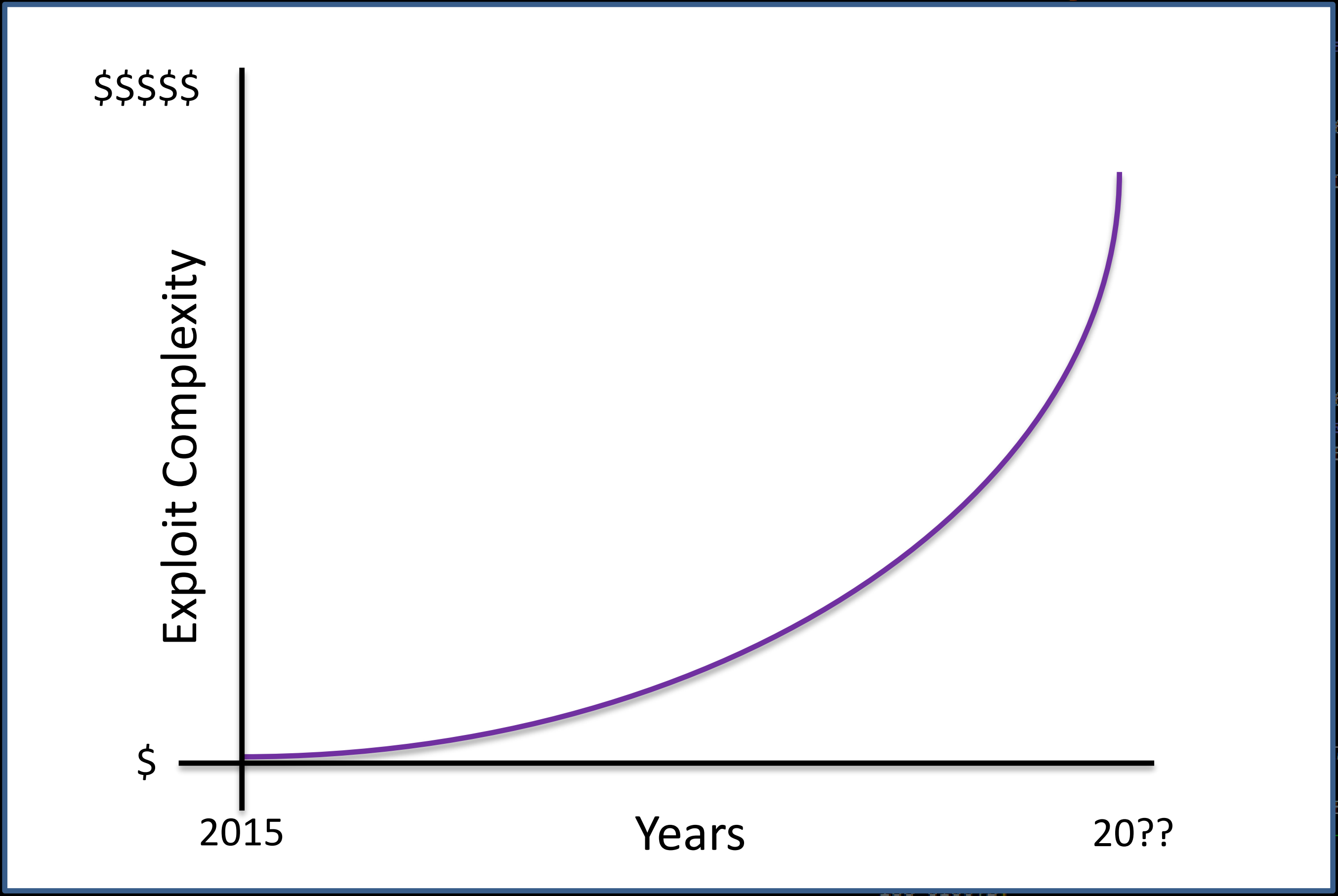
Unsustainable Complexity

- Exploits are getting more and more complex
  - More bugs
  - More time
  - More money



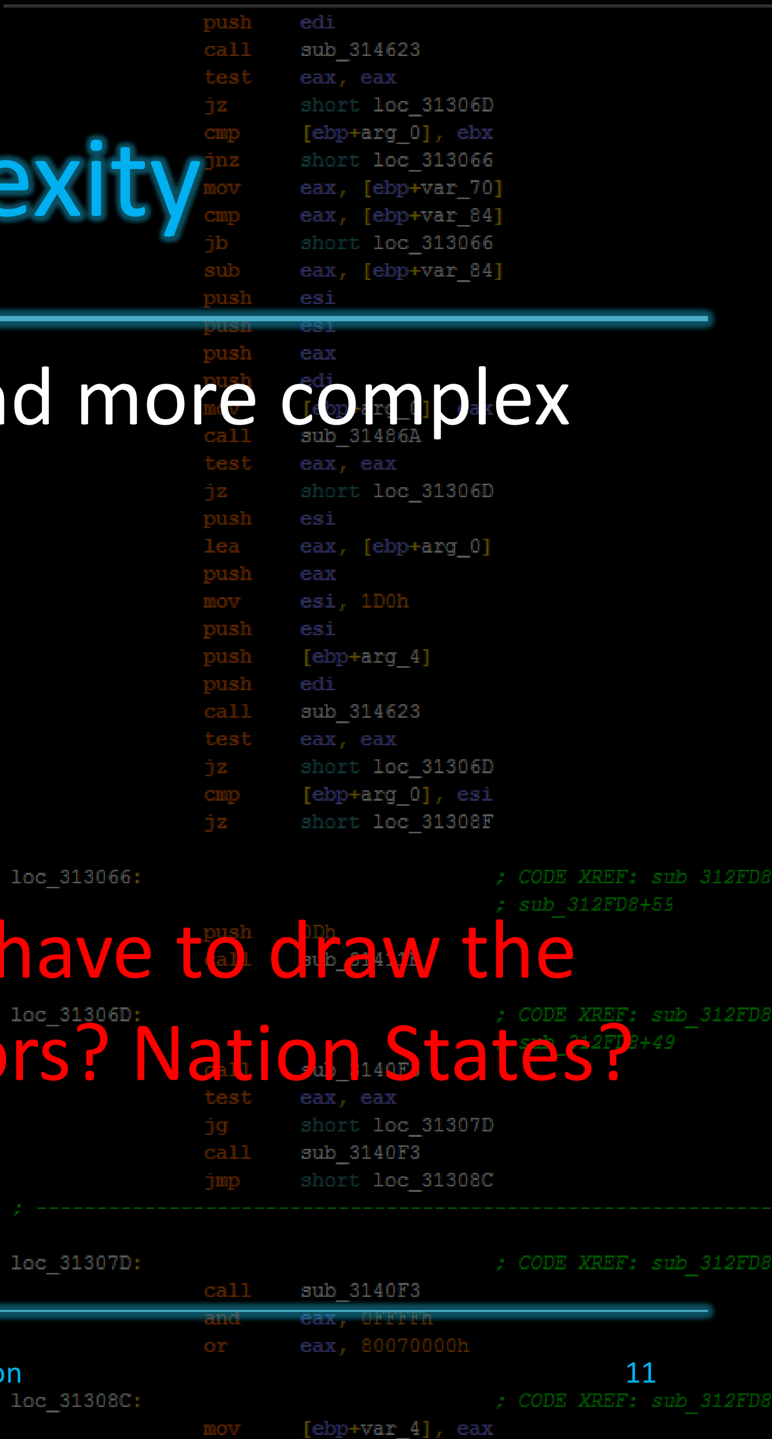
```
                push    edi
                call    sub_314623
                test    eax, eax
                jz      short loc_31306D
                cmp     [ebp+arg_0], ebx
                jnz     short loc_313066
                mov     eax, [ebp+var_70]
                cmp     eax, [ebp+var_84]
                jb      short loc_313066
                sub     eax, [ebp+var_84]
                push    esi
                push    esi
                push    eax
                push    edi
                mov     [ebp+arg_0], eax
                call    sub_31486A
                test    eax, eax
                jz      short loc_31306D
                push    esi
                lea     eax, [ebp+arg_0]
                push    eax
                mov     esi, 1D0h
                push    esi
                push    [ebp+arg_4]
                push    edi
                call    sub_314623
                test    eax, eax
                jz      short loc_31306D
                cmp     [ebp+arg_0], esi
                jz      short loc_31308F

loc_313066:                             ; CODE XREF: sub_312FD8
                                        ; sub_312FD8+55
                push    0Dh
                call    sub_31411B

loc_31306D:                             ; CODE XREF: sub_312FD8
                                        ; sub_312FD8+49
                call    sub_3140F3
                test    eax, eax
                jg      short loc_31307D
                call    sub_3140F3
                jmp     short loc_31308C
; ---------------------------------------------------------------

loc_31307D:                             ; CODE XREF: sub_312FD8
                call    sub_3140F3
                and     eax, 0FFFFh
                or      eax, 80070000h

loc_31308C:                             ; CODE XREF: sub_312FD8
                mov     [ebp+var_4], eax
```

Unsustainable Complexity



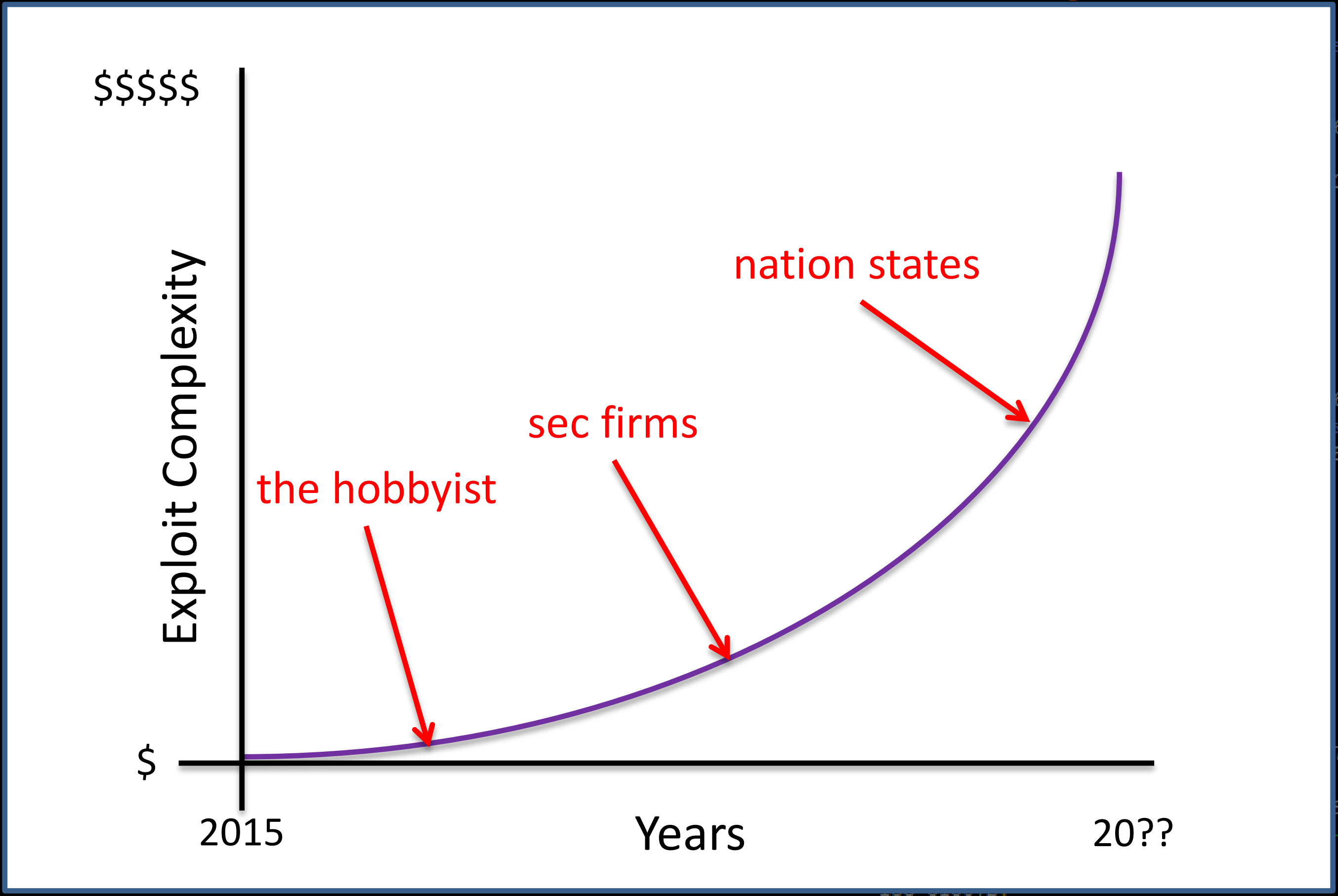
Unsustainable Complexity

- Exploits are getting more and more complex
  - More bugs
  - More time
  - More money
- At what point do hobbyists have to draw the line? Companies? Contractors? Nation States?



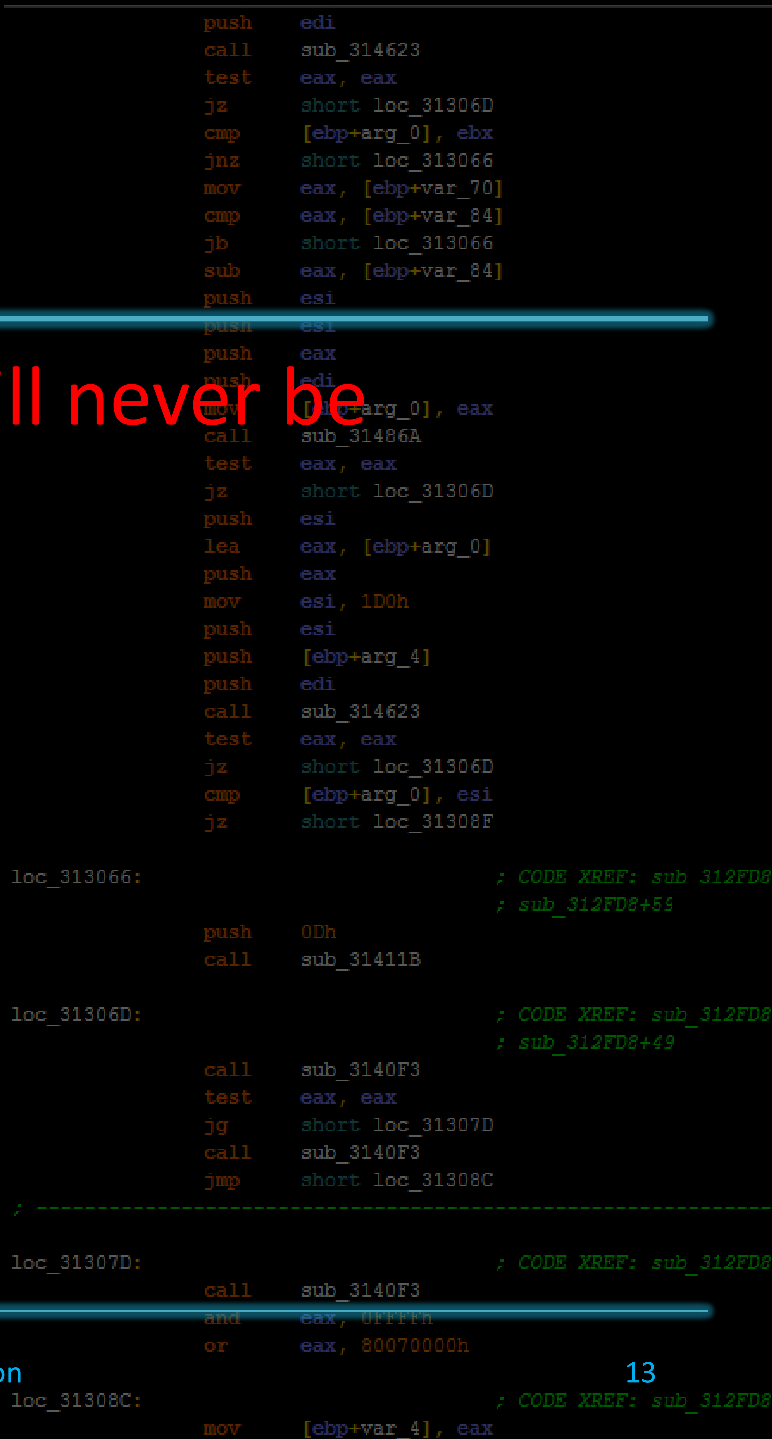
```
push    edi
call    sub_314623
test    eax, eax
jz      short loc_31306D
cmp     [ebp+arg_0], ebx
jnz     short loc_313066
mov     eax, [ebp+var_70]
cmp     eax, [ebp+var_84]
jb      short loc_313066
sub     eax, [ebp+var_84]
push    esi
push    esi
push    eax
push    edi
mov     [ebp+arg_0], eax
call    sub_31486A
test    eax, eax
jz      short loc_31306D
push    esi
lea     eax, [ebp+arg_0]
push    eax
mov     esi, 1D0h
push    esi
push    [ebp+arg_4]
push    edi
call    sub_314623
test    eax, eax
jz      short loc_31306D
cmp     [ebp+arg_0], esi
jz      short loc_31308F

loc_313066:                             ; CODE XREF: sub_312FD8
                                        ; sub_312FD8+55
push    0Dh
call    sub_3141

loc_31306D:                             ; CODE XREF: sub_312FD8
                                        ; sub_312FD8+49
call    sub_3140F3
test    eax, eax
jg      short loc_31307D
call    sub_3140F3
jmp     short loc_31308C
; ---------------------------------------------------------------

loc_31307D:                             ; CODE XREF: sub_312FD8
call    sub_3140F3
and     eax, 0FFFFh
or      eax, 80070000h

loc_31308C:                             ; CODE XREF: sub_312FD8
mov     [ebp+var_4], eax
```

Unsustainable Complexity



The Security Mindset

- Systems and applications will never be perfectly secure. Period.

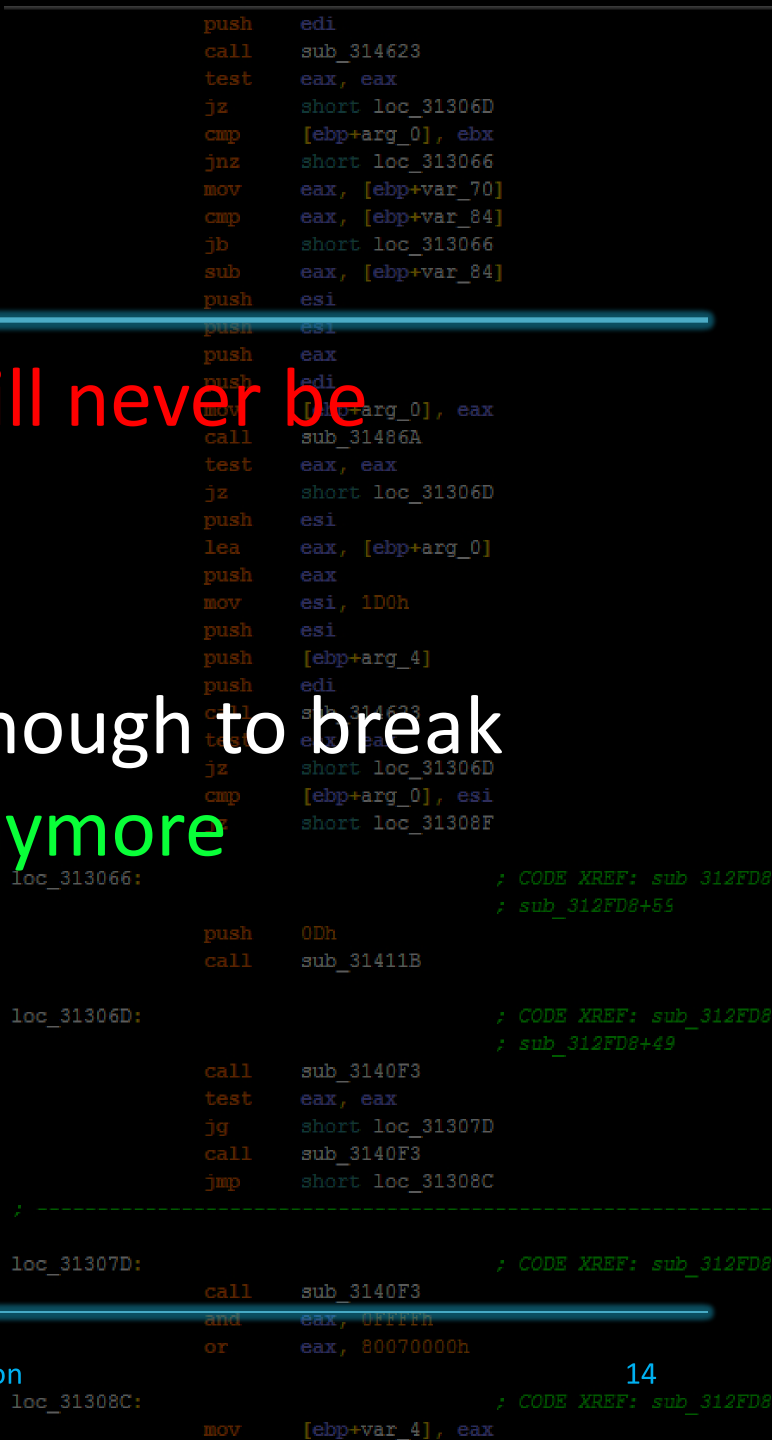


```
                push    edi
                call    sub_314623
                test    eax, eax
                jz      short loc_31306D
                cmp     [ebp+arg_0], ebx
                jnz     short loc_313066
                mov     eax, [ebp+var_70]
                cmp     eax, [ebp+var_84]
                jb      short loc_313066
                sub     eax, [ebp+var_84]
                push    esi
                push    esi
                push    eax
                push    edi
                mov     [ebp+arg_0], eax
                call    sub_31486A
                test    eax, eax
                jz      short loc_31306D
                push    esi
                lea     eax, [ebp+arg_0]
                push    eax
                mov     esi, 1D0h
                push    esi
                push    [ebp+arg_4]
                push    edi
                call    sub_314623
                test    eax, eax
                jz      short loc_31306D
                cmp     [ebp+arg_0], esi
                jz      short loc_31308F

loc_313066:                             ; CODE XREF: sub_312FD8
                                        ; sub_312FD8+55
                push    0Dh
                call    sub_31411B

loc_31306D:                             ; CODE XREF: sub_312FD8
                                        ; sub_312FD8+49
                call    sub_3140F3
                test    eax, eax
                jg      short loc_31307D
                call    sub_3140F3
                jmp     short loc_31308C
; ---------------------------------------------------------------------------

loc_31307D:                             ; CODE XREF: sub_312FD8
                call    sub_3140F3
                and     eax, 0FFFFh
                or      eax, 80070000h

loc_31308C:                             ; CODE XREF: sub_312FD8
                mov     [ebp+var_4], eax
```

