



So, like, l'm in Majuro...







DRINKING...



Then Dave is all...







CONFERENCE APRIL 16-17, FONTAINEBLEALTH





LONG STORY SHORT...

FUZZING OSX

AT SCALE

I PROPOSED...

- 8 weeks
- Write better instrumentation than CrashWrangler
- Run OSX on commodity hardware
- Run ''shiny new tools'' like AFL
- Centralised C&C

GROUND RULES

- This was new work, didn't know how it would go
- I knew ~nothing about OSX
- I never release bugs in fuzzing talks
- Userland only

FUZZING IS A PROCESS!

- I. Acquire Knowledge
- 2. Instrumentation
- 3. Delivery
- 4. Generation
- 5. Scale



ACQUIRE KNOWLEDGE







- Processes Tasks (Threads Threads)
- IPC 🖸 Mach Ports
- System Calls Traps
- Libraries 🖸 Frameworks





- gcc 🖸 clang
- Idd, objdump, many others D otool



http://uninformed.org/?v=4&a=3&t=sumry

http://web.mit.edu/darwin/src/modules/xnu/osfmk/man/

https://github.com/shoumikhin/Mach-O-Hook

ACQUIRE KNOWLEDGE



INSTRUMENTATION



INSTRUMENTATION

- Mach Exception Ports?
- "Normal" Unix tools?
- CrashWrangler?
- LLDB C++ / SWIG API (
- GDB?



- exc_handler ~ I 300LOC ObjC
- CrashLog.rb ~800LOC (reimplements half of it?)
- Moar Ruby scripts to do bucketing
- bash scripts for instrumentation

CRASHWRANGLER 😂

```
14
   if [ -n "${CW TIMEOUT+x}" ]; then
15
    echo Using timeout $CW_TIMEOUT
16
17
    else
        CW TIMEOUT=3
18
19
   fi
20
21
    ./exc_handler "/Applications/Preview.app/Contents/MacOS/Preview" "$1" &
    EXCPID=$! # get the PID for the last added background process
22
23
24
   ruby -e "begin ; sleep $CW_TIMEOUT; Process.kill(\"USR1\", $EXCPID) if no
25
26
    # wait for exc handler to exit, and get exit value
27
   wait $EXCPID
28
   EXIT_VALUE=$?
29
30
   # return exit value of exc handler
31
    exit $EXIT_VALUE
```



- About 30 C tests to repro common crashes
- Includes OSX specific faults
 - ObjC, CoreFoundation ...

GDB EXPLOITABLE

- Initially developed at CERT
- https://github.com/jfoote/exploitable
- Runs as a GDB plugin
- Another ~20 tests



LLDB API

- Actually really good! And has samples!
- http://lldb.llvm.org/python_reference/index.html
- Complete, unsweetened, 2 wrapper
- Tools runs AS a debugger, not IN a debugger

EXPLOITABEN*

- Steals design from Exploitable
- Steals OSX heuristics from CrashWrangler
- Steals tests from both
- More or less ground-up rewrite (2)

EXPLOITABEN*

- Uses indicators, not classifications
- Tweaks to hash bucketing
- Assorted Frills
 - timeouts, attach-wait, env, command triggers ...

INDICATORS

- Memory patterns linked to UAF etc (0xbadbeef)
- ~50 suspicious OSX stack functions
- Access types: read, write, exec, recursion
- Heuristics on \$pc, \$sp
- Flags block movs

#define SIZE (1 << 30)
int main() {</pre>

char buf[SIZE]; char c = buf[0];

+

EXC BAD ACCESS (code=1, StopDesc: address=0x7fff1fbffe30) AvNearNull: False True AvNearSP: BadBeef: Faise Access Type: read **Registers:** d1=0x00000000000000078 BlockMov: False Weird PC: False True Weird SP: Suspicious Funcs: Illegal Insn: False Huge Stack: False

#include <string.h>
int main(int argc, char** argv)
{
 char buf[16];
 memset(buf, 'A', 65);
}

StopDesc: signal SIGABRT AvNearNull: False AvNearSP: False BadBeef: False Access Type: <not an access violation> **Registers:** BlockMov: False Weird PC: False Weird SP: False stack chk fail Suspicious Funcs: Illegal Insn: False Huge Stack: False

FRANCIS

- Simple Go package to parse exploitaben reports
- Seems pretty reliable so far
- Used as the OSX plugin for my afl-triage tool
 - Multicore, -every mode, Cache DB ...

