



Finding N-day Security Vulnerabilities in Third-party Software

Static Analysis Days @ Verifysoft, May 2021

Paul Anderson, VP of Engineering GrammaTech, Inc.

1

Outline



- N-day vulnerabilities
- Risks of third-party software
- Techniques for finding N-days
- N-days in Binaries

2

COTS Vulnerabilities



Forbes

Jul 9, 2019, 12:35am EDT | 56,886 views

Confirmed: Zoom Security Flaw Exposes Webcam Hijack Risk, Change Settings Now

Zak Doffman Contributor @ Cybersecurity
I write about security and surveillance.

Forbes

EDITORS' PICK | Dec 23, 2019, 03:44am EST | 136,547 views

Windows 10 Security Warning As Dropbox Zero-Day Is Confirmed

Davey Winder Senior Contributor @ Cybersecurity
I report and analyse breaking cybersecurity and privacy stories



The New York Times

irus Outbreak > **LIVE** Latest Updates Maps and Cases See Your Local Risk New Variants Tracker

Zoom's Security Woes Were No Secret to Business Partners Like Dropbox

Dropbox privately paid top hackers to find bugs in software by the videoconferencing company Zoom, then pressed it to fix them.



Adobe PDF Security Issues

Adobe PDF Security Issues, Acrobat Vulnerabilities, PDF Cracks



- DRM SECURITY ITEMS
- PDF Security Overview
- PDF Encryption Overview
- PDF Password Protection
- Our DRM Products

© GrammaTech, Inc. All rights reserved.

COTS Vulnerabilities



ZDNet

EDITION US

VIDEOS WINDOWS 10 5G BEST VPNs CLOUD SECURITY AI

MUST READ: The global quantum computing race has begun. What will it take to win it?

Slack fixes vulnerability exploitable for session hijacking, account takeovers

SecurityIntelligence

Home / Threat Research / Malware

Killer Music: Hackers Exploit Media Player Vulnerabilities



Home » Cybersecurity » Security Awareness » Collaboration Tools: Essential but Risky in Remote Work

Collaboration Tools: Essential but Risky in Remote Work

by Joan Goodchild on April 1, 2020

If your teams weren't already using popular collaboration tools such as Slack, Microsoft Teams, Zoom and Google, chances are they are now. The sudden widespread directive for most employees to work from home due to the spread of the COVID-19 virus...



threatpost

Cloud Security Malware Vulnerabilities InfoSec Insiders Podcasts

Android Keyboard App Could Swindle 40M Users Out of Millions Stubborn M

Google Discloses Chrome Flaw Exploited in the Wild



Author: Lindsey O'Donnell
November 2, 2019 12:35 am



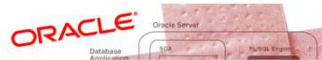
threatpost

Cloud Security Malware Vulnerabilities InfoSec Insiders Podcasts

Brian Donohue On Security and Journalism Apple Ph

PeopleSoft Vulnerabilities Elevate ERP Security Issues

Author: Michael Mirsoo



© GrammaTech, Inc. All rights reserved.

OWASP Top 10



A9

:2017

Using Components with Known Vulnerabilities

15

App. Specific	Exploitability: 2	Prevalence: 3	Detectability: 2	Technical: 2	Business ?
While it is easy to find already-written exploits for many known vulnerabilities, other vulnerabilities require concentrated effort to develop a custom exploit.		Prevalence of this issue is very widespread . Component-heavy development patterns can lead to development teams not even understanding which components they use in their application or API, much less keeping them up to date.		While some known vulnerabilities lead to only minor impacts, some of the largest breaches to date have relied on exploiting known vulnerabilities in components.	Depending on the assets you are protecting, perhaps this risk should be at the top of the list.

5

Definitions of Vulnerabilities



- 0-day
 - Just disclosed
 - No patch available
 - Exploits may not be widespread
- N-day
 - Patch *is* available
 - Exploits *likely* to be rampant in the wild

DARKReading

SIGN UP FOR OUR NEWSLETTERS

Authors Slideshows Video Tech Library University Security Now Calendar Black Hat News Om

THE EDGE ANALYTICS ATTACKS / BREACHES APP SEC CLOUD ENDPOINT IoT OPERATIONS PERIMETER

VULNERABILITIES / THREATS

3/26/2018
10:30 AM

Ang Cui
Commentary

Connect Directly

0 COMMENTS
COMMENT NOW

The Overlooked Problem of 'N-Day' Vulnerabilities

N-days -- or known vulnerabilities -- are a goldmine for attackers of industrial control systems. It's time for a new defense strategy.