The Security Mindset

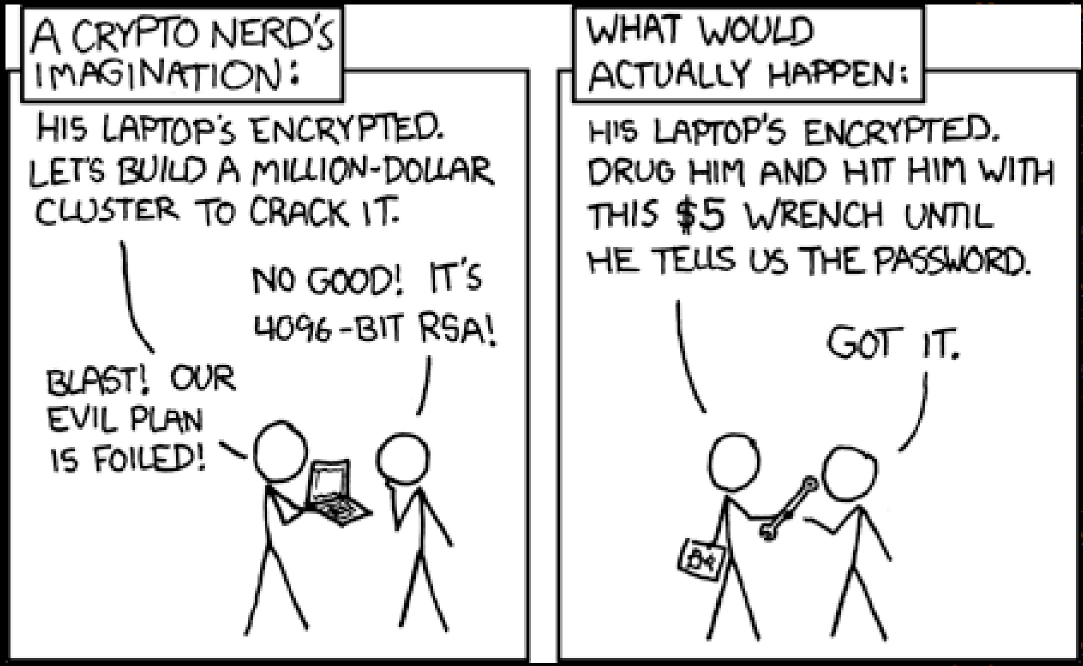
- Systems and applications will never be perfectly secure. Period.

- They just have to be hard enough to break that nobody can afford it anymore



```
                push    edi
                call    sub_314623
                test    eax, eax
                jz      short loc_31306D
                cmp     [ebp+arg_0], ebx
                jnz     short loc_313066
                mov     eax, [ebp+var_70]
                cmp     eax, [ebp+var_84]
                jb      short loc_313066
                sub     eax, [ebp+var_84]
                push    esi
                push    esi
                push    eax
                push    edi
                mov     [ebp+arg_0], eax
                call    sub_31486A
                test    eax, eax
                jz      short loc_31306D
                push    esi
                lea     eax, [ebp+arg_0]
                push    eax
                mov     esi, 1D0h
                push    esi
                push    [ebp+arg_4]
                push    edi
                call    sub_314623
                test    eax, eax
                jz      short loc_31306D
                cmp     [ebp+arg_0], esi
                jz      short loc_31308F

loc_313066:                             ; CODE XREF: sub_312FD8
                                        ; sub_312FD8+55
                push    0Dh
                call    sub_31411B

loc_31306D:                             ; CODE XREF: sub_312FD8
                                        ; sub_312FD8+49
                call    sub_3140F3
                test    eax, eax
                jg      short loc_31307D
                call    sub_3140F3
                jmp     short loc_31308C
; ---------------------------------------------------------------

loc_31307D:                             ; CODE XREF: sub_312FD8
                call    sub_3140F3
                and     eax, 0FFFFh
                or      eax, 80070000h

loc_31308C:                             ; CODE XREF: sub_312FD8
                mov     [ebp+var_4], eax
```

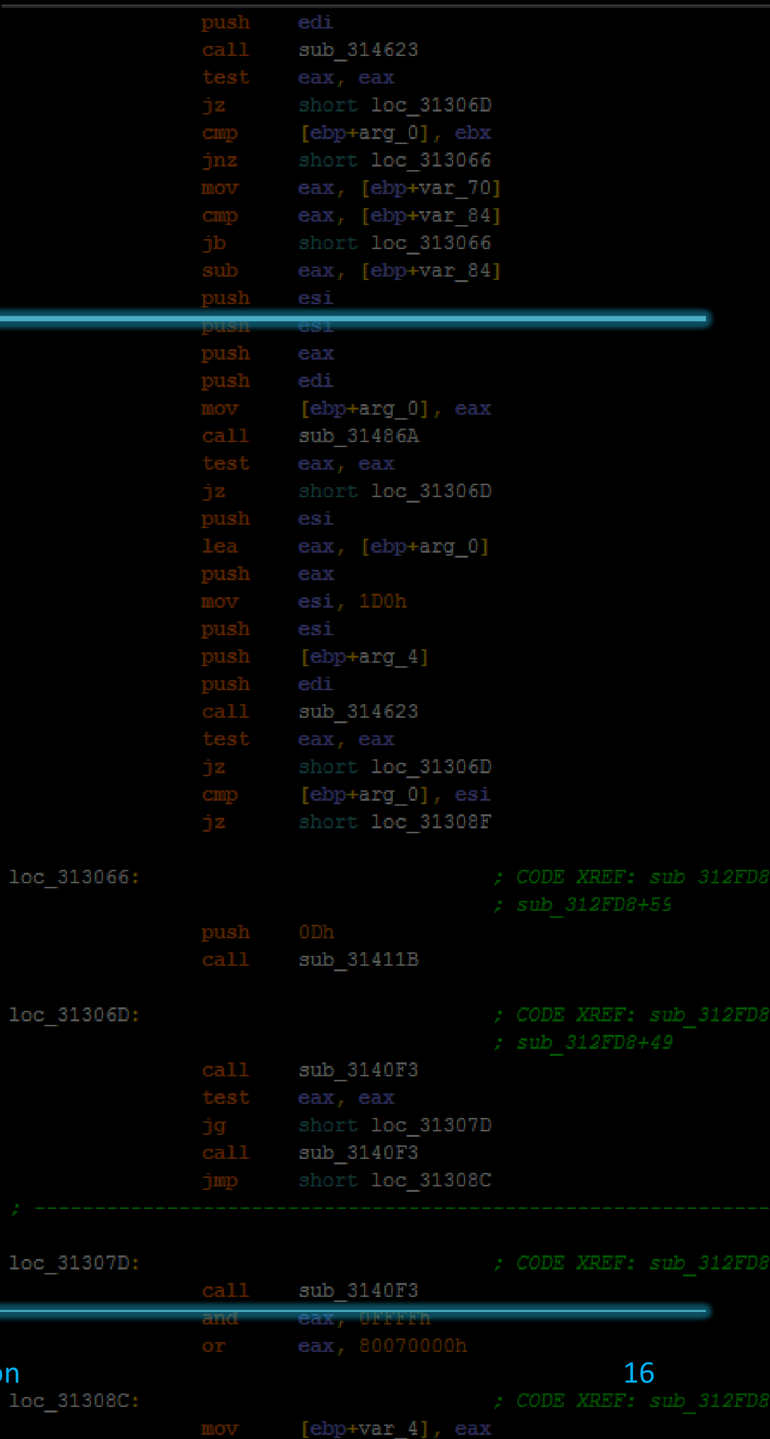
The Weakest Link - Humans


A CRYPTO NERD'S IMAGINATION:
HIS LAPTOP'S ENCRYPTED. LET'S BUILD A MILLION-DOLLAR CLUSTER TO CRACK IT.
NO GOOD! IT'S 4096-BIT RSA!
BLAST! OUR EVIL PLAN IS FOILED!
WHAT WOULD ACTUALLY HAPPEN:
HIS LAPTOP'S ENCRYPTED. DRUG HIM AND HIT HIM WITH THIS $5 WRENCH UNTIL HE TELLS US THE PASSWORD.
GOT IT.

https://xkcd.com/538/

Lecture Overview

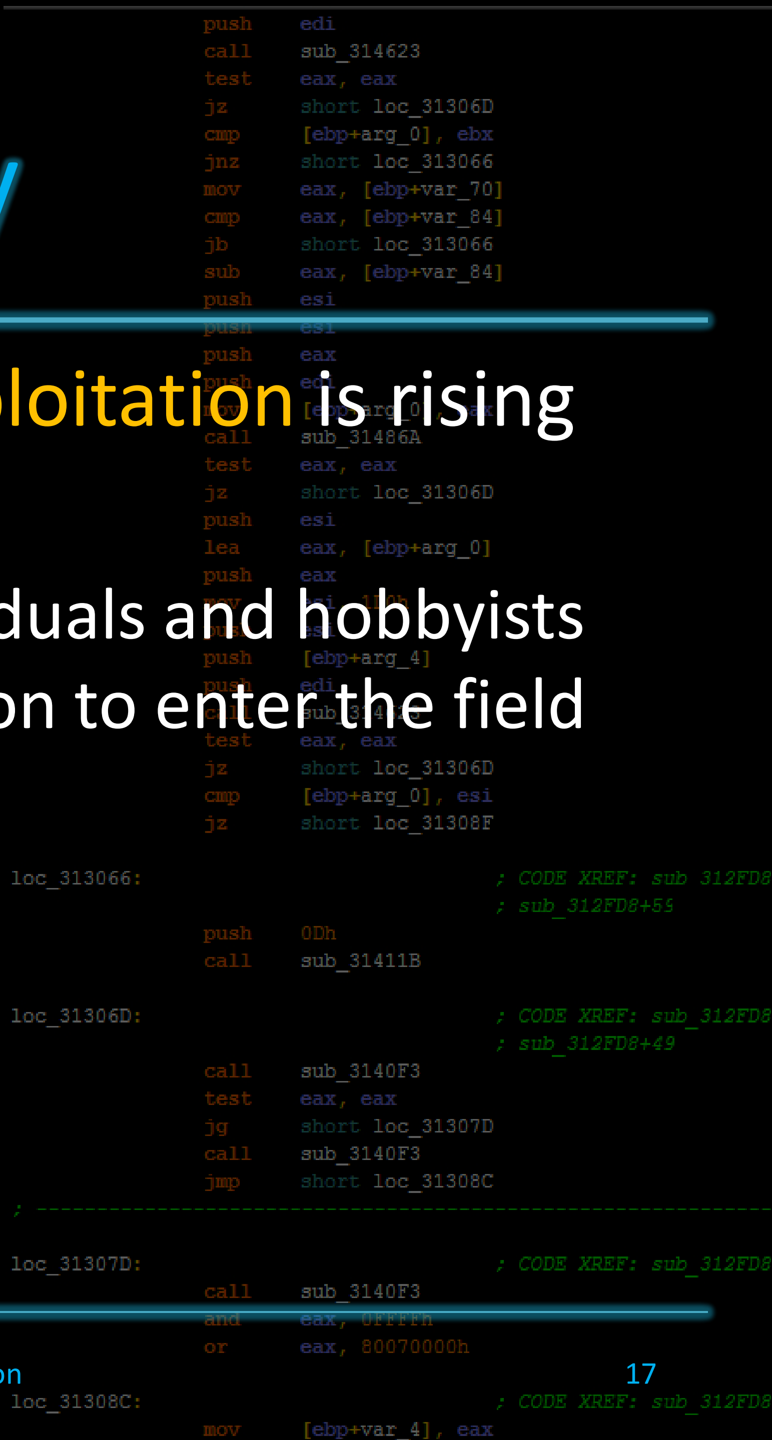
- Security
  - Security Today
  - Security Tomorrow
- Exploitation
  - Exploitation Today
  - Exploitation Tomorrow



```
                push    edi
                call    sub_314623
                test    eax, eax
                jz      short loc_31306D
                cmp     [ebp+arg_0], ebx
                jnz     short loc_313066
                mov     eax, [ebp+var_70]
                cmp     eax, [ebp+var_84]
                jb      short loc_313066
                sub     eax, [ebp+var_84]
                push    esi
                push    esi
                push    eax
                push    edi
                mov     [ebp+arg_0], eax
                call    sub_31486A
                test    eax, eax
                jz      short loc_31306D
                push    esi
                lea     eax, [ebp+arg_0]
                push    eax
                mov     esi, 1D0h
                push    esi
                push    [ebp+arg_4]
                push    edi
                call    sub_314623
                test    eax, eax
                jz      short loc_31306D
                cmp     [ebp+arg_0], esi
                jz      short loc_31308F

loc_313066:                             ; CODE XREF: sub_312FD8
                                        ; sub_312FD8+55
                push    0Dh
                call    sub_31411B

loc_31306D:                             ; CODE XREF: sub_312FD8
                                        ; sub_312FD8+49
                call    sub_3140F3
                test    eax, eax
                jg      short loc_31307D
                call    sub_3140F3
                jmp     short loc_31308C
; ---------------------------------------------------------------

loc_31307D:                             ; CODE XREF: sub_312FD8
                call    sub_3140F3
                and     eax, 0FFFFh
                or      eax, 80070000h

loc_31308C:                             ; CODE XREF: sub_312FD8
                mov     [ebp+var_4], eax
```

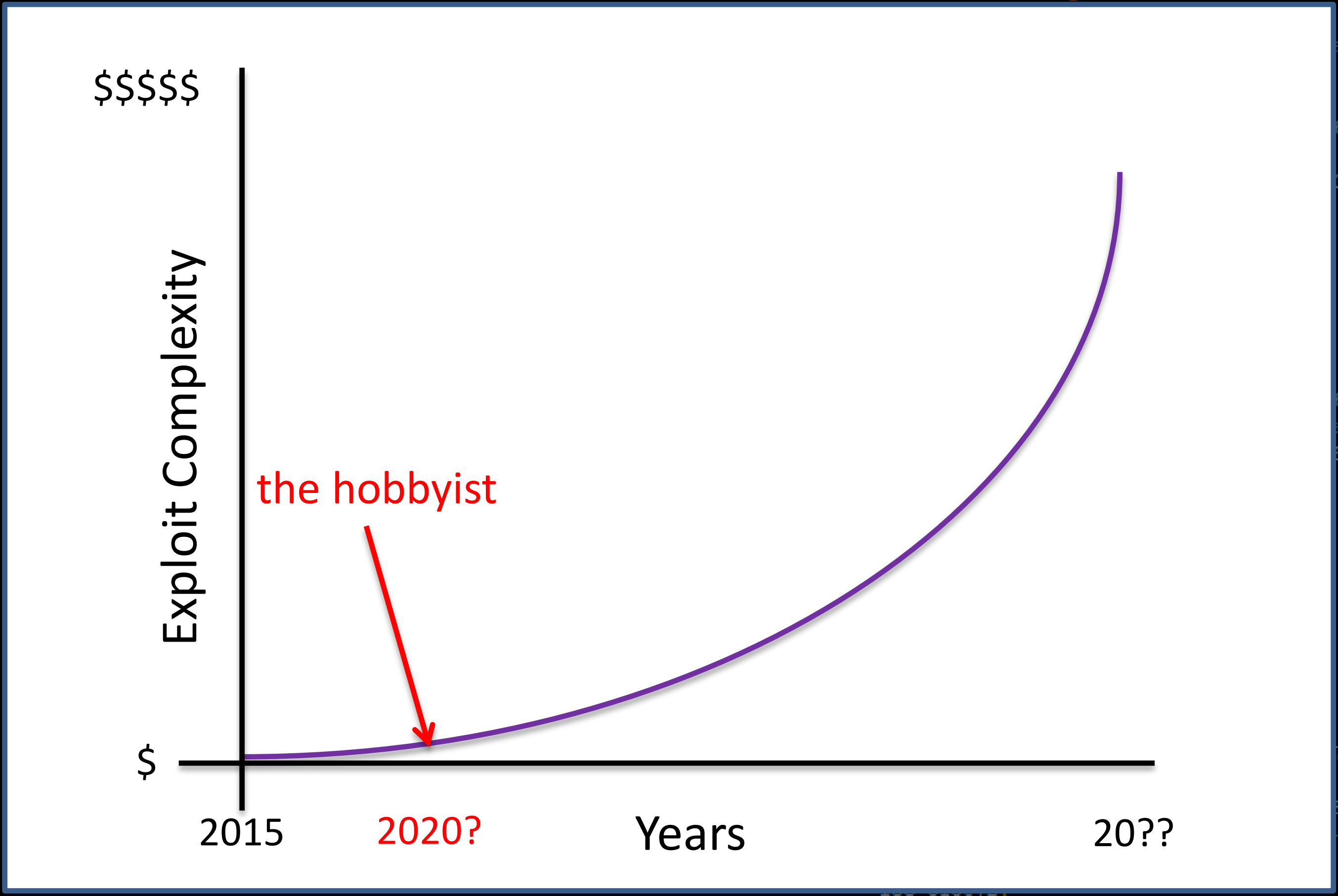
The Future of Security

- The entry bar for binary exploitation is rising faster and faster
  - It’s starting to outpace individuals and hobbyists interest, ability, and dedication to enter the field



```
                 push    edi
                 call    sub_314623
                 test    eax, eax
                 jz      short loc_31306D
                 cmp     [ebp+arg_0], ebx
                 jnz     short loc_313066
                 mov     eax, [ebp+var_70]
                 cmp     eax, [ebp+var_84]
                 jb      short loc_313066
                 sub     eax, [ebp+var_84]
                 push    esi
                 push    esi
                 push    eax
                 push    edi
                 mov     [ebp+arg_0], eax
                 call    sub_31486A
                 test    eax, eax
                 jz      short loc_31306D
                 push    esi
                 lea     eax, [ebp+arg_0]
                 push    eax
                 mov     esi, 110h
                 push    esi
                 push    [ebp+arg_4]
                 push    edi
                 call    sub_314623
                 test    eax, eax
                 jz      short loc_31306D
                 cmp     [ebp+arg_0], esi
                 jz      short loc_31308F

loc_313066:                             ; CODE XREF: sub_312FD8
                                        ; sub_312FD8+55
                 push    0Dh
                 call    sub_31411B

loc_31306D:                             ; CODE XREF: sub_312FD8
                                        ; sub_312FD8+49
                 call    sub_3140F3
                 test    eax, eax
                 jg      short loc_31307D
                 call    sub_3140F3
                 jmp     short loc_31308C
; ---------------------------------------------------------------------------

loc_31307D:                             ; CODE XREF: sub_312FD8
                 call    sub_3140F3
                 and     eax, 0FFFFh
                 or      eax, 80070000h

loc_31308C:                             ; CODE XREF: sub_312FD8
                 mov     [ebp+var_4], eax
```