AFL-TRIAGE DEMO

TERRY

- Very slow/lame fuzzer in Go
- Forks a radamsa server for Generation
- Invokes target under exploitaben
- Finds bugs 词
- <u>https://github.com/bnagy/terry</u>


AFL (FORESHADOWING)

- AFL has its own Instrumentation
- fork()server + signal handler
- polices memory limits and timeouts

SUMMARY

- Good enough instrumentation
- MUCH better crash triage than I'm used to
- OMGSYMBOLS!!! SOURCE CODE!!
- No dialog clickies etc
- No CPU monitoring, timeouts only

HOW WE DOIN?

- I'm in Palau
- Not drinking
- ~4 weeks into the project
- Dave is sending me motivational messages

MOTIVATION

...



Ben Nagy @rantyben · Feb 5 #febfast holy crap eating out is super cheap now o_0

1 + 2



•



@rantyben Why are you eating when you have os x fuzzing to be doing for infiltrate????????...

★ ti ★ ···

DELIVERY



PADDLE FASTER I HEAR BANJOS

AFL



http://prelkia.deviantart.com/art/Furious-Rabbit-157796019

A FRACTAL OF GOOD DESIGN

AFL

- "American Fuzzy Lop" (it's a rabbit)
- Revolutionary Fuzzing Tools (in historical order)
 - SPIKE
 - AFL

SRSLY?

- Compile-time instrumentation (or dynamic)
- Collect Coverage
- Dumb mutation
- Evolve files that create paths

ALL ABOUT EXECUTION!

- Coverage "bitmap"
- Forkserver
- Mutation algorithm curation
- Tokens discovered / user supplied

COVERAGE BITMAP

cur = <COMPILE_TIME_RANDOM>;
shared_mem[cur ^ prev]++;
prev = cur >> 1;

COVERAGE BITMAP



ALL ABOUT EXECUTION!

- Effector maps
- Greedy time allocation
- Support tools
 - cmin, tmin, showmap, peruvian-were-rabbit

CMIN

- Shrink corpus to approximate minimum set
- Prefers smaller files

TMIN

- Shrink one test to minimum size and clean
- Remove blocks, see if we lose coverage
- Alphabet normalisation
- Kinda slow (obviously)

PERUVIAN WERE-RABBIT

- Take one or more crashing inputs
- Run "normal" AFL, discard non-crashes
- Great for exploring promising null derefs etc

BUT... BUT... SAGE!

- AFL makes "generation" fuzzing practical
- For virtually any target*
- With no modifications*
- With no input limits*
- With no need to write code*

AFL WEAKNESSES

- Windows / Closed source in general
- Scale / Triage
- Complex Formats / Grammars*
- Effectiveness is exponential

Fuzzers alive : 47
Total run time : 119 days, 3 hours
Total execs : 22 million
Cumulative speed : 102 execs/sec
Pending paths : 27678 faves, 292486 total
Pending per fuzzer : 588 faves, 6223 total
Crashes found : 3 locally unique

Fuzzers alive : 47 Total run time : 875 days, 4 hours Total execs : 167 million Cumulative speed : 104 execs/sec Pending paths : 41987 faves, 800020 total Pending per fuzzer : 893 faves, 17021 total Crashes found : 3647 locally unique

TIME TO EXPERIMENT



TARGET: PREVIEW+PDF

- Day I: AFL supports qemu DBI for coverage
- Day 3: Except it doesn't work on OSX 😰
- Day 14: Even if it did it would be too slow 🗊

CORPUS DRIVEN FUZZING

- Great Corpora 🖸 Much Bug
- How to acquire?

BUILDING CORPORA

- "Prospector"
 - Download files from Internet
 - Trace for coverage
 - Select minimum files with maximum cover

PROSPECTOR

- Many people have had this idea
- I got it from Charlie
- Peach had a minset tool (but it was broken)
- AFAIK mine is the only public, "working" code

PROSPECTOR ISSUES

- Real World files are not a good way to get cov
- Scraping is hard / slow / annoying
- Files are bloated with useless user data

EXISTING CORPORA

• Great if you can get them!

• ?