Security Researcher Joseph Pantoga contributed to this article.

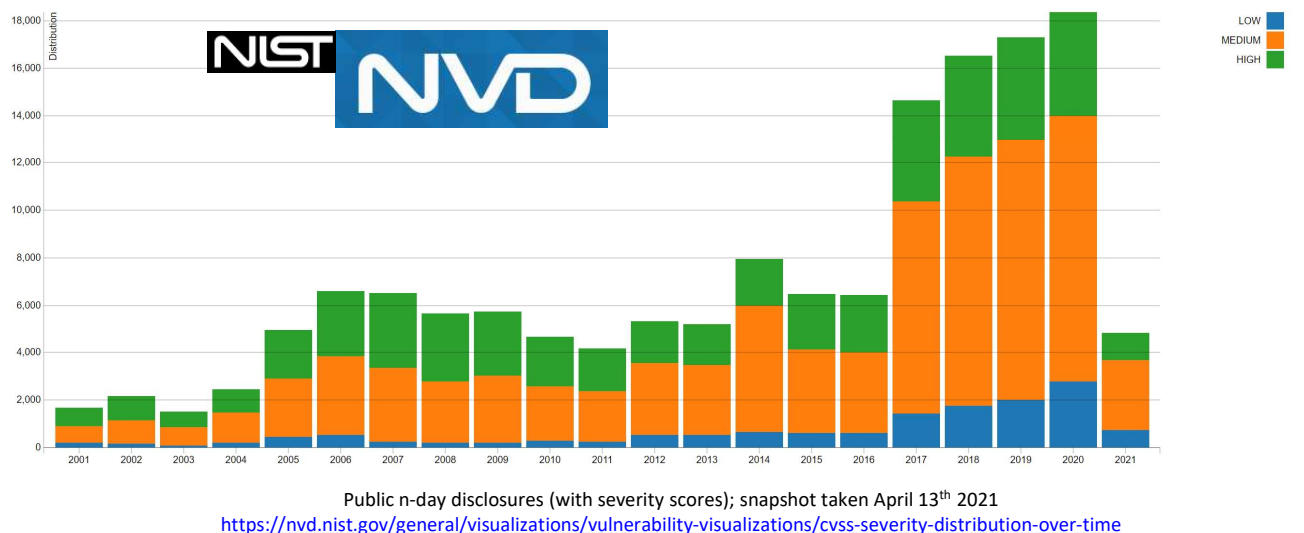
Zero-day attacks tend to steal the spotlight when it comes to cybersecurity threats, but it is actually the known vulnerability — the "N-day" — that poses a much larger problem for many organizations and particularly those in the industrial sectors.

Whereas zero-days are a class of vulnerability that is unknown to a software developer or hardware manufacturer, an N-day is a flaw that is already

<https://www.darkreading.com/vulnerabilities--threats/the-overlooked-problem-of-n-day-vulnerabilities/a/d-id/1331348>

6

Trend: SW Increasingly Under Attack



7

N-days: a Major Threat



99.9% of exploited n-days disclosed over a year

Verizon DBIR 2015

Computer systems used by federal agencies often have n-days

U.S General Accountability Office (GAO) Report on Information Security, December 2018

Weapons systems rely on commercial and open-source software, and are subject to n-day exploits

U.S GAO Report on Weapon Systems Cybersecurity, October 2018

8

Equifax Data Breach, 2017



- Sensitive personal data of 148 Million Americans stolen
 - Including SSN
- Cost Equifax at least \$380M (possibly up to \$600M)
- Estimated cost to Americans (for freezing credit): \$1.4B
- Exploited an **n-day** in the Java Struts framework
 - Left **unpatched for several months** (patch was publicly known)

Outline



- N-day vulnerabilities
- Risks of third-party software
- Techniques for finding N-days
- N-days in Binaries

OSS Is Integral to Modern Development



Libraries.io monitors 7,639,839 open source packages across 37 different package managers, so you don't have to. [Find out more](#)

<p>Discover new software</p> <p>Search 7.6M packages by license, language or keyword, or explore new, trending or popular packages.</p> <p>Explore</p>	<p>Monitor your dependencies</p> <p>Stay up to date with notifications of updates, license incompatibilities or deleted dependencies.</p> <p>Login</p>	<p>Maintain your OSS project</p> <p>Understand your users and make informed decisions about features with usage and version data.</p> <p>Login</p>	<p>Use Libraries.io data</p> <p>Use Libraries.io data in your applications, services or research. Use our API to stay up to date.</p> <p>Documentation</p>
---	---	---	---