Unsustainable Complexity

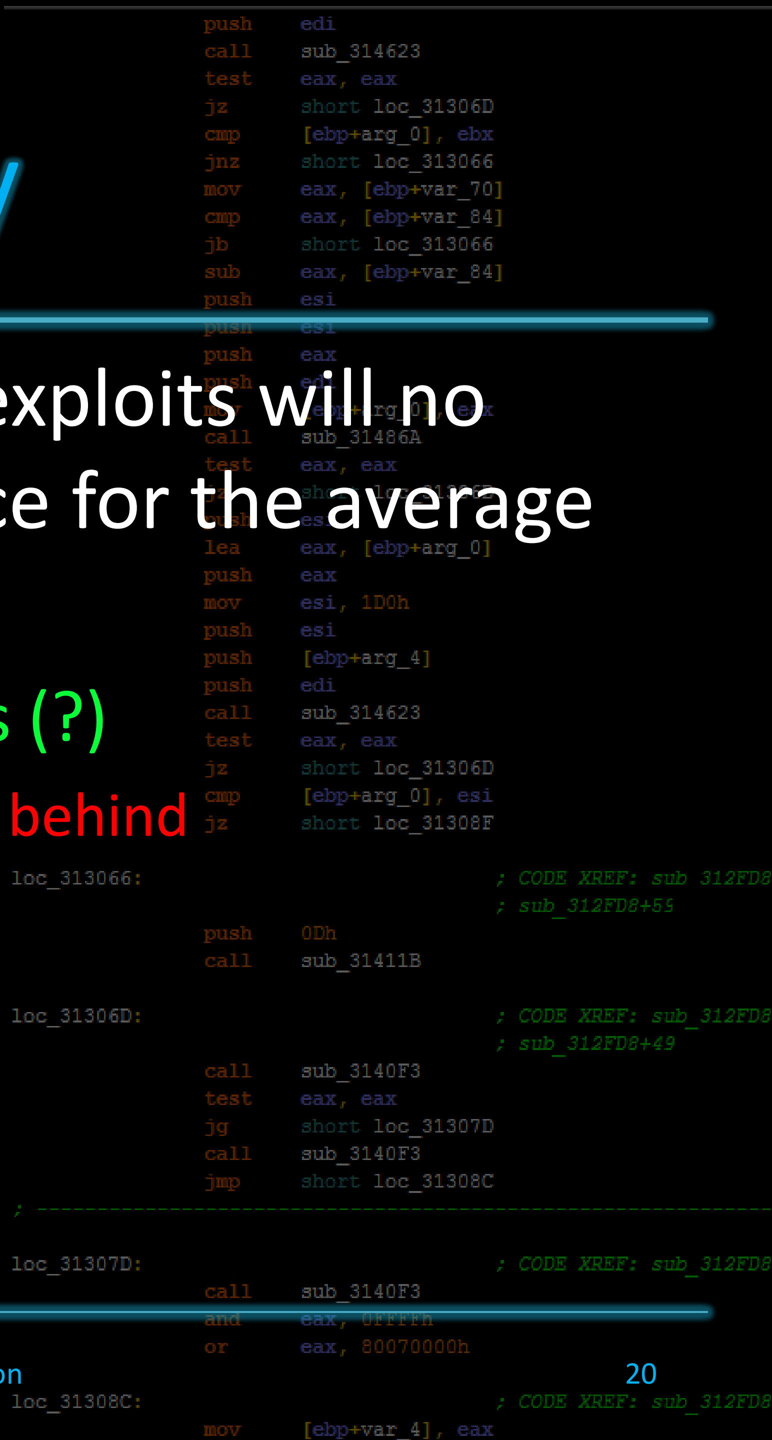


The Future of Security

- Memory corruption based exploits will no longer be feasible to produce for the average desktop or server

The Future of Security

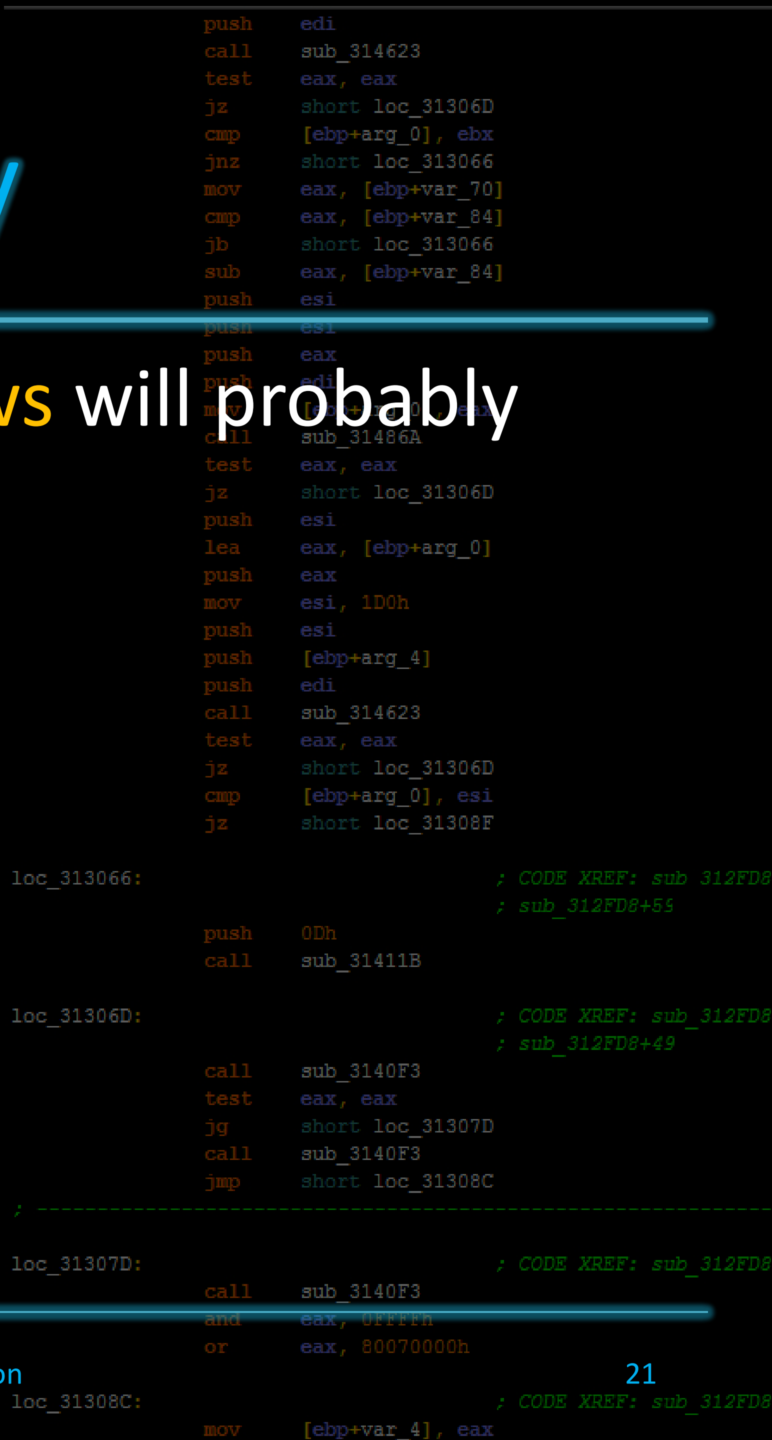
- Memory corruption based exploits will no longer be feasible to produce for the average desktop or server
  - In the immediate 10-20 years (?)
    - Embedded devices are further behind



```
                push    edi
                call    sub_314623
                test    eax, eax
                jz      short loc_31306D
                cmp     [ebp+arg_0], ebx
                jnz     short loc_313066
                mov     eax, [ebp+var_70]
                cmp     eax, [ebp+var_84]
                jb      short loc_313066
                sub     eax, [ebp+var_84]
                push    esi
                push    esi
                push    eax
                push    edi
                mov     [ebp+arg_0], eax
                call    sub_31486A
                test    eax, eax
                jz      short loc_31306D
                push    esi
                lea     eax, [ebp+arg_0]
                push    eax
                mov     esi, 1D0h
                push    esi
                push    [ebp+arg_4]
                push    edi
                call    sub_314623
                test    eax, eax
                jz      short loc_31306D
                cmp     [ebp+arg_0], esi
                jz      short loc_31308F

loc_313066:                             ; CODE XREF: sub_312FD8
                                        ; sub_312FD8+55
                push    0Dh
                call    sub_31411B

loc_31306D:                             ; CODE XREF: sub_312FD8
                                        ; sub_312FD8+49
                call    sub_3140F3
                test    eax, eax
                jg      short loc_31307D
                call    sub_3140F3
                jmp     short loc_31308C
; ---------------------------------------------------------------------------

loc_31307D:                             ; CODE XREF: sub_312FD8
                call    sub_3140F3
                and     eax, 0FFFFh
                or      eax, 80070000h

loc_31308C:                             ; CODE XREF: sub_312FD8
                mov     [ebp+var_4], eax
```

The Future of Security

- Implementation & logic flaws will probably always exist
  - You can't really fix stupid



```
                push    edi
                call    sub_314623
                test    eax, eax
                jz      short loc_31306D
                cmp     [ebp+arg_0], ebx
                jnz     short loc_313066
                mov     eax, [ebp+var_70]
                cmp     eax, [ebp+var_84]
                jb      short loc_313066
                sub     eax, [ebp+var_84]
                push    esi
                push    esi
                push    eax
                push    edi
                mov     [ebp+arg_0], eax
                call    sub_31486A
                test    eax, eax
                jz      short loc_31306D
                push    esi
                lea     eax, [ebp+arg_0]
                push    eax
                mov     esi, 1D0h
                push    esi
                push    [ebp+arg_4]
                push    edi
                call    sub_314623
                test    eax, eax
                jz      short loc_31306D
                cmp     [ebp+arg_0], esi
                jz      short loc_31308F

loc_313066:                             ; CODE XREF: sub_312FD8
                                        ; sub_312FD8+55
                push    0Dh
                call    sub_31411B

loc_31306D:                             ; CODE XREF: sub_312FD8
                                        ; sub_312FD8+49
                call    sub_3140F3
                test    eax, eax
                jg      short loc_31307D
                call    sub_3140F3
                jmp     short loc_31308C
; ---------------------------------------------------------------------------

loc_31307D:                             ; CODE XREF: sub_312FD8
                call    sub_3140F3
                and     eax, 0FFFFh
                or      eax, 80070000h

loc_31308C:                             ; CODE XREF: sub_312FD8
                mov     [ebp+var_4], eax
```

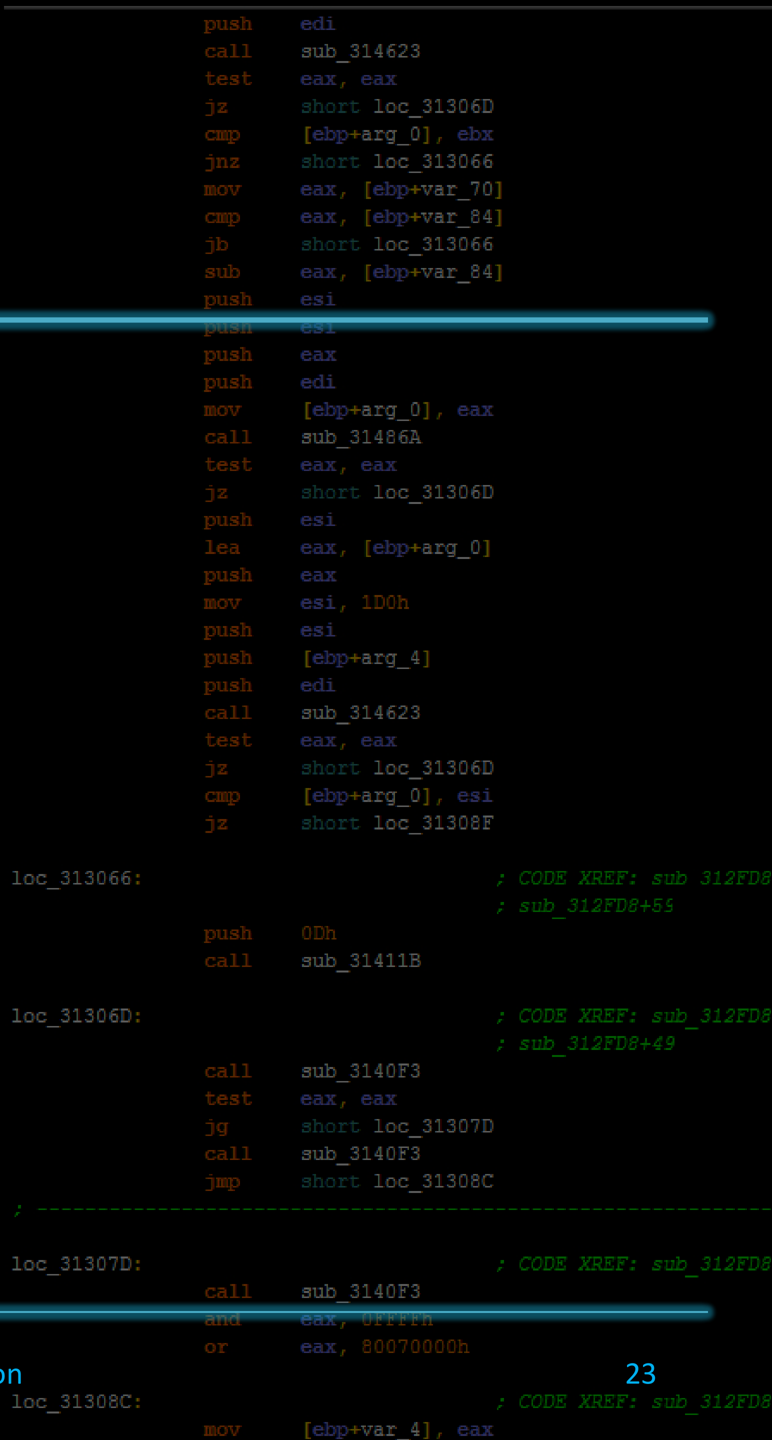
The Future of Security

- Implementation & logic flaws will probably always exist
  - You can't really fix stupid

- What we will see and discover more of:
  - Sponsored backdoors, 'cheating'
  - Hardware backdoors, flaws, supply chain trust
  - Crypto backdoors, subtle design flaws

Lecture Overview

- Security
  - Security Today
  - Security Tomorrow
- Exploitation
  - Exploitation Today
  - Exploitation Tomorrow



```
                push    edi
                call    sub_314623
                test    eax, eax
                jz      short loc_31306D
                cmp     [ebp+arg_0], ebx
                jnz     short loc_313066
                mov     eax, [ebp+var_70]
                cmp     eax, [ebp+var_84]
                jb      short loc_313066
                sub     eax, [ebp+var_84]
                push    esi
                push    esi
                push    eax
                push    edi
                mov     [ebp+arg_0], eax
                call    sub_31486A
                test    eax, eax
                jz      short loc_31306D
                push    esi
                lea     eax, [ebp+arg_0]
                push    eax
                mov     esi, 1D0h
                push    esi
                push    [ebp+arg_4]
                push    edi
                call    sub_314623
                test    eax, eax
                jz      short loc_31306D
                cmp     [ebp+arg_0], esi
                jz      short loc_31308F

loc_313066:                             ; CODE XREF: sub_312FD8
                                        ; sub_312FD8+55
                push    0Dh
                call    sub_31411B

loc_31306D:                             ; CODE XREF: sub_312FD8
                                        ; sub_312FD8+49
                call    sub_3140F3
                test    eax, eax
                jg      short loc_31307D
                call    sub_3140F3
                jmp     short loc_31308C
; ---------------------------------------------------------------

loc_31307D:                             ; CODE XREF: sub_312FD8
                call    sub_3140F3
                and     eax, 0FFFFh
                or      eax, 80070000h

loc_31308C:                             ; CODE XREF: sub_312FD8
                mov     [ebp+var_4], eax
```

This Course

- You spent hours looking for bugs
- You spent hours reversing in IDA
- You spent hours debugging with GDB
- You spent hours writing python

```
                push    edi
                call    sub_314623
                test    eax, eax
                jz      short loc_31306D
                cmp     [ebp+arg_0], ebx
                jnz     short loc_313066
                mov     eax, [ebp+var_70]
                cmp     eax, [ebp+var_84]
                jb      short loc_313066
                sub     eax, [ebp+var_84]
                push    esi
                push    esi
                push    eax
                push    edi
                mov     [ebp+arg_0], eax
                call    sub_31486A
                test    eax, eax
                jz      short loc_31306D
                push    esi
                lea     eax, [ebp+arg_0]
                push    eax
                mov     esi, 1D0h
                push    esi
                push    [ebp+arg_4]
                push    edi
                call    sub_314623
                test    eax, eax
                jz      short loc_31306D
                cmp     [ebp+arg_0], esi
                jz      short loc_31308F

loc_313066:                             ; CODE XREF: sub_312FD8
                                        ; sub_312FD8+55
                push    0Dh
                call    sub_31411B

loc_31306D:                             ; CODE XREF: sub_312FD8
                                        ; sub_312FD8+49
                call    sub_3140F3
                test    eax, eax
                jg      short loc_31307D
                call    sub_3140F3
                jmp     short loc_31308C
; ---------------------------------------------------------------------------

loc_31307D:                             ; CODE XREF: sub_312FD8
                call    sub_3140F3
                and     eax, 0FFFFh
                or      eax, 80070000h

loc_31308C:                             ; CODE XREF: sub_312FD8
                mov     [ebp+var_4], eax
```

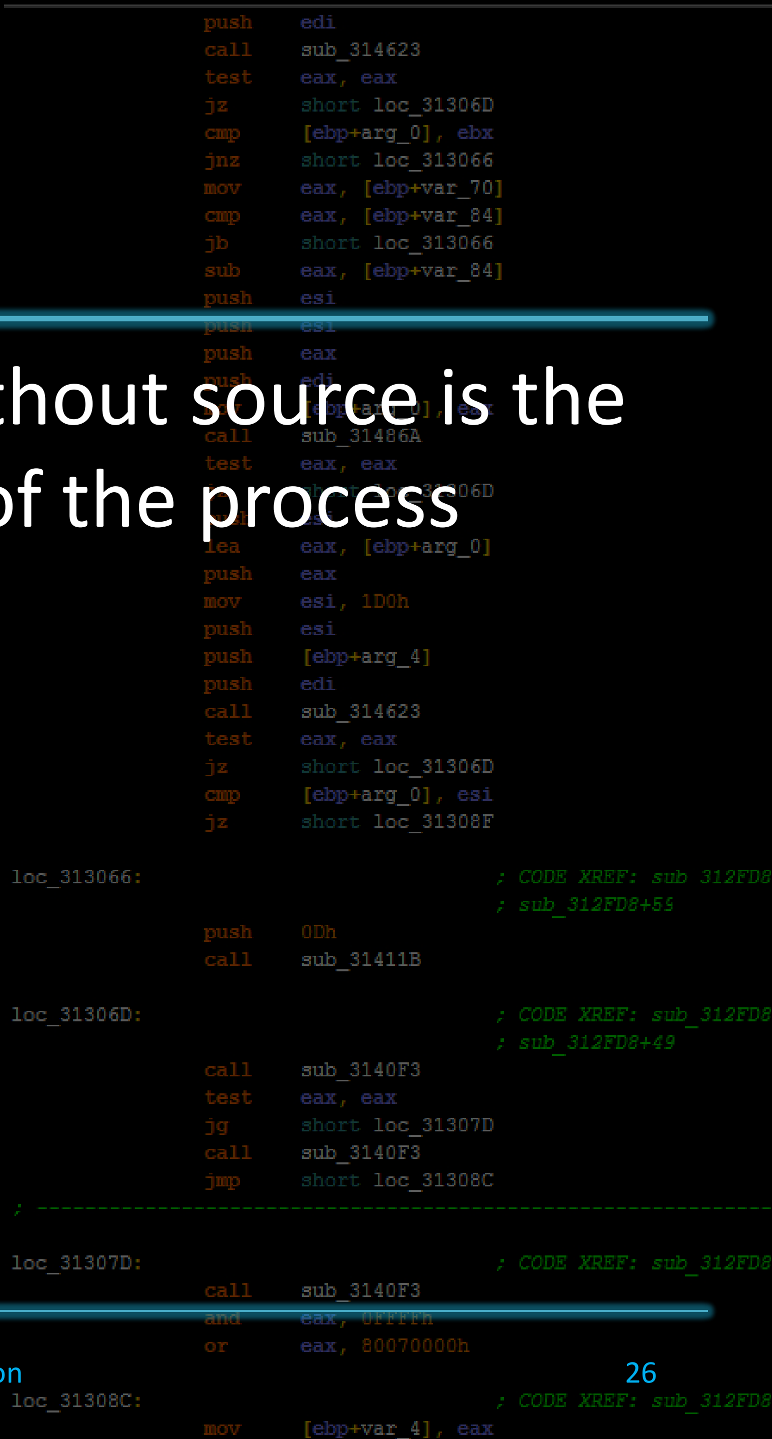
This Course

- You spent hours looking for bugs
- You spent hours reversing in IDA
- You spent hours debugging with GDB
- You spent hours writing python

```
                push    edi
                call    sub_314623
                test    eax, eax
                jz      short loc_31306D
                cmp     [ebp+arg_0], ebx
                jnz     short loc_313066
                mov     eax, [ebp+var_70]
                cmp     eax, [ebp+var_84]
                jb      short loc_313066
                sub     eax, [ebp+var_84]
                push    esi
                push    esi
                push    eax
                push    edi
                mov     [ebp+arg_0], eax
                call    sub_31486A
                test    eax, eax
                jz      short loc_31306D
                push    esi
                lea     eax, [ebp+arg_0]
                push    eax
                mov     esi, 1D0h
                push    esi
                push    [ebp+arg_4]
                push    edi
                call    sub_314623
                test    eax, eax
                jz      short loc_31306D
                cmp     [ebp+arg_0], esi
                jz      short loc_31308F

loc_313066:                             ; CODE XREF: sub_312FD8
                                        ; sub_312FD8+55
                push    0Dh
                call    sub_31411B

loc_31306D:                             ; CODE XREF: sub_312FD8
                                        ; sub_312FD8+49
                call    sub_3140F3
                test    eax, eax
                jg      short loc_31307D
                call    sub_3140F3
                jmp     short loc_31308C
; ---------------------------------------------------------------------------

loc_31307D:                             ; CODE XREF: sub_312FD8
                call    sub_3140F3
                and     eax, 0FFFFh
                or      eax, 80070000h

loc_31308C:                             ; CODE XREF: sub_312FD8
                mov     [ebp+var_4], eax
```