- http://acroeng.adobe.com/wp/?page_id=10
- https://github.com/corkami/pocs/tree/master/pdf
- https://code.google.com/p/imagetestsuite/

AFL CORPORA

- Generate against fast targets
- OSS parsers different versions, options
- Minimise with cmin / tmin
- (Complex Grammars need help)



SYNTHESIS

CDFVS PDF

- AFL side wrote a lexer to tokenise PDF
- https://github.com/bnagy/pdflex
 - Extracted ~1500 tokens
 - Curated by hand

CDFVS PDF

- Bugged lcamtuf to add a "fixup" feature
- AFL can now invoke a custom .so for each test
- Wrote a shim .so that talks to a unix socket
- <u>https://github.com/bnagy/aflfix</u>

PDF FIXUPS

- Wrote a Go fixer to patch startxref offsets
- Deeper coverage, moar* bugs

american fuzzy lop 1.56b (pdftoppm UNFIXED)

- process timing		— overall results ——
run time - 9 daus 1 brs 3 mi	run time - 9 daus 4 brs 3 min 48 sec	
last new nath • A days, 4 hrs, 8 min, 40 sec		total paths • 7272
last upid opach - 0 days, 0 ms, 0 m	min EE coo	unia onochos - 1/1
last unity crash : 0 uays, 11 mrs, 42	min, 33 Sec	
Tast uniq nang : 4 days, 17 nrs, 15	min, 39 Sec	uniq nangs : 500+
- cycle progress	— map coverage —	
now processing : 7264 (99.89%)	map density :	10.9k (16.59%)
paths timed out : 0 (0.00%)	count coverage :	5.44 bits/tuple
– stage progress	 findings in dep 	oth
now trying : splice 13	favored paths :	550 (7.56%)
<pre>stage execs : 1355/3360 (40.33%)</pre>	new edges on :	1070 (14.71%)
total execs : 83.7M	total crashes :	1330 (161 unique)
exec speed : 93.33/sec (slow!)	total hangs :	72.0k (500+ unique)
fuzzing strategy yields		- path geometry
bit flips : n/a, n/a, n/a	g for the comment	levels : 29
byte flips : n/a, n/a, n/a		pending : 5845
arithmetics : n/a, n/a, n/a	g for new comment	pend fav : 3
known ints : n/a, n/a, n/a		own finds : 7271
dictionary : n/a, n/a, n/a	g for new comment	<pre>imported : n/a</pre>
havoc : 4936/44.1M, 2496/38.8M		variable : 34
trim: 0.39%/723k, n/a		
		[CDU: 78%]

american fuzzy lop 1.56b (pdftoppm FIXED)

- process timing	overall results
process criming	
run cime : 7 uays, 4 nrs, 5 mi	
last new path : 0 days, 0 hrs, 14 m	min, 22 sec total paths : /101
last uniq crash : 0 days, 2 hrs, 53 m	min, 29 sec uniq crashes : 156
last uniq hang : 4 days, 15 hrs, 48	min, 13 sec uniq hangs : 500+
- cycle progress	— map coverage —
now processing : 5238* (73.76%)	map density : 10.9k (16.68%)
paths timed out : 0 (0.00%)	count coverage : 5.22 bits/tuple
- stage progress	findings in depth
now trying : splice 17	favored paths : 584 (8.22%)
<pre>stage execs : 900/1680 (53.57%)</pre>	new edges on : 1088 (15.32%)
total execs : 87.4M	total crashes : 3480 (156 unique)
exec speed : 124.4/sec	total hangs : 61.7k (500+ unique)
– fuzzing strategy yields	path geometry
bit flips : n/a, n/a, n/a	levels : 48
byte flips : n/a, n/a, n/a	pending : 5605
arithmetics : n/a, n/a, n/a	pend fav : 1
known ints : n/a, n/a, n/a	own finds : 7100
dictionary : n/a, n/a, n/a	imported : n/a
havoc : 4800/46.0M. 2456/40.8M	variable : 144
trim: 0.30%/544k, n/a	

[cpu: **78**%]

PDF SHRINKIES

- Partial parser, built on top of my lexer
- Blindly truncate PDF stream objects
- Fix up **xref** section so indirect refs all work
- "Real Enough" so parsers get deep coverage

CDF RESULTS

- AFL corpus, after many revisions
- Acroeng samples run through pdfshrink
- Feed to terry
- Fuzz qlmanage (actual Apple software)

CDF RESULTS

- Terry results > dumb AFL results
- AFL 50% faster than Terry
- The concept works!