Supported Package Managers

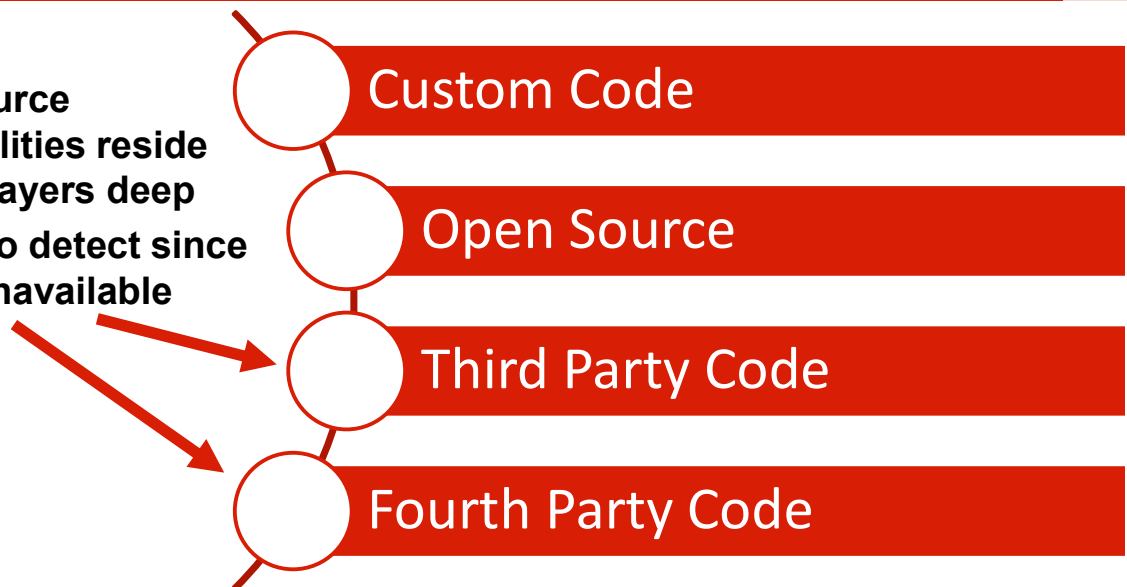
Go 1.89M Packages	npm 1.27M Packages	Maven 277K Packages	Packageist 267K Packages
PyPI 256K Packages	NuGet 255K Packages	Rubygems 198K Packages	CocoaPods 76.4K Packages
WordPress 70.4K Packages	Bower 69.7K Packages	Cargo 50.6K Packages	Clojars 43.8K Packages
CPAN 37.8K Packages	CRAN 12.9K Packages	Pub 7.9K Packages	Package 13.4K Packages
Meteor 13.4K Packages	Atom 13.3K Packages	Hex 10.8K Packages	PlatformIO 8.68K Packages
Puppet 6.72K Packages	Emacs 5.27K Packages	Homebrew 4.7K Packages	SwiftPM 4.21K Packages
Carthage 4.19K Packages	Julia 3.09K Packages	conda 2.83K Packages	Sublime 2.03K Packages
Dub 1.89K Packages	Racket 1.85K Packages	Haxelib 1.55K Packages	Nimble 1.51K Packages
Elm 1.51K Packages	Jam 772 Packages	PureScript 690 Packages	Alcatraz 454 Packages
Include 524 Packages	Shards 33 Packages		



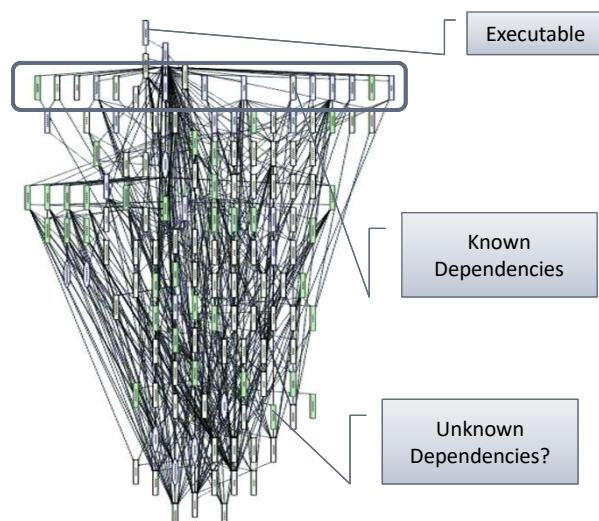
Hidden Open Source Vulnerabilities



Open-Source vulnerabilities reside multiple layers deep
Difficult to detect since source unavailable