Bug Hunting

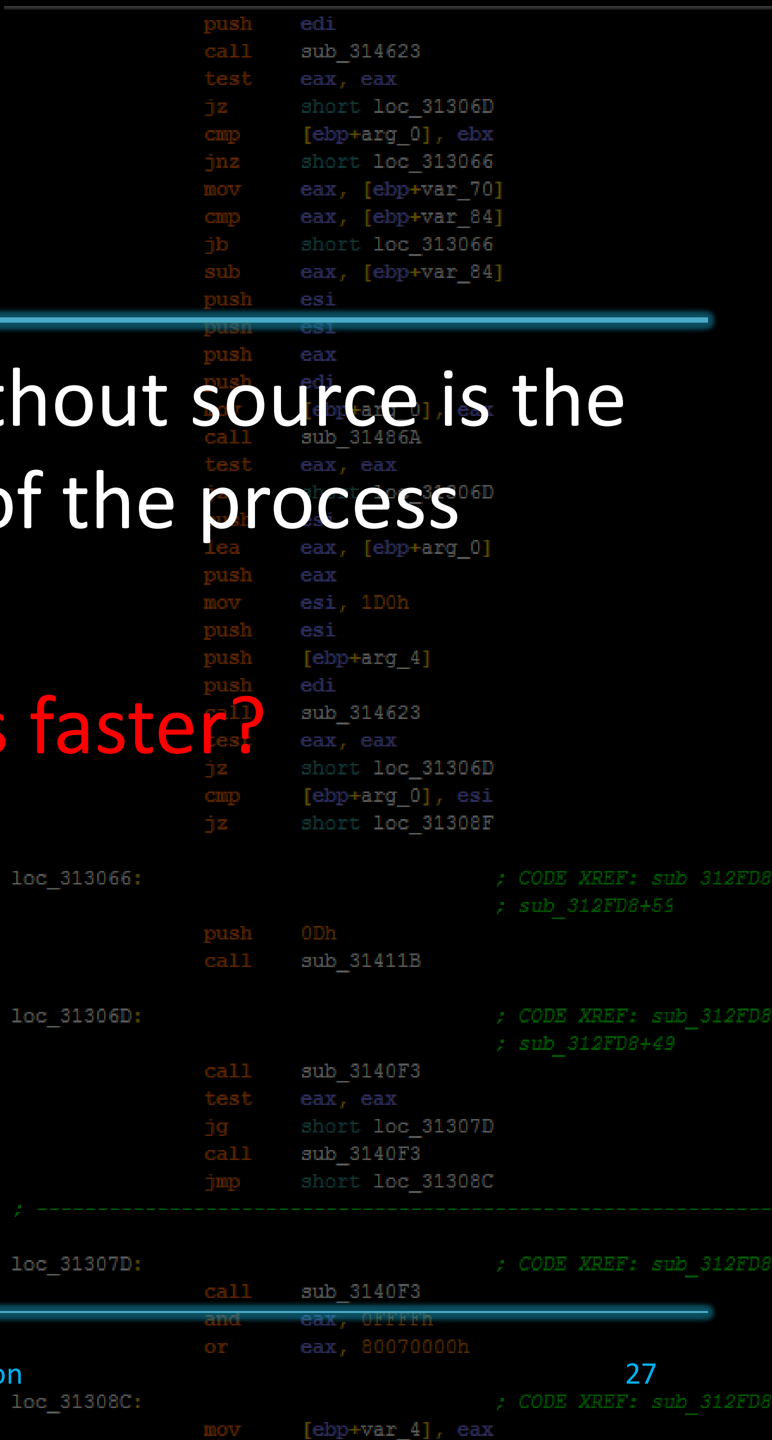
- Looking for bugs with or without source is the most time consuming part of the process



```
                  push    edi
                  call    sub_314623
                  test    eax, eax
                  jz      short loc_31306D
                  cmp     [ebp+arg_0], ebx
                  jnz     short loc_313066
                  mov     eax, [ebp+var_70]
                  cmp     eax, [ebp+var_84]
                  jb      short loc_313066
                  sub     eax, [ebp+var_84]
                  push    esi
                  push    esi
                  push    eax
                  push    edi
                  mov     [ebp+arg_0], esi
                  call    sub_31486A
                  test    eax, eax
                  jz      short loc_31306D
                  push    esi
                  lea     eax, [ebp+arg_0]
                  push    eax
                  mov     esi, 1D0h
                  push    esi
                  push    [ebp+arg_4]
                  push    edi
                  call    sub_314623
                  test    eax, eax
                  jz      short loc_31306D
                  cmp     [ebp+arg_0], esi
                  jz      short loc_31308F

loc_313066:                                  ; CODE XREF: sub_312FD8
                                             ; sub_312FD8+55
                  push    0Dh
                  call    sub_31411B

loc_31306D:                                  ; CODE XREF: sub_312FD8
                                             ; sub_312FD8+49
                  call    sub_3140F3
                  test    eax, eax
                  jg      short loc_31307D
                  call    sub_3140F3
                  jmp     short loc_31308C
; ---------------------------------------------------------------

loc_31307D:                                  ; CODE XREF: sub_312FD8
                  call    sub_3140F3
                  and     eax, 0FFFFh
                  or      eax, 80070000h

loc_31308C:                                  ; CODE XREF: sub_312FD8
                  mov     [ebp+var_4], eax
```

Bug Hunting

- Looking for bugs with or without source is the most time consuming part of the process
- How can we find these bugs faster?

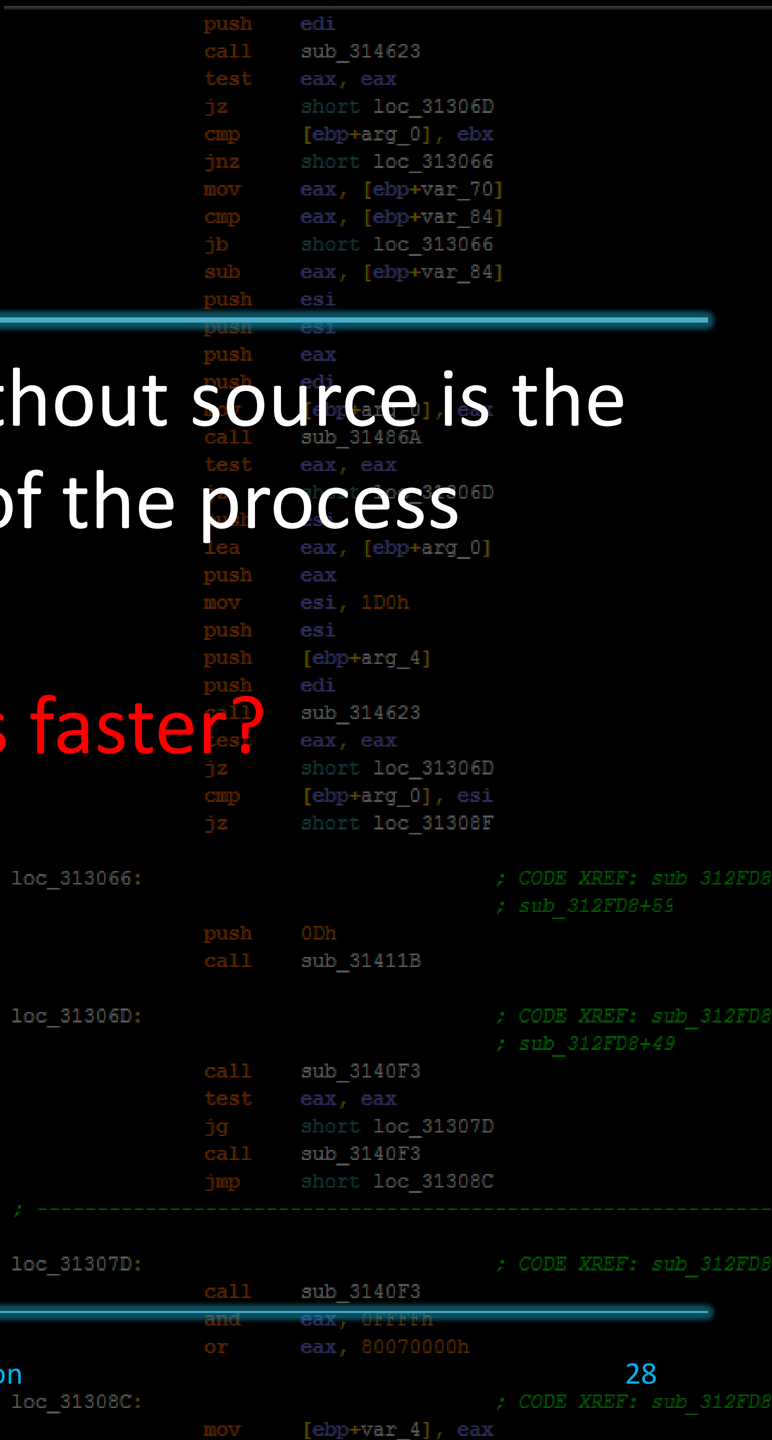


```
                push    edi
                call    sub_314623
                test    eax, eax
                jz      short loc_31306D
                cmp     [ebp+arg_0], ebx
                jnz     short loc_313066
                mov     eax, [ebp+var_70]
                cmp     eax, [ebp+var_84]
                jb      short loc_313066
                sub     eax, [ebp+var_84]
                push    esi
                push    esi
                push    eax
                push    edi
                [illegible] [ebp+arg_0], esi
                call    sub_31486A
                test    eax, eax
                jz      short loc_31306D
                push    esi
                lea     eax, [ebp+arg_0]
                push    eax
                mov     esi, 1D0h
                push    esi
                push    [ebp+arg_4]
                push    edi
                call    sub_314623
                test    eax, eax
                jz      short loc_31306D
                cmp     [ebp+arg_0], esi
                jz      short loc_31308F

loc_313066:                             ; CODE XREF: sub_312FD8
                                        ; sub_312FD8+55
                push    0Dh
                call    sub_31411B

loc_31306D:                             ; CODE XREF: sub_312FD8
                                        ; sub_312FD8+49
                call    sub_3140F3
                test    eax, eax
                jg      short loc_31307D
                call    sub_3140F3
                jmp     short loc_31308C
; ---------------------------------------------------------------------------

loc_31307D:                             ; CODE XREF: sub_312FD8
                call    sub_3140F3
                and     eax, 0FFFFh
                or      eax, 80070000h

loc_31308C:                             ; CODE XREF: sub_312FD8
                mov     [ebp+var_4], eax
```

Bug Hunting

- Looking for bugs with or without source is the most time consuming part of the process

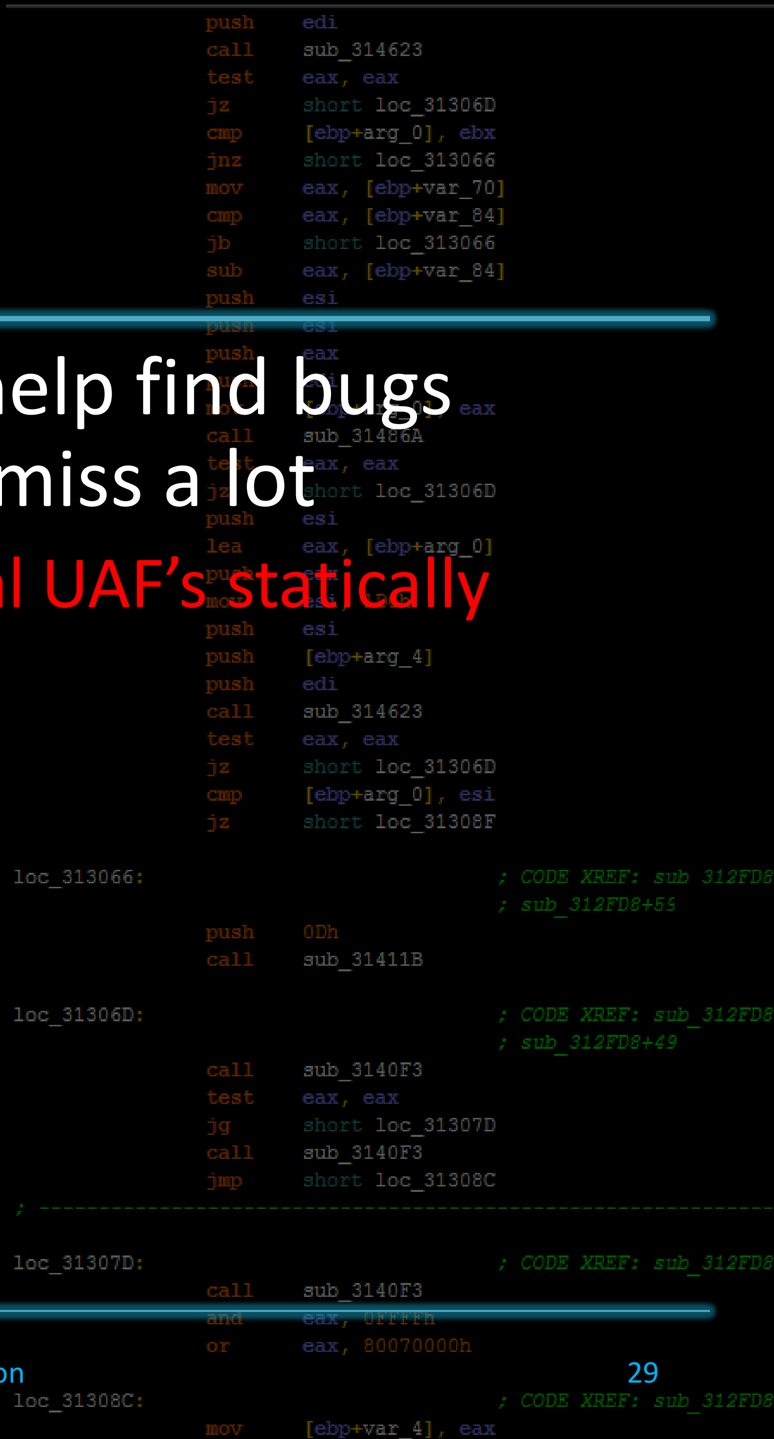
- How can we find these bugs faster?
  - Automation



```
                push    edi
                call    sub_314623
                test    eax, eax
                jz      short loc_31306D
                cmp     [ebp+arg_0], ebx
                jnz     short loc_313066
                mov     eax, [ebp+var_70]
                cmp     eax, [ebp+var_84]
                jb      short loc_313066
                sub     eax, [ebp+var_84]
                push    esi
                push    esi
                push    eax
                push    edi
                        [ebp+arg_0], esi
                call    sub_31486A
                test    eax, eax
                        short loc_31306D
                        esi
                lea     eax, [ebp+arg_0]
                push    eax
                mov     esi, 1D0h
                push    esi
                push    [ebp+arg_4]
                push    edi
                call    sub_314623
                test    eax, eax
                jz      short loc_31306D
                cmp     [ebp+arg_0], esi
                jz      short loc_31308F

loc_313066:                             ; CODE XREF: sub_312FD8
                                        ; sub_312FD8+55
                push    0Dh
                call    sub_31411B

loc_31306D:                             ; CODE XREF: sub_312FD8
                                        ; sub_312FD8+49
                call    sub_3140F3
                test    eax, eax
                jg      short loc_31307D
                call    sub_3140F3
                jmp     short loc_31308C
; ---------------------------------------------------------------

loc_31307D:                             ; CODE XREF: sub_312FD8
                call    sub_3140F3
                and     eax, 0FFFFh
                or      eax, 80070000h

loc_31308C:                             ; CODE XREF: sub_312FD8
                mov     [ebp+var_4], eax
```

Static Code Analyzers

- Source code analyzers can help find bugs statically, but they can also miss a lot
  - Very hard to detect many real UAF’s statically

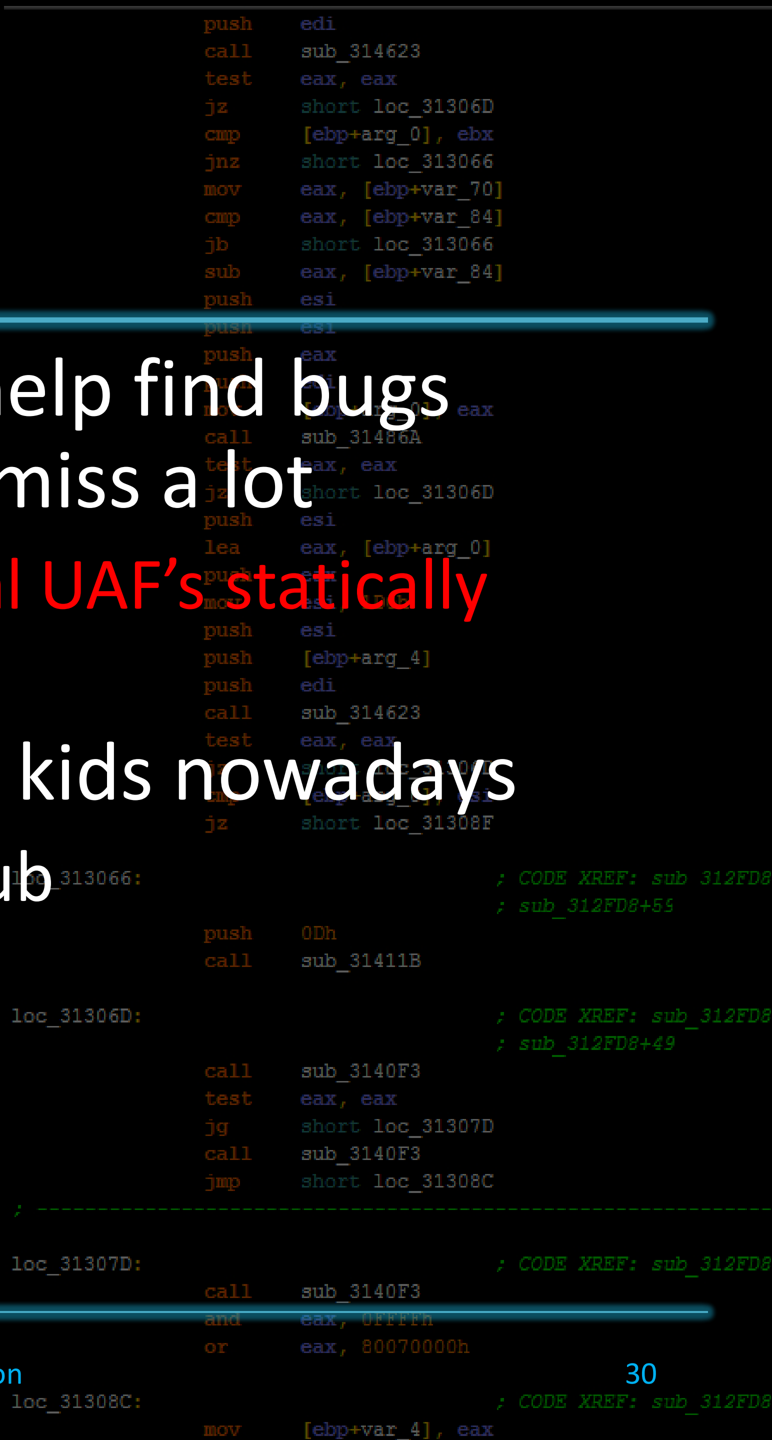


```
                push    edi
                call    sub_314623
                test    eax, eax
                jz      short loc_31306D
                cmp     [ebp+arg_0], ebx
                jnz     short loc_313066
                mov     eax, [ebp+var_70]
                cmp     eax, [ebp+var_84]
                jb      short loc_313066
                sub     eax, [ebp+var_84]
                push    esi
                push    esi
                push    eax
                push    edi
                mov     [ebp+arg_0], eax
                call    sub_31486A
                test    eax, eax
                jz      short loc_31306D
                push    esi
                lea     eax, [ebp+arg_0]
                push    eax
                mov     esi, [illegible]
                push    esi
                push    [ebp+arg_4]
                push    edi
                call    sub_314623
                test    eax, eax
                jz      short loc_31306D
                cmp     [ebp+arg_0], esi
                jz      short loc_31308F

loc_313066:                             ; CODE XREF: sub_312FD8
                                        ; sub_312FD8+55
                push    0Dh
                call    sub_31411B

loc_31306D:                             ; CODE XREF: sub_312FD8
                                        ; sub_312FD8+49
                call    sub_3140F3
                test    eax, eax
                jg      short loc_31307D
                call    sub_3140F3
                jmp     short loc_31308C
; ---------------------------------------------------------------

loc_31307D:                             ; CODE XREF: sub_312FD8
                call    sub_3140F3
                and     eax, 0FFFFh
                or      eax, 80070000h

loc_31308C:                             ; CODE XREF: sub_312FD8
                mov     [ebp+var_4], eax
```