- xattr -c can clear quarantine metadata
- If that fails, use open -a Preview foo.pdf
- Mount ramdisks like:

diskutil erasevolume HFS+ 'ramdisk' \
`hdiutil attach -nomount ram://1048576`

- Try MallocScribble / MallocGuardEdges
- Disable CrashReporter (AFL will remind you)

• Magic compatability mode for the C++ linker

export CFLAGS="-mmacosx-version-min=10.4"

Force content type for qlmanage

-c com.adobe.pdf



Developer Tools Access needs to take control of another process for debugging to continue. Type your password to allow this.

Username:	Ben Nagy
Password:	
	Cancel Continue

run DevToolsSecurity -enable





I AM TOTALLY CALM



Ben Nagy @rantyben · Mar 14 (I can feel @daveaitel judging me for spending four hours writing lexers instead of Applescript dialog clickers)

🛧 17,2 🛧 2 📶 🚥

MOTIVATION INTENSIFIES



daveaitel @daveaitel · Mar 16 We are really pretty hard on the dry runs. I spend a lot of time imagining the font choices from the back of the room.

◆ 13 ★1 ···



@daveaitel YOU ARE NOT HELPING SHUT UP SHUT UP

◆ t∓ ★ di …

RETWEET



HOW WE DOIN?

- I'm in Adelaide now.
- 6 weeks in Total panic.
- Biggest time sinks:
 - Writing solid instrumentation
 - Doing multi-day benchmarks
 - Tooling to handle PDF

VIRTUALIZING OSX

- No leetness required!
- http://www.contrib.andrew.cmu.edu/~somlo/OSXKVM/
- http://blog.ostanin.org/2014/02/11/playing-with-mac-os-x-on-kvm/

HOW IT WAS DONE (BY SMARTER PEOPLE)

- Upstream KVM patches
 - ACPI issues, unsupported instructions
- Upstream QEMU patches
 - Patch SeaBIOS
 - Add SMC chip emulation

HOW IT WAS DONE (BY SMARTER PEOPLE)

- Chameleon bootloader is the final missing piece!
 - This is real OSX from real install media
 - (the bootloader is probably still backdoored)

OSXVIRTTIPS

- Use OSX VNC Client
 - (Finder **#**K 🖸 <u>vnc://ip.addr.of.server:5900</u>)
- Use OSXVNC Server
 - System Preferences Screen Sharing

OSXVIRTTIPS

- Use Chimera
- DO NOT INSTALL on your Macbook! 😳
- Assorted Useful Features (autostart, better res...)
- org.chameleon.Boot.plist 🗲 CASE SENSITIVE

OSXVIRTTIPS

- Download a real Installer from App Store
 - Recovery Partition image didn't work for me
- Convert to an ISO:
 - ~somlo/OSXKVM/MakeInstallDVD.sh
- Easy VNC port forwarding using -net user

hostfwd=tcp:172.16.216.135:5901-:5900

Create a Computer Account

QEMU

...

s/ =q -k

or ri 5e on

gu

ch

tØ

fs

or ri 5e

on

or ri

5e on

or

Fill out the following information to create your computer account.

David Alter			
zappy			
This will be the name of	your home folder.		
l: •••••	•••••		
near Indonesia			
🗹 Require password	to unlock screen		
 Set time zone base Send Diagnostics Help Apple improve in automatically and per usage data. About Diagnostical 	ed on current location & Usage data to Apple ts products and services l riodically sending diagnost agnostics and Privacy	n e by stic and	
Back	Continue		
	 zappy This will be the name of near Indonesia Require password Set time zone bas Send Diagnostics a Help Apple improve i automatically and per usage data. About Diagnostics 	 zappy This will be the name of your home folder. Image: Indonesia Image: Indonesia Image: Require password to unlock screen Require password to unlock screen Set time zone based on current location Set time zone based on current location Send Diagnostics & Usage data to Apple Help Apple improve its products and services I automatically and periodically sending diagno-usage data. About Diagnostics and Privacy Back Continue	 zappy This will be the name of your home folder. mear Indonesia Require password to unlock screen Set time zone based on current location Send Diagnostics & Usage data to Apple Help Apple improve its products and services by automatically and periodically sending diagnostic and usage data. About Diagnostics and Privacy

9.8

. 13

. 13

:12

All virtualisation research was carried out on genuine Apple® hardware using properly licensed software.