Reality of Modern SW Development

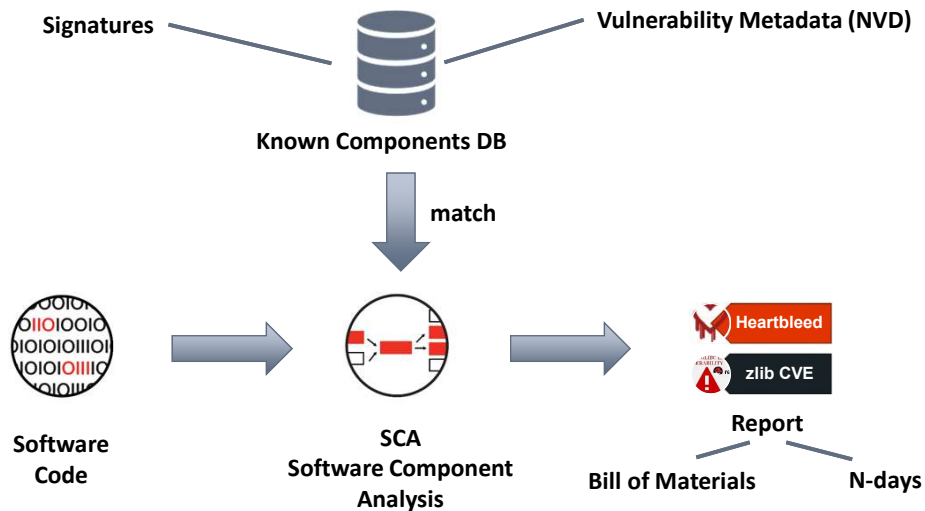


Outline



- N-day vulnerabilities
- Risks of third-party software
- Techniques for finding N-days
- N-days in Binaries

Software Component Analysis (SCA)



Source vs Binary SCA



- Source-code SCA is well-established
 - Several commercial tools and services available
 - Techniques from Information Retrieval scale and work well
 - Hashing, Signature matching, n-grams, etc.
- Binary SCA is *hard*....

Binary SCA Use Cases



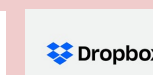
Application Development - Internal and External Use

Product Configurators
Sales Forecasting
Database applications
Workflow applications
Customer facing apps
ERP

Product Development



COTS Deployed Internally



17

Outline



- N-day vulnerabilities
- Risks of third-party software
- Techniques for finding N-days
- **N-days in Binaries**

18

Why Binary SCA is Difficult



```

308 /* One of the argv values. */
309 if (file_names != NULL)
310 {
311     row_file = fopen(file_names->name, "r");
312     if (row_file != NULL)
313     {
314         switch (inst)
315         {
316             case 'A': /* increment array variable (add one). */
317                 var_name = byte(addr);
318                 if ((var_name & 0x0F) != 0)
319                     var_name = ((var_name & 0x0F) << 8) + byte(addr);
320                 inst_array[var_name];
321                 break;
322             case 'B': /* Branch to a label. If DOS != 0. Remove value on DOS. */
323                 c_code = hex2zero(hex(inst)&0x0F);
324                 jmp (c);
325             case 'C': /* Jump to a label. */
326                 label_name = byte(addr); /* Use address bits first. */
327                 label_name = byte(addr) << 8;
328                 if (inst == 'J') (inst == 'J' && c_code)
329                     || (inst == 'J' && !c_code);
330                 ep = funcaddr(addr, PC, PC, PC, PC);
331                 if (label_name >> 16) inst |= 0x01;
332                 if (label_name >> 8) inst |= 0x02;
333                 while (1) { ep = ep + 1; next; }
334                 if (inst == 'J') ep = ep + 1; next;
335             default:
336                 break;
337         }
338     }
339 }
    
```

Source Code

Components:

- libxyz v2.3
- openABC v4.1.2



Binary Code

*Cut-and-paste
Forking and refactoring
Compilers
Linkers*

What components?
What versions?