Static Code Analyzers

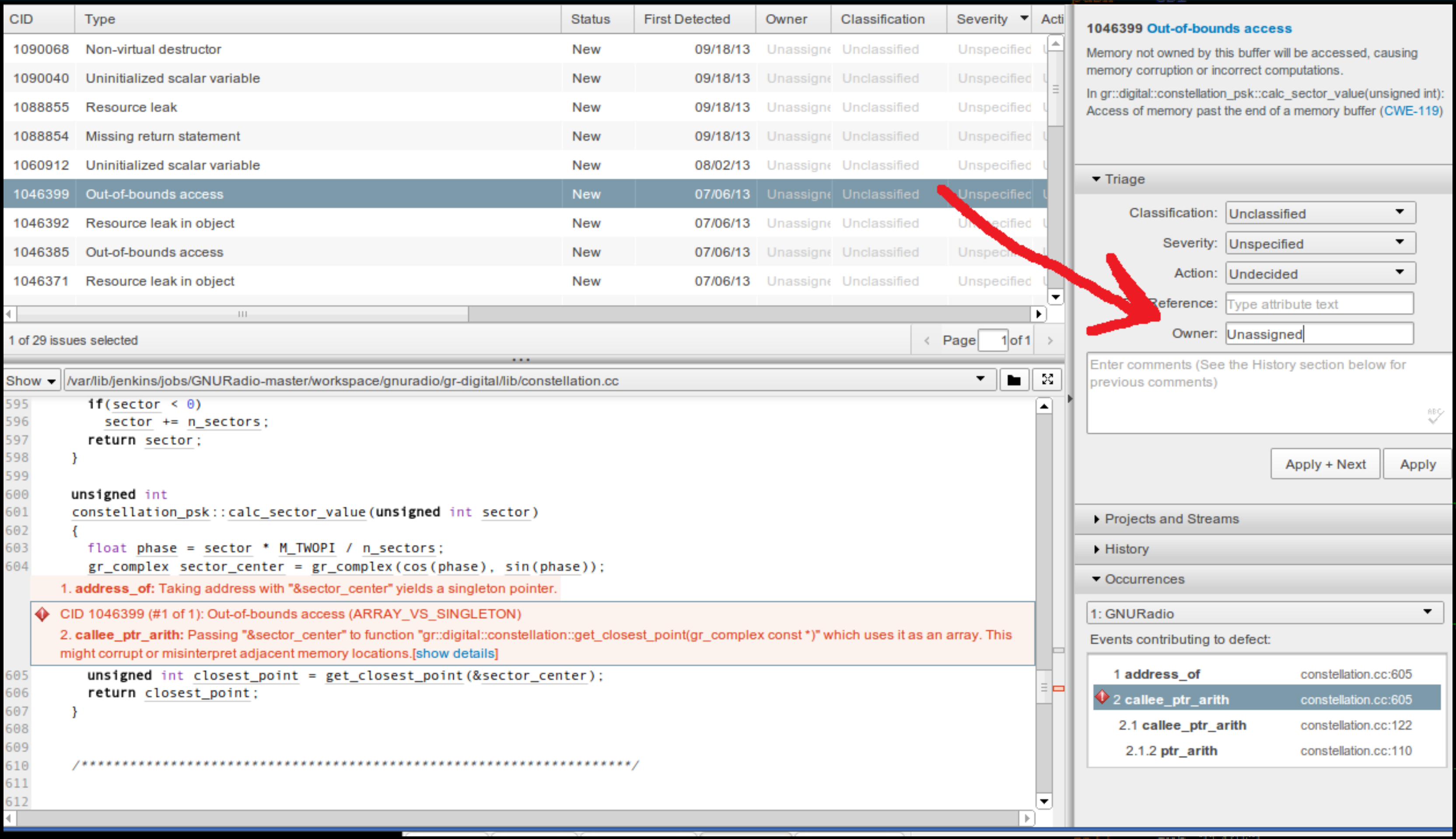
- Source code analyzers can help find bugs statically, but they can also miss a lot
  - Very hard to detect many real UAF’s statically

- Coverity is popular with the kids nowadays
  - integrates straight with GitHub



```
                push    edi
                call    sub_314623
                test    eax, eax
                jz      short loc_31306D
                cmp     [ebp+arg_0], ebx
                jnz     short loc_313066
                mov     eax, [ebp+var_70]
                cmp     eax, [ebp+var_84]
                jb      short loc_313066
                sub     eax, [ebp+var_84]
                push    esi
                push    esi
                push    eax
                push    edi
                mov     [ebp+arg_0], eax
                call    sub_31486A
                test    eax, eax
                jz      short loc_31306D
                push    esi
                lea     eax, [ebp+arg_0]
                push    eax
                mov     esi, [illegible]
                push    esi
                push    [ebp+arg_4]
                push    edi
                call    sub_314623
                test    eax, eax
                jz      short loc_3130[illegible]
                cmp     [ebp+arg_0], esi
                jz      short loc_31308F

loc_313066:                             ; CODE XREF: sub_312FD8
                                        ; sub_312FD8+55
                push    0Dh
                call    sub_31411B

loc_31306D:                             ; CODE XREF: sub_312FD8
                                        ; sub_312FD8+49
                call    sub_3140F3
                test    eax, eax
                jg      short loc_31307D
                call    sub_3140F3
                jmp     short loc_31308C
; ---------------------------------------------------------------

loc_31307D:                             ; CODE XREF: sub_312FD8
                call    sub_3140F3
                and     eax, 0FFFFh
                or      eax, 80070000h

loc_31308C:                             ; CODE XREF: sub_312FD8
                mov     [ebp+var_4], eax
```

Coverity



CID	Type	Status	First Detected	Owner	Classification	Severity	Acti
1090068	Non-virtual destructor	New	09/18/13	Unassigne	Unclassified	Unspecified	U
1090040	Uninitialized scalar variable	New	09/18/13	Unassigne	Unclassified	Unspecified	U
1088855	Resource leak	New	09/18/13	Unassigne	Unclassified	Unspecified	U
1088854	Missing return statement	New	09/18/13	Unassigne	Unclassified	Unspecified	U
1060912	Uninitialized scalar variable	New	08/02/13	Unassigne	Unclassified	Unspecified	U
1046399	Out-of-bounds access	New	07/06/13	Unassigne	Unclassified	Unspecified	U
1046392	Resource leak in object	New	07/06/13	Unassigne	Unclassified	Unspecified	U
1046385	Out-of-bounds access	New	07/06/13	Unassigne	Unclassified	Unspecified	U
1046371	Resource leak in object	New	07/06/13	Unassigne	Unclassified	Unspecified	U

1 of 29 issues selected

/var/lib/jenkins/jobs/GNURadio-master/workspace/gnuradio/gr-digital/lib/constellation.cc

```
    if(sector < 0)
      sector += n_sectors;
    return sector;
  }

  unsigned int
  constellation_psk::calc_sector_value(unsigned int sector)
  {
    float phase = sector * M_TWOPI / n_sectors;
    gr_complex sector_center = gr_complex(cos(phase), sin(phase));
```

1. **address_of:** Taking address with "&sector_center" yields a singleton pointer.

CID 1046399 (#1 of 1): Out-of-bounds access (ARRAY_VS_SINGLETON)

2. **callee_ptr_arith:** Passing "&sector_center" to function "gr::digital::constellation::get_closest_point(gr_complex const *)" which uses it as an array. This might corrupt or misinterpret adjacent memory locations.[show details]

```
    unsigned int closest_point = get_closest_point(&sector_center);
    return closest_point;
  }



  /*********************************************************************/


```

1046399 Out-of-bounds access

Memory not owned by this buffer will be accessed, causing memory corruption or incorrect computations.

In gr::digital::constellation_psk::calc_sector_value(unsigned int): Access of memory past the end of a memory buffer (CWE-119)

Triage

Classification: Unclassified

Severity: Unspecified

Action: Undecided

Reference: Type attribute text

Owner: Unassigned

Enter comments (See the History section below for previous comments)

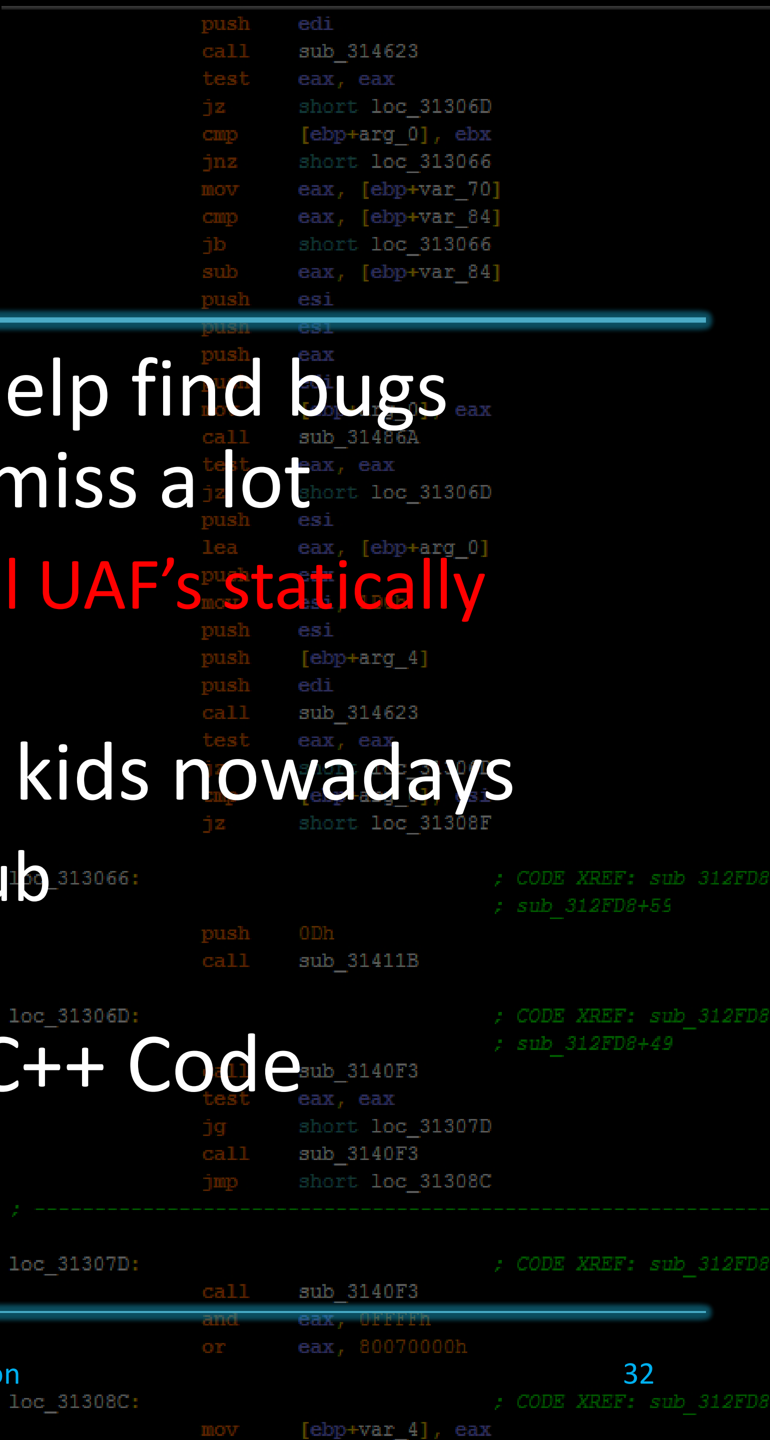
1: GNURadio

Events contributing to defect:

Event	Location
1 **address_of**	constellation.cc:605
2 **callee_ptr_arith**	constellation.cc:605
2.1 **callee_ptr_arith**	constellation.cc:122
2.1.2 **ptr_arith**	constellation.cc:110

Static Code Analyzers

- Source code analyzers can help find bugs statically, but they can also miss a lot
  - Very hard to detect many real UAF's statically
- Coverity is popular with the kids nowadays
  - integrates straight with GitHub
- Tons of good options for C/C++ Code
  - http://spinroot.com/static/

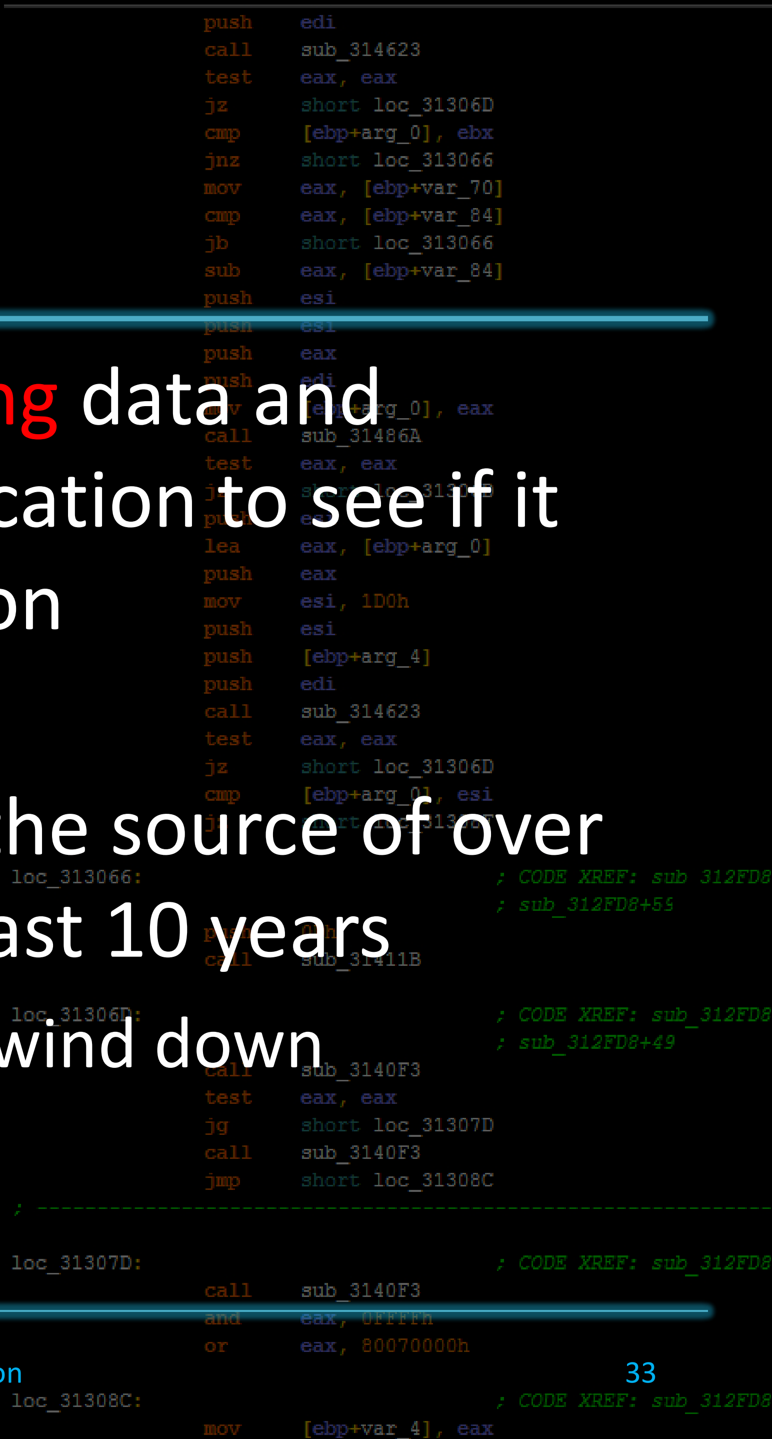


```
        push    edi
        call    sub_314623
        test    eax, eax
        jz      short loc_31306D
        cmp     [ebp+arg_0], ebx
        jnz     short loc_313066
        mov     eax, [ebp+var_70]
        cmp     eax, [ebp+var_84]
        jb      short loc_313066
        sub     eax, [ebp+var_84]
        push    esi
        push    esi
        push    eax
        [illegible]
        mov     [illegible] eax
        call    sub_31486A
        test    eax, eax
        jz      short loc_31306D
        push    esi
        lea     eax, [ebp+arg_0]
        [illegible]
        mov     esi, [illegible]
        push    esi
        push    [ebp+arg_4]
        push    edi
        call    sub_314623
        test    eax, eax
        [illegible]
        cmp     [ebp+arg_0], esi
        jz      short loc_31308F

loc_313066:                             ; CODE XREF: sub_312FD8
                                        ; sub_312FD8+55
        push    0Dh
        call    sub_31411B

loc_31306D:                             ; CODE XREF: sub_312FD8
                                        ; sub_312FD8+49
        call    sub_3140F3
        test    eax, eax
        jg      short loc_31307D
        call    sub_3140F3
        jmp     short loc_31308C
; ---------------------------------------------------------------------------

loc_31307D:                             ; CODE XREF: sub_312FD8
        call    sub_3140F3
        and     eax, 0FFFFh
        or      eax, 80070000h

loc_31308C:                             ; CODE XREF: sub_312FD8
        mov     [ebp+var_4], eax
```

Fuzzing

- Fuzzing – The act of mangling data and throwing it at a target application to see if it mishandles it in some fashion

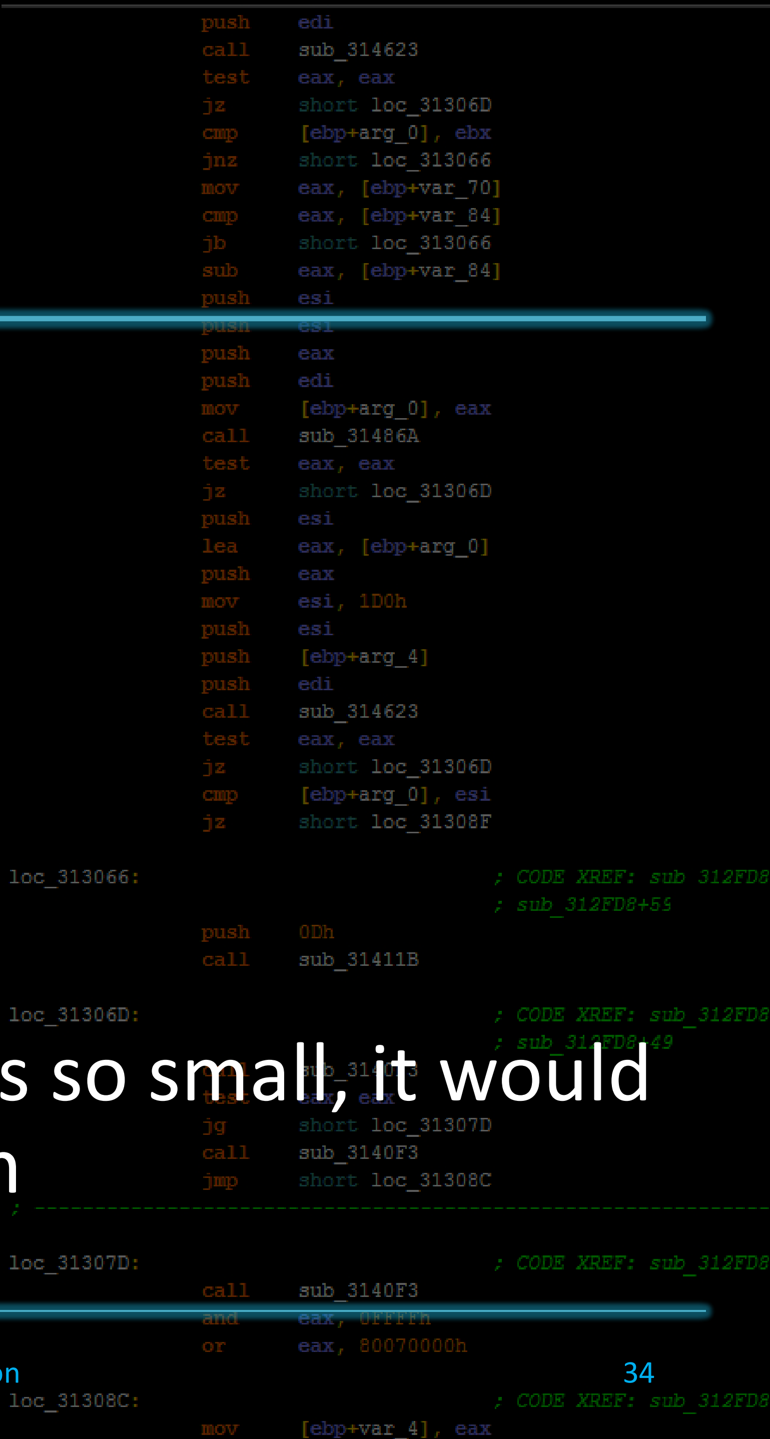
- Fuzzing has probably been the source of over 95% of the bugs from the past 10 years
  – The fuzzing era is starting to wind down



```
        push    edi
        call    sub_314623
        test    eax, eax
        jz      short loc_31306D
        cmp     [ebp+arg_0], ebx
        jnz     short loc_313066
        mov     eax, [ebp+var_70]
        cmp     eax, [ebp+var_84]
        jb      short loc_313066
        sub     eax, [ebp+var_84]
        push    esi
        push    esi
        push    eax
        push    edi
        mov     [ebp+arg_0], eax
        call    sub_31486A
        test    eax, eax
        jz      short loc_31306D
        push    eax
        lea     eax, [ebp+arg_0]
        push    eax
        mov     esi, 1D0h
        push    esi
        push    [ebp+arg_4]
        push    edi
        call    sub_314623
        test    eax, eax
        jz      short loc_31306D
        cmp     [ebp+arg_0], esi
        jz      short loc_313

loc_313066:                             ; CODE XREF: sub_312FD8
                                        ; sub_312FD8+55
        push
        call    sub_31411B

loc_31306D:                             ; CODE XREF: sub_312FD8
                                        ; sub_312FD8+49
        call    sub_3140F3
        test    eax, eax
        jg      short loc_31307D
        call    sub_3140F3
        jmp     short loc_31308C
; ---------------------------------------------------------------------------

loc_31307D:                             ; CODE XREF: sub_312FD8
        call    sub_3140F3
        and     eax, 0FFFFh
        or      eax, 80070000h

loc_31308C:                             ; CODE XREF: sub_312FD8
        mov     [ebp+var_4], eax
```