This research was undertaken for educational purposes only.

This presentation does not constitute legal advice.

The adjective "pavonine" meaning "of or like a peacock" is from the Latin *pavoninus*.

SCALING OSX?

- Planned on building a massive OSX Cloud
- Decided it was a bad idea
- Scale by CDF instead

WHY CDF?

- pdftoppm is "slow" for AFL 160 tests/sec
- **ghostscript** even slower ~6
- Preview is about 0.6
- CDF is a force multiplier

WHY CDF?

- http://lcamtuf.blogspot.com/2015/03/anotherround-of-image-bugs-png-and.html
- IO image parsing bugs in IE
- WITHOUT EVER FUZZING IE

Corpus Driven Fuzzing: 50 ethical! GNUY linox cluster Mostly useles CSS IBB Crashes. Try AFL Park pennies [patinba] Poppler 95 Preview c/womong e cmin Crashess ARPUS (TC) WORKIN Continuall evoly fshrink Ange Guten Free! Tests/ amples

HOW WE DOIN?

- I'm back in Majuro now
- Liver hurts from SySCAN
- Biggest failure was lack of C&C tooling
- Biggest success was solidifying CDF approach
- Approach seems viable



FUTURE WORK

http://glitched9700.deviantart.com/art/Broken-Robot-II-82773491

FUTURE WORK

- AFL Clouds
 - Designed lots of stuff, figured it was niche
 - NSQ, Docker, Kubernetes blah blah blah?
 - Ansible, Puppet blah blah?

FUTURE WORK

- Finish full triage framework (queues, DB, etc)
 - Unify indicators with GDB plugin
- Write ObjC apps that use Apple frameworks?
 - PrivateFrameworks/CorePDF

FUTURE WORK

- OSX Instrumentation (OSS)
 - New LLVM passes being discussed RIGHT NOW
- OSX Instrumentation (Closed)
 - New results with dyninst
 - Intel PT on Broadwell CPUs?

SOFTWARE

HTTPS://GITHUB.COM/BNAGY/...

- aflfix
- crashwalk (includes afl-triage)
- francis (includes exploitaben.py)
- gootool (toy Mach-O disassembler based on Capstone engine)
- pdflex (includes pdftok, pdfshrink)
- terry

ben@coseinc.com

@rantyben





QUESTIONS?

SOME LINKS

http://web.mit.edu/darwin/src/modules/xnu/osfmk/man/ Mach IPC Interface https://github.com/shoumikhin/Mach-O-Hook Mach-O Hooking http://en.wikipedia.org/wiki/Mach_%28kernel%29 http://en.wikipedia.org/wiki/XNU http://en.wikipedia.org/wiki/Darwin_(operating_system) http://brendanzagaeski.appspot.com/0004.html minimal pdf http://www.ghostscript.com/doc/9.15/Use.htm how to use ghostscript http://events.linuxfoundation.org/sites/events/files/slides/lcna13_kleen.pdf Intel PT http://www.adobe.com/content/dam/Adobe/en/devnet/acrobat/pdfs/PDF32000_2008.pdf PDF spec https://github.com/jfoote/exploitable gdb exploitable http://superuser.com/questions/178587/how-do-i-detach-a-process-from-terminal-entirely http://baptiste-wicht.com/posts/2011/07/profile-applications-linux-perf-tools.html http://lldb.llvm.org/python_reference/index.html http://ho.ax/tag/lldb/ http://www.opensource.apple.com/source/xnu/xnu-2050.22.13/osfmk/mach/kern_return.h # exception subtypes http://www.opensource.apple.com/source/xnu/xnu-1456.1.26/osfmk/mach/exception_types.h?txt # exception types https://code.google.com/p/honggfuzz/source/browse/trunk/mac/arch_mac.c http://llvm.org/svn/llvm-project/lldb/trunk/examples/python/disasm.py http://lldb.llvm.org/python_reference/index.html https://developer.apple.com/library/mac/documentation/Performance/Conceptual/ManagingMemory/Articles/MallocDebug.html MallocScribble http://blog.ostanin.org/2014/02/11/playing-with-mac-os-x-on-kvm/ http://www.contrib.andrew.cmu.edu/~somlo/OSXKVM/