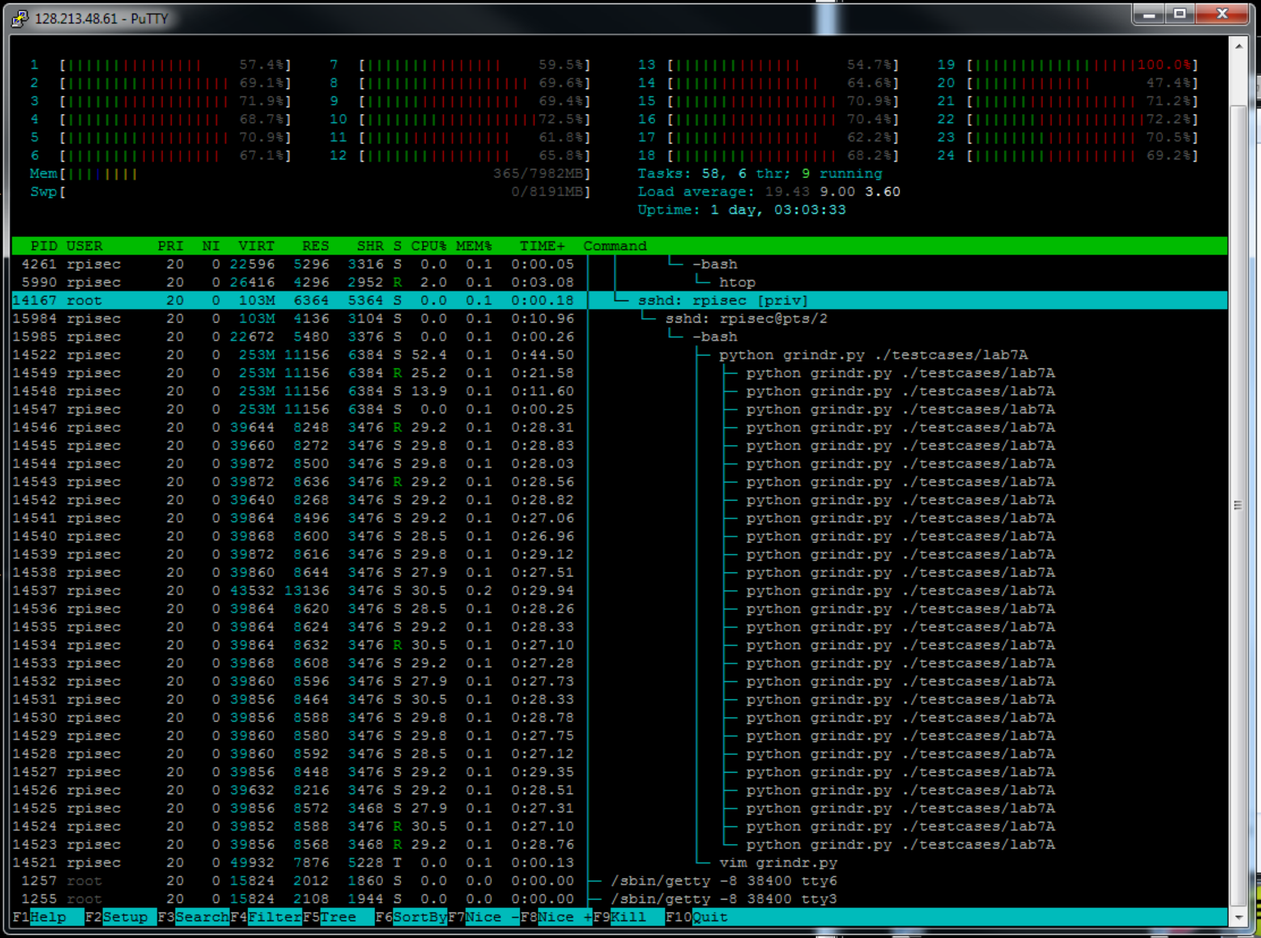
Fuzzing

- Remember these labs?
  - 7C
  - 7A
  - 9C
  - 9A
  - …
- Since the scope of the labs is so small, it would have been easy to fuzz them



```
                push    edi
                call    sub_314623
                test    eax, eax
                jz      short loc_31306D
                cmp     [ebp+arg_0], ebx
                jnz     short loc_313066
                mov     eax, [ebp+var_70]
                cmp     eax, [ebp+var_84]
                jb      short loc_313066
                sub     eax, [ebp+var_84]
                push    esi
                push    esi
                push    eax
                push    edi
                mov     [ebp+arg_0], eax
                call    sub_31486A
                test    eax, eax
                jz      short loc_31306D
                push    esi
                lea     eax, [ebp+arg_0]
                push    eax
                mov     esi, 1D0h
                push    esi
                push    [ebp+arg_4]
                push    edi
                call    sub_314623
                test    eax, eax
                jz      short loc_31306D
                cmp     [ebp+arg_0], esi
                jz      short loc_31308F

loc_313066:                             ; CODE XREF: sub_312FD8
                                        ; sub_312FD8+55
                push    0Dh
                call    sub_31411B

loc_31306D:                             ; CODE XREF: sub_312FD8
                                        ; sub_312FD8+49
                call    sub_3140F3
                test    eax, eax
                jg      short loc_31307D
                call    sub_3140F3
                jmp     short loc_31308C
; ---------------------------------------------------------------------------

loc_31307D:                             ; CODE XREF: sub_312FD8
                call    sub_3140F3
                and     eax, 0FFFFh
                or      eax, 80070000h

loc_31308C:                             ; CODE XREF: sub_312FD8
                mov     [ebp+var_4], eax
```

Instant Bugs

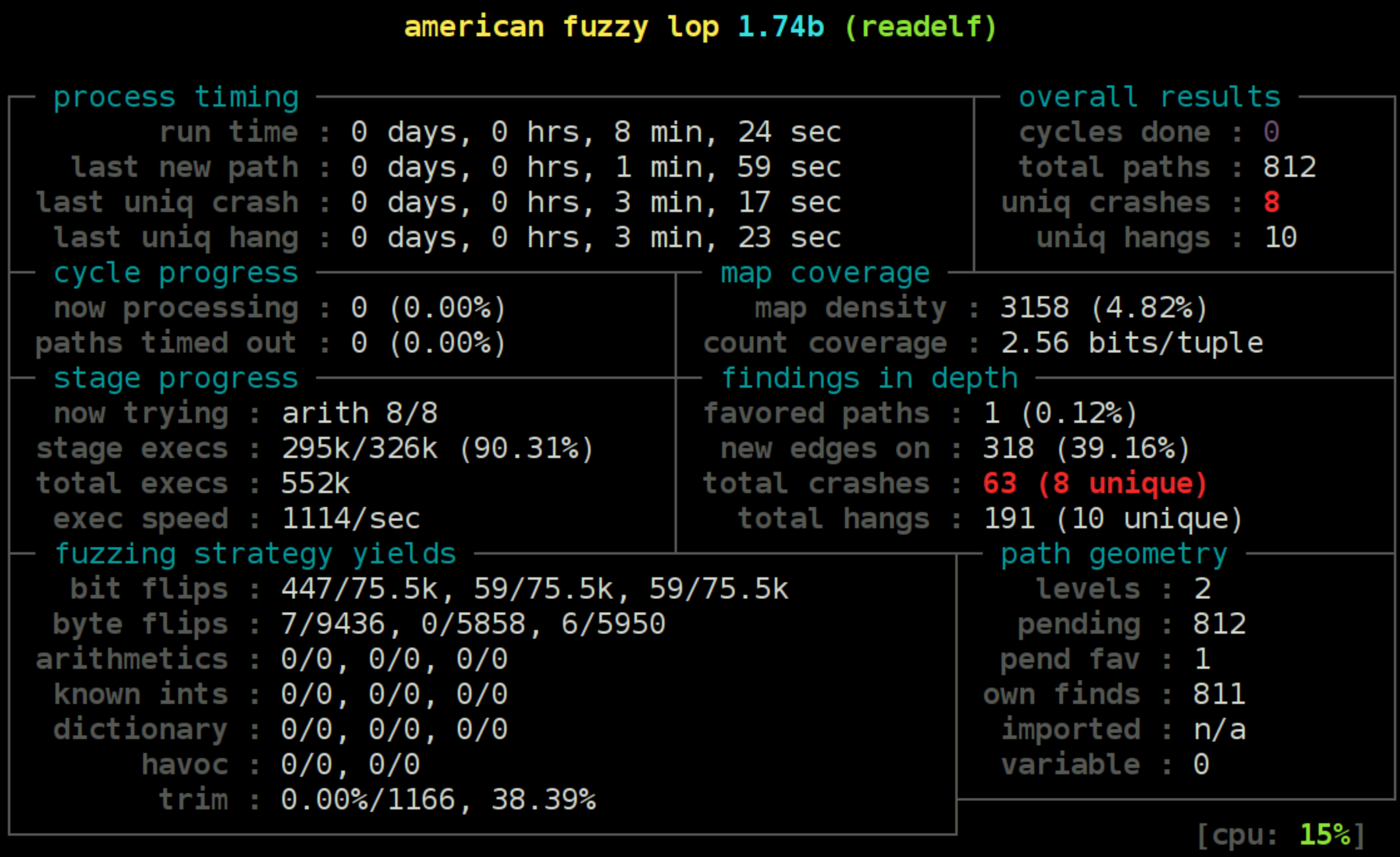


```
  1  [||||||||||||||||      57.4%]   7  [||||||||||||||||      59.5%]   13 [||||||||||||||||      54.7%]   19 [|||||||||||||||||||||100.0%]
  2  [||||||||||||||||||||| 69.1%]   8  [||||||||||||||||||||| 69.6%]   14 [||||||||||||||||||    64.6%]   20 [||||||||||||||         47.4%]
  3  [||||||||||||||||||||| 71.9%]   9  [|||||||||||||||||||   69.4%]   15 [||||||||||||||||||||| 70.9%]   21 [|||||||||||||||||||||| 71.2%]
  4  [|||||||||||||||||||   68.7%]   10 [||||||||||||||||||||||72.5%]   16 [||||||||||||||||||||| 70.4%]   22 [||||||||||||||||||||||72.2%]
  5  [||||||||||||||||||||| 70.9%]   11 [||||||||||||||||||    61.8%]   17 [||||||||||||||||||    62.2%]   23 [|||||||||||||||||||||| 70.5%]
  6  [|||||||||||||||||||   67.1%]   12 [||||||||||||||||||    65.8%]   18 [||||||||||||||||||||| 68.2%]   24 [|||||||||||||||||||||| 69.2%]
  Mem[||||||||                                   365/7982MB]   Tasks: 58, 6 thr; 9 running
  Swp[                                             0/8191MB]   Load average: 19.43 9.00 3.60
                                                               Uptime: 1 day, 03:03:33

  PID USER      PRI  NI  VIRT   RES   SHR S CPU% MEM%   TIME+  Command
 4261 rpisec     20   0 22596  5296  3316 S  0.0  0.1  0:00.05  │     └─ -bash
 5990 rpisec     20   0 26416  4296  2952 R  2.0  0.1  0:03.08  │        └─ htop
14167 root       20   0  103M  6364  5364 S  0.0  0.1  0:00.18  └─ sshd: rpisec [priv]
15984 rpisec     20   0  103M  4136  3104 S  0.0  0.1  0:10.96     └─ sshd: rpisec@pts/2
15985 rpisec     20   0 22672  5480  3376 S  0.0  0.1  0:00.26        └─ -bash
14522 rpisec     20   0  253M 11156  6384 S 52.4  0.1  0:44.50           ├─ python grindr.py ./testcases/lab7A
14549 rpisec     20   0  253M 11156  6384 R 25.2  0.1  0:21.58           │  ├─ python grindr.py ./testcases/lab7A
14548 rpisec     20   0  253M 11156  6384 S 13.9  0.1  0:11.60           │  ├─ python grindr.py ./testcases/lab7A
14547 rpisec     20   0  253M 11156  6384 S  0.0  0.1  0:00.25           │  ├─ python grindr.py ./testcases/lab7A
14546 rpisec     20   0 39644  8248  3476 R 29.2  0.1  0:28.31           │  ├─ python grindr.py ./testcases/lab7A
14545 rpisec     20   0 39660  8272  3476 S 29.8  0.1  0:28.83           │  ├─ python grindr.py ./testcases/lab7A
14544 rpisec     20   0 39872  8500  3476 S 29.8  0.1  0:28.03           │  ├─ python grindr.py ./testcases/lab7A
14543 rpisec     20   0 39872  8636  3476 R 29.2  0.1  0:28.56           │  ├─ python grindr.py ./testcases/lab7A
14542 rpisec     20   0 39640  8268  3476 S 29.2  0.1  0:28.82           │  ├─ python grindr.py ./testcases/lab7A
14541 rpisec     20   0 39864  8496  3476 S 29.2  0.1  0:27.06           │  ├─ python grindr.py ./testcases/lab7A
14540 rpisec     20   0 39868  8600  3476 S 28.5  0.1  0:26.96           │  ├─ python grindr.py ./testcases/lab7A
14539 rpisec     20   0 39872  8616  3476 S 29.8  0.1  0:29.12           │  ├─ python grindr.py ./testcases/lab7A
14538 rpisec     20   0 39860  8644  3476 S 27.9  0.1  0:27.51           │  ├─ python grindr.py ./testcases/lab7A
14537 rpisec     20   0 43532 13136  3476 S 30.5  0.2  0:29.94           │  ├─ python grindr.py ./testcases/lab7A
14536 rpisec     20   0 39864  8620  3476 S 28.5  0.1  0:28.26           │  ├─ python grindr.py ./testcases/lab7A
14535 rpisec     20   0 39864  8624  3476 S 29.2  0.1  0:28.33           │  ├─ python grindr.py ./testcases/lab7A
14534 rpisec     20   0 39864  8632  3476 R 30.5  0.1  0:27.10           │  ├─ python grindr.py ./testcases/lab7A
14533 rpisec     20   0 39868  8608  3476 S 29.2  0.1  0:27.28           │  ├─ python grindr.py ./testcases/lab7A
14532 rpisec     20   0 39860  8596  3476 S 27.9  0.1  0:27.73           │  ├─ python grindr.py ./testcases/lab7A
14531 rpisec     20   0 39856  8464  3476 S 30.5  0.1  0:28.33           │  ├─ python grindr.py ./testcases/lab7A
14530 rpisec     20   0 39856  8588  3476 S 29.8  0.1  0:28.78           │  ├─ python grindr.py ./testcases/lab7A
14529 rpisec     20   0 39860  8580  3476 S 29.8  0.1  0:27.75           │  ├─ python grindr.py ./testcases/lab7A
14528 rpisec     20   0 39860  8592  3476 S 28.5  0.1  0:27.12           │  ├─ python grindr.py ./testcases/lab7A
14527 rpisec     20   0 39856  8448  3476 S 29.2  0.1  0:29.35           │  ├─ python grindr.py ./testcases/lab7A
14526 rpisec     20   0 39632  8216  3476 S 29.2  0.1  0:28.51           │  ├─ python grindr.py ./testcases/lab7A
14525 rpisec     20   0 39856  8572  3468 S 27.9  0.1  0:27.31           │  ├─ python grindr.py ./testcases/lab7A
14524 rpisec     20   0 39852  8588  3476 R 30.5  0.1  0:27.10           │  ├─ python grindr.py ./testcases/lab7A
14523 rpisec     20   0 39856  8568  3468 R 29.2  0.1  0:28.76           │  └─ python grindr.py ./testcases/lab7A
14521 rpisec     20   0 49932  7876  5228 T  0.0  0.1  0:00.13           └─ vim grindr.py
 1257 root       20   0 15824  2012  1860 S  0.0  0.0  0:00.00  ├─ /sbin/getty -8 38400 tty6
 1255 root       20   0 15824  2108  1944 S  0.0  0.0  0:00.00  ├─ /sbin/getty -8 38400 tty3
F1Help  F2Setup F3SearchF4FilterF5Tree  F6SortByF7Nice -F8Nice +F9Kill  F10Quit
```

American Fuzzy Lop (AFL)

- A ‘security-oriented’ fuzzer that inserts and utilizes instrumentation that it inserts at compile time
  - Requires source code to be super effective

American Fuzzy Lop (AFL)



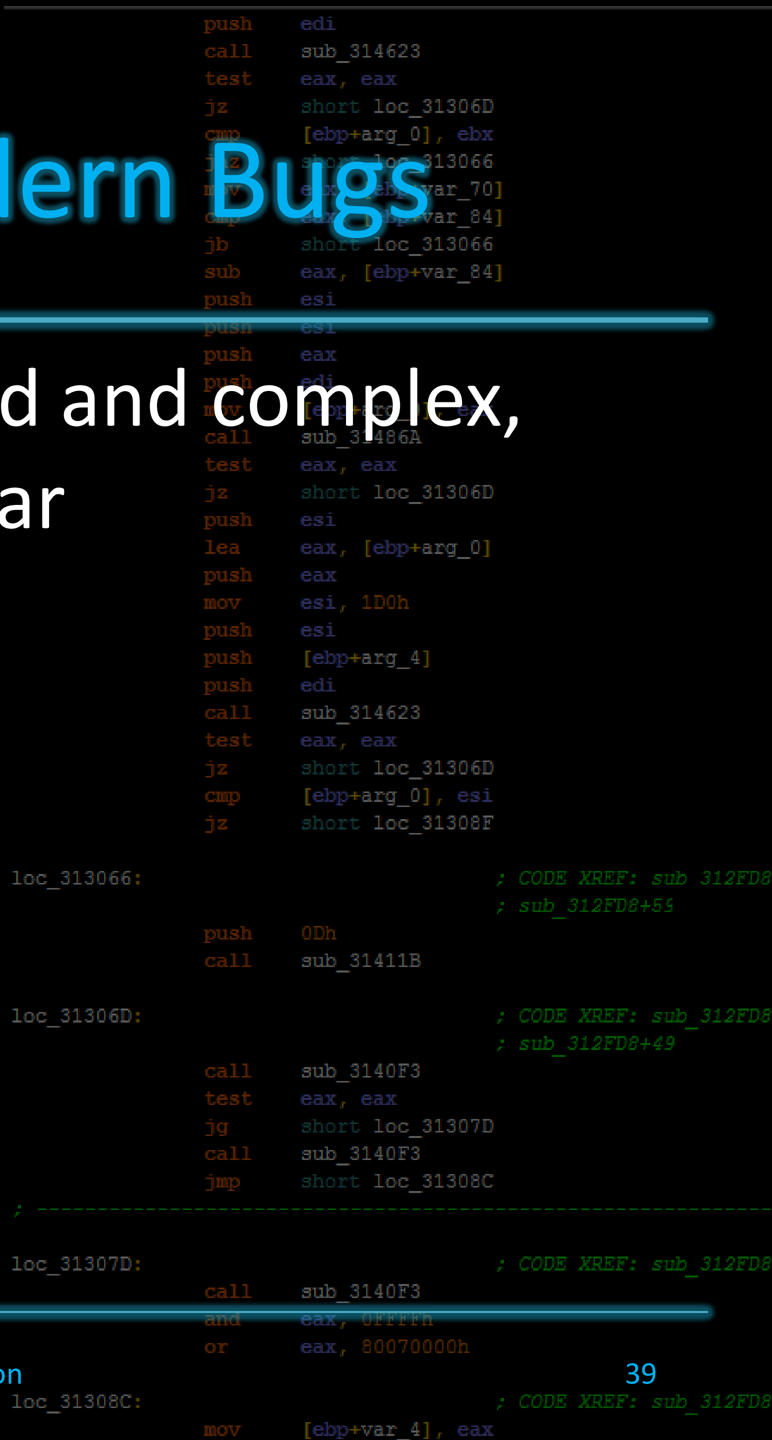
```
                 american fuzzy lop 1.74b (readelf)

┌─ process timing ─────────────────────────────────────┬─ overall results ─────┐
│        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 : 8      │
│  last uniq hang : 0 days, 0 hrs, 3 min, 23 sec        │   uniq hangs : 10     │
├─ cycle progress ────────────────────┬─ map coverage ─┴───────────────────────┤
│  now processing : 0 (0.00%)         │    map density : 3158 (4.82%)          │
│ paths timed out : 0 (0.00%)         │ count coverage : 2.56 bits/tuple       │
├─ stage progress ────────────────────┼─ findings in depth ────────────────────┤
│  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 : 63 (8 unique)          │
│  exec speed : 1114/sec              │   total hangs : 191 (10 unique)        │
├─ fuzzing strategy yields ───────────┴───────────────┬─ path geometry ────────┤
│   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%]
```

American Fuzzy Lop (AFL)

- A ‘security-oriented’ fuzzer that inserts and utilizes instrumentation that it inserts at compile time
  - Requires target source code to be super effective
- Great for file format fuzzing!
  - Generally not that useful for CTF fuzzing :/
- http://lcamtuf.coredump.cx/afl/

Fundamentals of Modern Bugs

- As the bugs get more refined and complex, fuzzing will only take us so far

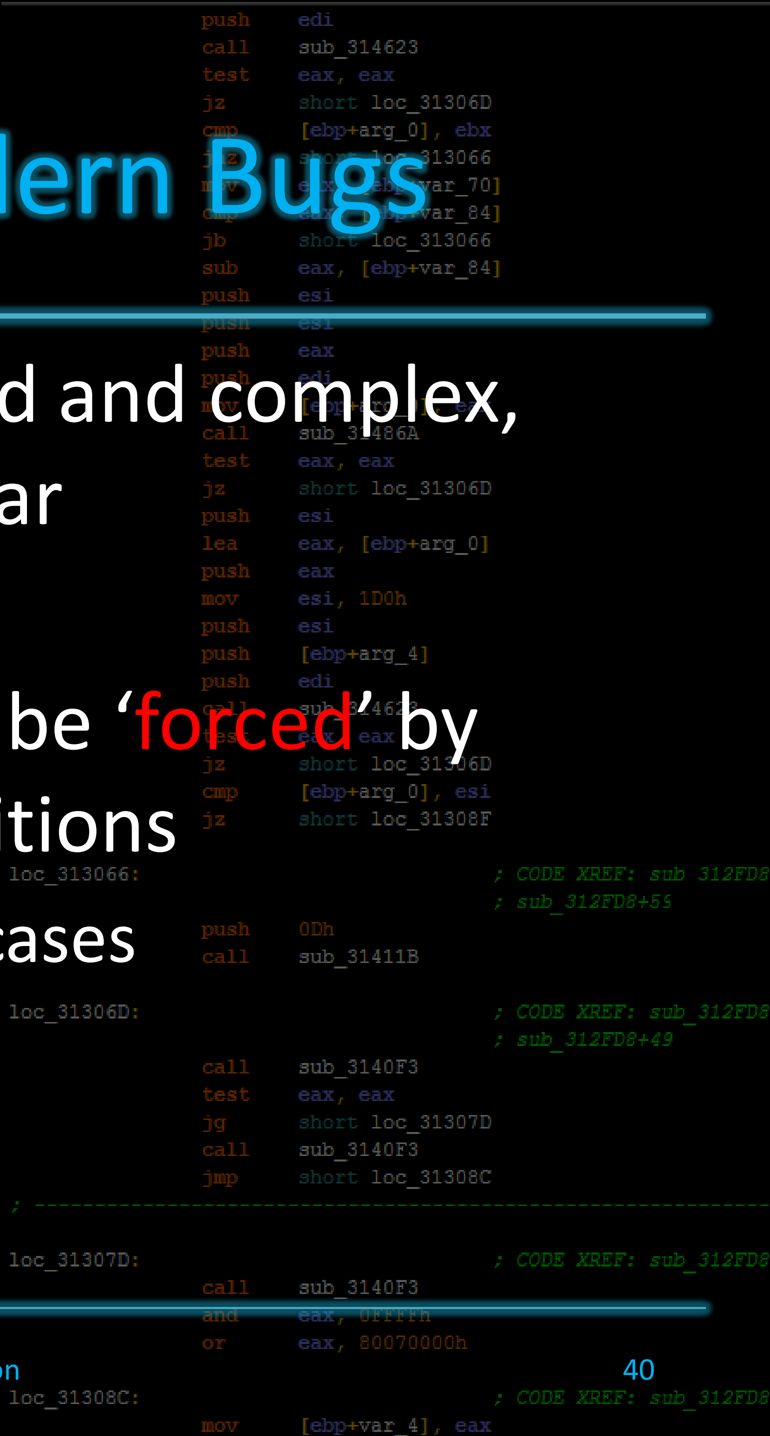


```
                push    edi
                call    sub_314623
                test    eax, eax
                jz      short loc_31306D
                cmp     [ebp+arg_0], ebx
                jz      short loc_313066
                mov     eax, [ebp+var_70]
                cmp     eax, [ebp+var_84]
                jb      short loc_313066
                sub     eax, [ebp+var_84]
                push    esi
                push    esi
                push    eax
                push    edi
                mov     [ebp+arg_0], eax
                call    sub_31486A
                test    eax, eax
                jz      short loc_31306D
                push    esi
                lea     eax, [ebp+arg_0]
                push    eax
                mov     esi, 1D0h
                push    esi
                push    [ebp+arg_4]
                push    edi
                call    sub_314623
                test    eax, eax
                jz      short loc_31306D
                cmp     [ebp+arg_0], esi
                jz      short loc_31308F

loc_313066:                             ; CODE XREF: sub_312FD8
                                        ; sub_312FD8+55
                push    0Dh
                call    sub_31411B

loc_31306D:                             ; CODE XREF: sub_312FD8
                                        ; sub_312FD8+49
                call    sub_3140F3
                test    eax, eax
                jg      short loc_31307D
                call    sub_3140F3
                jmp     short loc_31308C
; ---------------------------------------------------------------

loc_31307D:                             ; CODE XREF: sub_312FD8
                call    sub_3140F3
                and     eax, 0FFFFh
                or      eax, 80070000h

loc_31308C:                             ; CODE XREF: sub_312FD8
                mov     [ebp+var_4], eax
```

Fundamentals of Modern Bugs

- As the bugs get more refined and complex, fuzzing will only take us so far

- Many modern bugs have to be ‘forced’ by requiring very specific conditions
  - like some sort of crazy edge cases



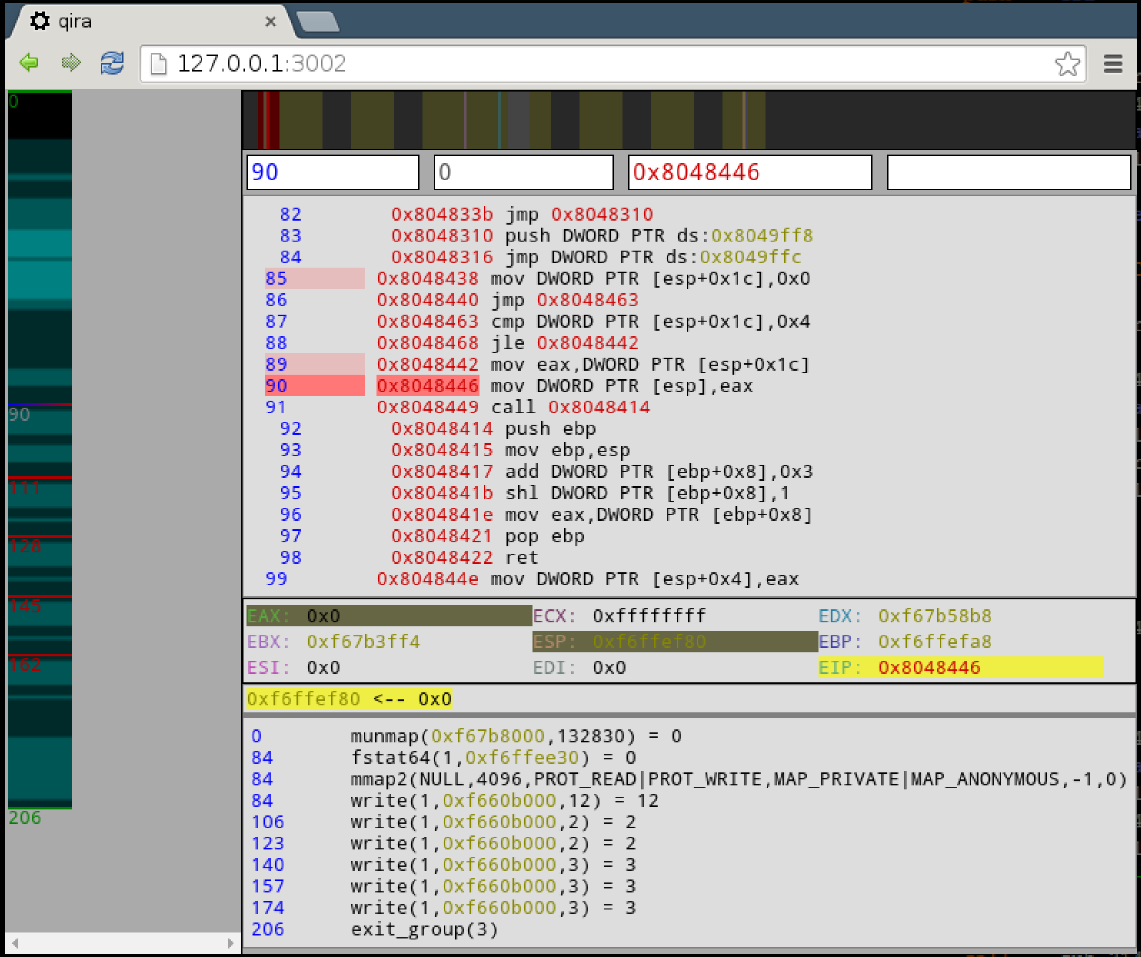
```
                push    edi
                call    sub_314623
                test    eax, eax
                jz      short loc_31306D
                cmp     [ebp+arg_0], ebx
                jnz     short loc_313066
                mov     eax, [ebp+var_70]
                cmp     eax, [ebp+var_84]
                jb      short loc_313066
                sub     eax, [ebp+var_84]
                push    esi
                push    esi
                push    eax
                push    edi
                mov     [ebp+arg_0], eax
                call    sub_31486A
                test    eax, eax
                jz      short loc_31306D
                push    esi
                lea     eax, [ebp+arg_0]
                push    eax
                mov     esi, 1D0h
                push    esi
                push    [ebp+arg_4]
                push    edi
                call    sub_314623
                test    eax, eax
                jz      short loc_31306D
                cmp     [ebp+arg_0], esi
                jz      short loc_31308F

loc_313066:                             ; CODE XREF: sub_312FD8
                                        ; sub_312FD8+55
                push    0Dh
                call    sub_31411B

loc_31306D:                             ; CODE XREF: sub_312FD8
                                        ; sub_312FD8+49
                call    sub_3140F3
                test    eax, eax
                jg      short loc_31307D
                call    sub_3140F3
                jmp     short loc_31308C
; ---------------------------------------------------------------------------

loc_31307D:                             ; CODE XREF: sub_312FD8
                call    sub_3140F3
                and     eax, 0FFFFh
                or      eax, 80070000h

loc_31308C:                             ; CODE XREF: sub_312FD8
                mov     [ebp+var_4], eax
```

QIRA

- A ‘timeless debugger’ – By GeoHot
  - Observe a binary at any point of its execution state for a given input
  - You can move forwards and backwards in time

```
                push    edi
                call    sub_314623
                test    eax, eax
                jz      short loc_31306D
                cmp     [ebp+arg_0], ebx
                jnz     short loc_313066
                mov     eax, [ebp+var_70]
                cmp     eax, [ebp+var_84]
                jb      short loc_313066
                sub     eax, [ebp+var_84]
                push    esi
                push    esi
                push    eax
                push    edi
                mov     [ebp+arg_0], eax
                call    sub_31486A
                test    eax, eax
                jz      short loc_31306D
                push    esi
                lea     eax, [ebp+arg_0]
                push    eax
                mov     esi, 1D0h
                push    esi
                push    [ebp+arg_4]
                push    edi
                call    sub_314623
                test    eax, eax
                jz      short loc_31306D
                cmp     [ebp+arg_0], esi
                jz      short loc_31308F

loc_313066:                             ; CODE XREF: sub_312FD8
                                        ; sub_312FD8+55
                push    0Dh
                call    sub_31411B

loc_31306D:                             ; CODE XREF: sub_312FD8
                                        ; sub_312FD8+49
                call    sub_3140F3
                test    eax, eax
                jg      short loc_31307D
                call    sub_3140F3
                jmp     short loc_31308C
; ---------------------------------------------------------------

loc_31307D:                             ; CODE XREF: sub_312FD8
                call    sub_3140F3
                and     eax, 0FFFFh
                or      eax, 80070000h

loc_31308C:                             ; CODE XREF: sub_312FD8
                mov     [ebp+var_4], eax
```

QIRA



```
90        0        0x8048446

  82    0x804833b jmp 0x8048310
  83    0x8048310 push DWORD PTR ds:0x8049ff8
  84    0x8048316 jmp DWORD PTR ds:0x8049ffc
 85     0x8048438 mov DWORD PTR [esp+0x1c],0x0
 86     0x8048440 jmp 0x8048463
 87     0x8048463 cmp DWORD PTR [esp+0x1c],0x4
 88     0x8048468 jle 0x8048442
 89     0x8048442 mov eax,DWORD PTR [esp+0x1c]
 90     0x8048446 mov DWORD PTR [esp],eax
 91     0x8048449 call 0x8048414
  92    0x8048414 push ebp
  93    0x8048415 mov ebp,esp
  94    0x8048417 add DWORD PTR [ebp+0x8],0x3
  95    0x804841b shl DWORD PTR [ebp+0x8],1
  96    0x804841e mov eax,DWORD PTR [ebp+0x8]
  97    0x8048421 pop ebp
  98    0x8048422 ret
 99     0x804844e mov DWORD PTR [esp+0x4],eax

EAX:  0x0                ECX:  0xffffffff         EDX:  0xf67b58b8
EBX:  0xf67b3ff4         ESP:  0xf6ffef80         EBP:  0xf6ffefa8
ESI:  0x0                EDI:  0x0                EIP:  0x8048446

0xf6ffef80 <-- 0x0

0       munmap(0xf67b8000,132830) = 0
84      fstat64(1,0xf6ffee30) = 0
84      mmap2(NULL,4096,PROT_READ|PROT_WRITE,MAP_PRIVATE|MAP_ANONYMOUS,-1,0)
84      write(1,0xf660b000,12) = 12
106     write(1,0xf660b000,2) = 2
123     write(1,0xf660b000,2) = 2
140     write(1,0xf660b000,3) = 3
157     write(1,0xf660b000,3) = 3
174     write(1,0xf660b000,3) = 3
206     exit_group(3)
```

QIRA

- A ‘timeless debugger’ – By GeoHot
  - Observe a binary at any point of its execution state for a given input
  - You can move forwards and backwards in time

- Super basic taint sort of functionality
  - Helps visualize r/w of specific memory addresses

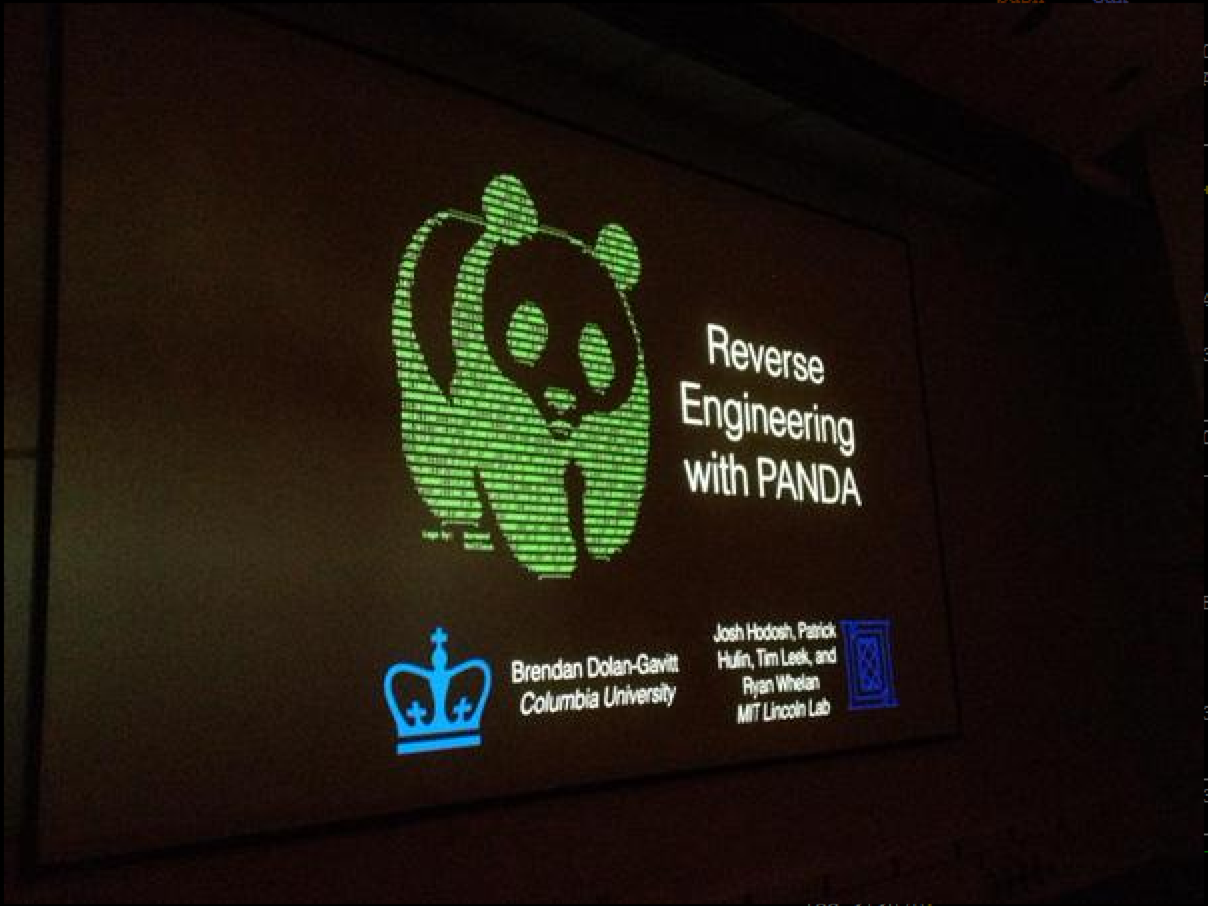
- http://qira.me/

PANDA

- An ‘open-source Platform for Architecture-Neutral Dynamic Analysis’ – By MITLL

```
                push    edi
                call    sub_314623
                test    eax, eax
                jz      short loc_31306D
                cmp     [ebp+arg_0], ebx
                jnz     short loc_313066
                mov     eax, [ebp+var_70]
                cmp     eax, [ebp+var_84]
                jb      short loc_313066
                sub     eax, [ebp+var_84]
                push    esi
                push    esi
                push    eax
                push    edi
                mov     [ebp+arg_0], eax
                call    sub_31486A
                test    eax, eax
                jz      short loc_31306D
                push    esi
                lea     eax, [ebp+arg_0]
                push    eax
                mov     esi, 1D0h
                push    esi
                push    [ebp+arg_4]
                push    edi
                call    sub_314623
                test    eax, eax
                jz      short loc_31306D
                cmp     [ebp+arg_0], esi
                jz      short loc_31308F

loc_313066:                             ; CODE XREF: sub_312FD8
                                        ; sub_312FD8+55
                push    0Dh
                call    sub_31411B

loc_31306D:                             ; CODE XREF: sub_312FD8
                                        ; sub_312FD8+49
                call    sub_3140F3
                test    eax, eax
                jg      short loc_31307D
                call    sub_3140F3
                jmp     short loc_31308C
; ---------------------------------------------------------------

loc_31307D:                             ; CODE XREF: sub_312FD8
                call    sub_3140F3
                and     eax, 0FFFFh
                or      eax, 80070000h

loc_31308C:                             ; CODE XREF: sub_312FD8
                mov     [ebp+var_4], eax
```

PANDA



Reverse Engineering with PANDA

Brendan Dolan-Gavitt
Columbia University

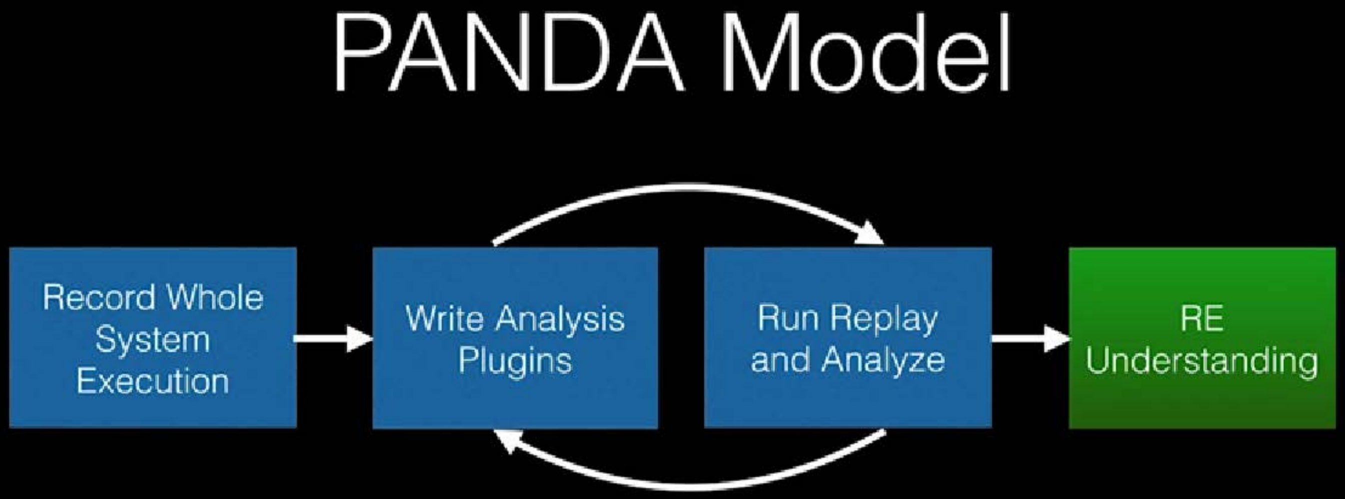
Josh Hodosh, Patrick Hulin, Tim Leek, and Ryan Whelan
MIT Lincoln Lab

PANDA

- An ‘open-source Platform for Architecture-Neutral Dynamic Analysis’ – By MITLL

- Built on top of QEMU, allows instrumentation, analysis, and replay of an entire system

PANDA



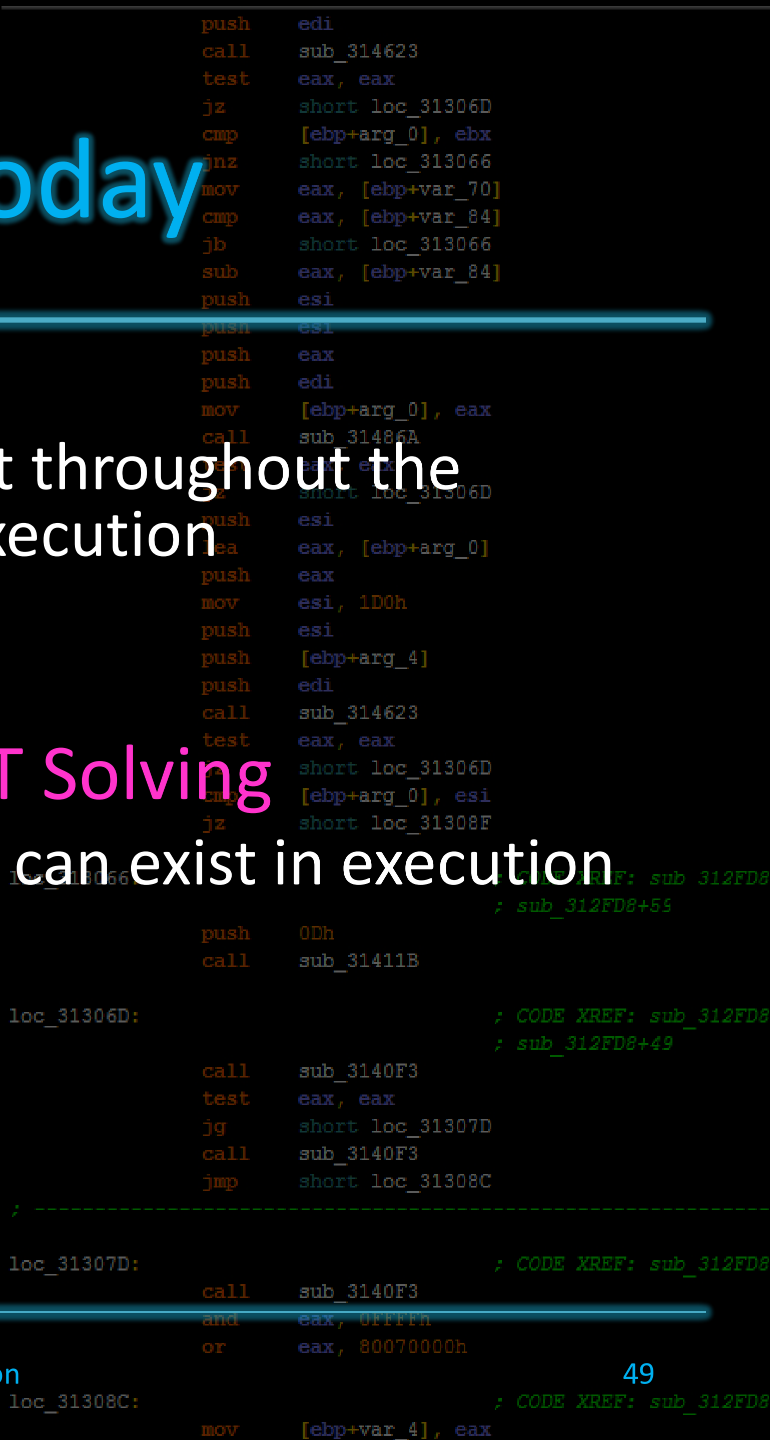
PANDA Model

PANDA

- An ‘open-source Platform for Architecture-Neutral Dynamic Analysis’ – By MITLL
- Built on top of QEMU, allows instrumentation, analysis, and replay of an entire system
- Awesome plugin infrastructure
  - Utilizes LLVM Intermediate Representation to make one size fits all (CPU’s) analysis plugins
- https://github.com/moyix/panda

Advanced Concepts Today

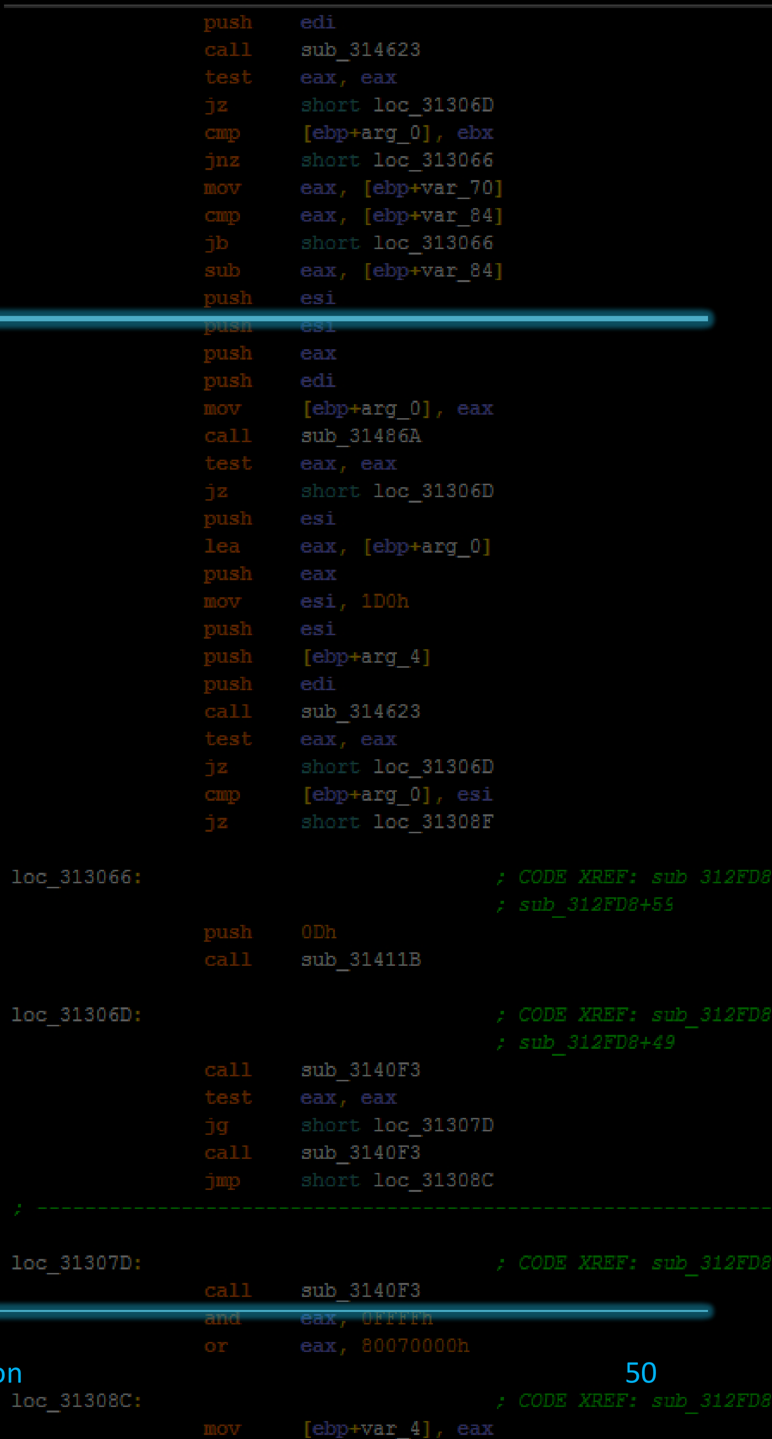
- Taint Analysis
  - Tracing the impact of user input throughout the binary, and how it influences execution
  - PANDA, QIRA
- Symbolic Execution + SAT/SMT Solving
  - Proving that specific conditions can exist in execution to manifest difficult bugs
  - Z3, SMT-LIB
- Machine Learning



```
                push    edi
                call    sub_314623
                test    eax, eax
                jz      short loc_31306D
                cmp     [ebp+arg_0], ebx
                jnz     short loc_313066
                mov     eax, [ebp+var_70]
                cmp     eax, [ebp+var_84]
                jb      short loc_313066
                sub     eax, [ebp+var_84]
                push    esi
                push    esi
                push    eax
                push    edi
                mov     [ebp+arg_0], eax
                call    sub_31486A
                test    eax, eax
                jz      short loc_31306D
                push    esi
                lea     eax, [ebp+arg_0]
                push    eax
                mov     esi, 1D0h
                push    esi
                push    [ebp+arg_4]
                push    edi
                call    sub_314623
                test    eax, eax
                jz      short loc_31306D
                cmp     [ebp+arg_0], esi
                jz      short loc_31308F

loc_313066:                             ; CODE XREF: sub_312FD8
                                        ; sub_312FD8+55
                push    0Dh
                call    sub_31411B

loc_31306D:                             ; CODE XREF: sub_312FD8
                                        ; sub_312FD8+49
                call    sub_3140F3
                test    eax, eax
                jg      short loc_31307D
                call    sub_3140F3
                jmp     short loc_31308C
; ---------------------------------------------------------------

loc_31307D:                             ; CODE XREF: sub_312FD8
                call    sub_3140F3
                and     eax, 0FFFFh
                or      eax, 80070000h

loc_31308C:                             ; CODE XREF: sub_312FD8
                mov     [ebp+var_4], eax
```

Lecture Overview

- Security
  - – Security Today
  - – Security Tomorrow
- Exploitation
  - – Exploitation Today
  - – Exploitation Tomorrow



```
                push    edi
                call    sub_314623
                test    eax, eax
                jz      short loc_31306D
                cmp     [ebp+arg_0], ebx
                jnz     short loc_313066
                mov     eax, [ebp+var_70]
                cmp     eax, [ebp+var_84]
                jb      short loc_313066
                sub     eax, [ebp+var_84]
                push    esi
                push    esi
                push    eax
                push    edi
                mov     [ebp+arg_0], eax
                call    sub_31486A
                test    eax, eax
                jz      short loc_31306D
                push    esi
                lea     eax, [ebp+arg_0]
                push    eax
                mov     esi, 1D0h
                push    esi
                push    [ebp+arg_4]
                push    edi
                call    sub_314623
                test    eax, eax
                jz      short loc_31306D
                cmp     [ebp+arg_0], esi
                jz      short loc_31308F

loc_313066:                             ; CODE XREF: sub_312FD8
                                        ; sub_312FD8+55
                push    0Dh
                call    sub_31411B

loc_31306D:                             ; CODE XREF: sub_312FD8
                                        ; sub_312FD8+49
                call    sub_3140F3
                test    eax, eax
                jg      short loc_31307D
                call    sub_3140F3
                jmp     short loc_31308C
; ---------------------------------------------------------------

loc_31307D:                             ; CODE XREF: sub_312FD8
                call    sub_3140F3
                and     eax, 0FFFFh
                or      eax, 80070000h

loc_31308C:                             ; CODE XREF: sub_312FD8
                mov     [ebp+var_4], eax
```

DARPA’s Cyber Grand Challenge

DARPA’s Cyber Grand Challenge

"DARPA’s Cyber Grand Challenge"

https://www.youtube.com/watch?v=OVV_k73z3E0

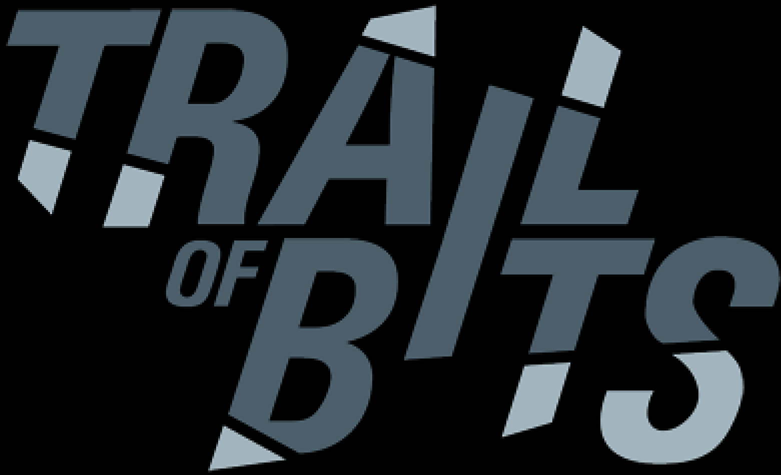
About CGC

- A challenge set forth by DARPA

- Can we develop a completely autonomous system that is capable of…
  - finding vulnerabilities (whitebox and blackbox)
  - patching said vulnerabilities
  - writing exploits for said vulnerabilities

- http://www.darpa.mil/cybergrandchallenge/

Some CGC Competitors



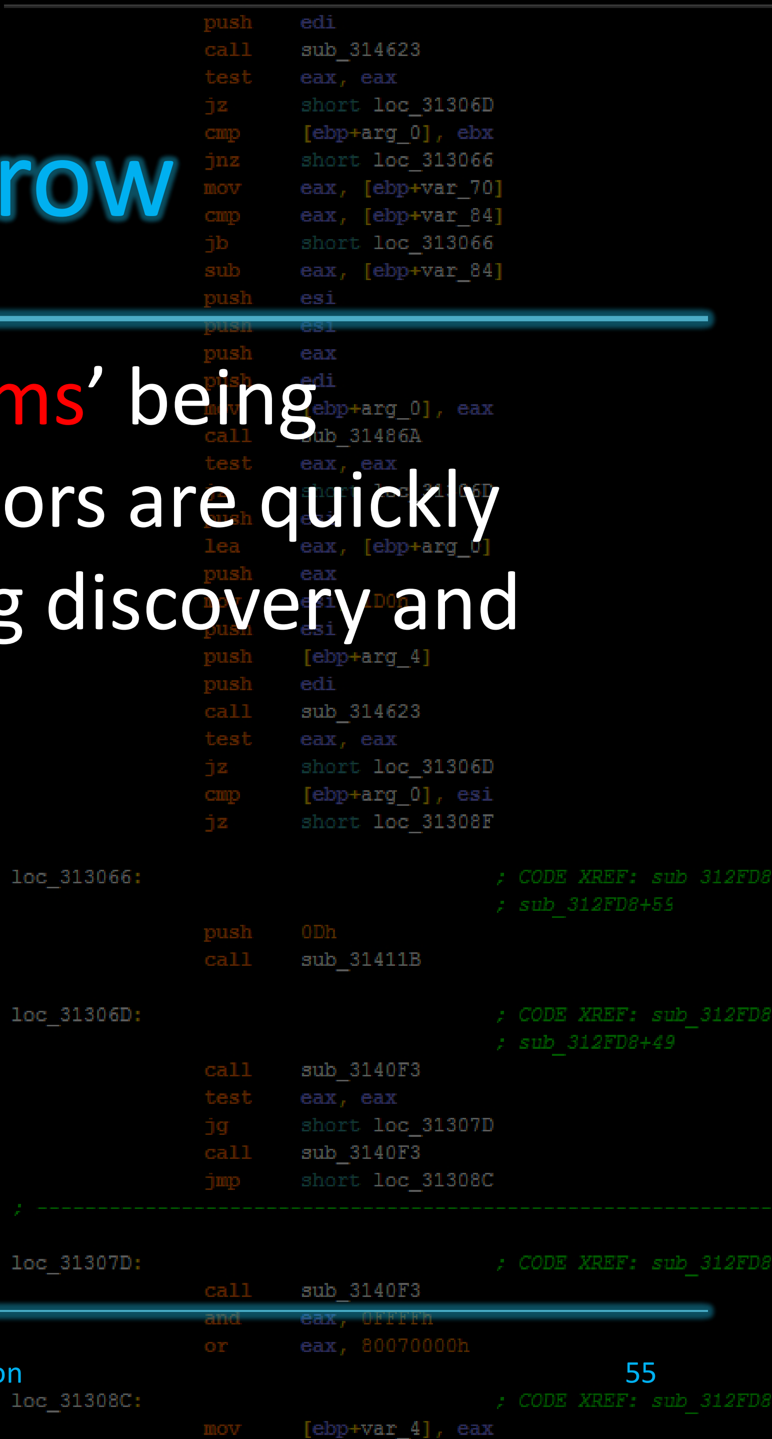
SRI
International

LINCOLN LABORATORY
MASSACHUSETTS INSTITUTE OF TECHNOLOGY

Berkeley
UNIVERSITY OF CALIFORNIA

Exploitation of Tomorrow

- The ‘Cyber Reasoning Systems’ being developed by CGC competitors are quickly pushing the envelope of bug discovery and exploitation



```
                push    edi
                call    sub_314623
                test    eax, eax
                jz      short loc_31306D
                cmp     [ebp+arg_0], ebx
                jnz     short loc_313066
                mov     eax, [ebp+var_70]
                cmp     eax, [ebp+var_84]
                jb      short loc_313066
                sub     eax, [ebp+var_84]
                push    esi
                push    esi
                push    eax
                push    edi
                mov     [ebp+arg_0], eax
                call    sub_31486A
                test    eax, eax
                jz      short loc_31306D
                push    esi
                lea     eax, [ebp+arg_0]
                push    eax
                mov     esi, 1D0h
                push    esi
                push    [ebp+arg_4]
                push    edi
                call    sub_314623
                test    eax, eax
                jz      short loc_31306D
                cmp     [ebp+arg_0], esi
                jz      short loc_31308F

loc_313066:                             ; CODE XREF: sub_312FD8
                                        ; sub_312FD8+55
                push    0Dh
                call    sub_31411B

loc_31306D:                             ; CODE XREF: sub_312FD8
                                        ; sub_312FD8+49
                call    sub_3140F3
                test    eax, eax
                jg      short loc_31307D
                call    sub_3140F3
                jmp     short loc_31308C
; ---------------------------------------------------------------------------

loc_31307D:                             ; CODE XREF: sub_312FD8
                call    sub_3140F3
                and     eax, 0FFFFh
                or      eax, 80070000h

loc_31308C:                             ; CODE XREF: sub_312FD8
                mov     [ebp+var_4], eax
```

Exploitation of Tomorrow

- The ‘Cyber Reasoning Systems’ being developed by CGC competitors are quickly pushing the envelope of bug discovery and exploitation

- The technology behind them is likely to be some smart fuzzers guided by taint analysis, constraint solvers, and more