Critical Technical Challenge



Assembly listings for the Heartbleed bug, compiled using gcc-4.4 and clang-3.2:

```

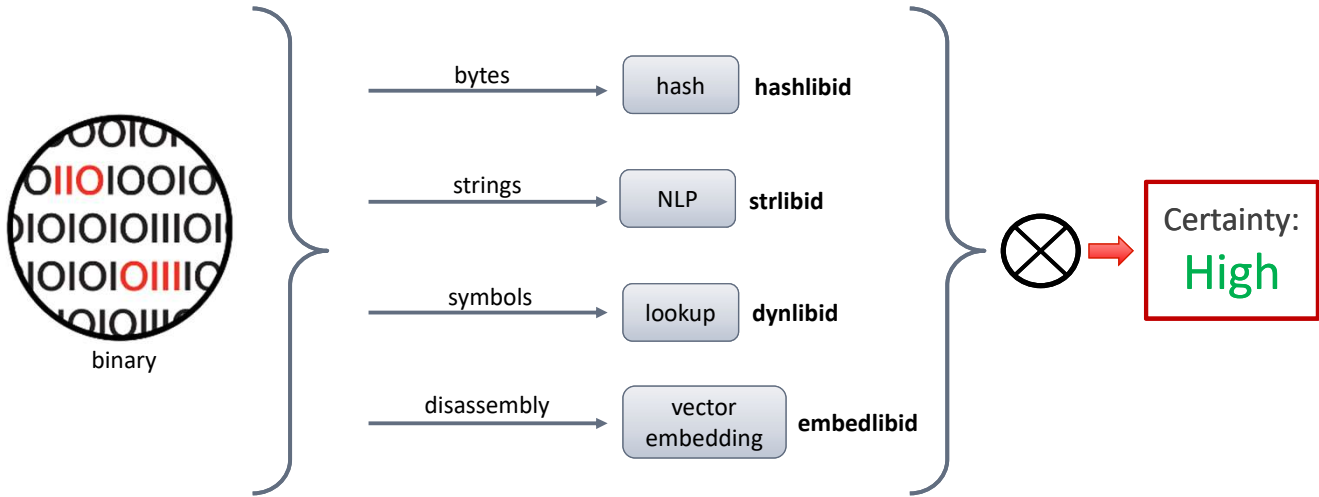
tls1_process_heartbeat: gcc-4.4
sub esp,0x4C
mov dword [esp+76+var.C],esi
mov esi,dword [esp+76+s]
mov dword [esp+76+var.10],ebx
mov dword [esp+76+var.8],edi
mov dword [esp+76+var.4],ebp
mov ecx,dword [esi+88]
call __i686.get_pc_thunk.bx
add ebx,0x2D6F1
mov ebp,dword [ecx+280]
movzx eax,byte [ebp]
mov word [esp+76+var.24],ax
movzx edx,byte [ebp+1]
movzx edi,byte [ebp+2]
shl edx,8
or edi,edx
...
    
```

```

tls1_process_heartbeat: clang-3.2
push ebp
push ebx
push edi
push esi
sub esp,0x2C
call loc.2B3EC
loc.2B3EC: pop ebx
add ebx,0x2660C
mov eax,dword [esp+60+arg.0]
mov ecx,dword [eax+88]
mov edi,dword [eax+100]
mov dword [esp+60+var.20],edi
mov ebp,dword [ecx+280]
movzx edx,byte [ebp+2]
mov dword [esp+60+var.14],edx
movzx esi,byte [ebp+1]
...
    
```

The same source code can look very different at the binary level

Approach: Biometrics for Components



21

Superficial Properties of Binaries



- Properties are independent of the instruction set architecture
- Hashes
 - Compute hash of the file and look up in catalog of hashes
- Strings
 - Extract sequences of bytes that make up strings
- Symbols
 - Extract symbolic references

```

0114040 355 283 u y 1 216 367 1 266 p 346 v 241 315 f
0114060 224 022 1 277 333 263 256 207 330 h 003 r 002 302 r
0114100 332 224 363 r \a < w 350 u $ 347 005 240 035 304 201
0114120 001 202 k 037 7 017 235 3 233 360 o 314 235 326 205 201
0114140 317 313 347 375 j 351 222 270 367 \c r 8 + 351 w 371
0114180 o 217 ( e > 036 275 \b 206 334 215 036 021 - 367 1
0114200 273 207 v 306 236 L T a 320 g 333 223 204 332 1 207
0114220 003 222 347 237 343 351 317 z V 310 h 340 R i 224 \
0114240 1 202 002 q 0 202 002 m 002 001 001 0 201 252 0 201
0114260 225 1 \v 0 \c 006 003 0 004 006 003 002 0 s 1 \v
0114300 0 \c 006 003 0 004 \b 023 002 n T 1 027 0 025 006
0114320 003 0 004 \a 023 016 S a l t L a k e
0114340 c i t y 0 036 0 034 006 003 0 004 \r 023 025 1
0114360 n e u s E R T R U S T N e t
0114400 w o r k l i 0 037 006 003 0 004 \v 023 030 h
0114420 t t p : / / w w w . u s e r t . r
0114440 u s e t . c o m 1 035 0 033 006 003 0 004 003
0114460 023 024 U T N - u s e r f i e s t e
0114500 0 b i e c t 002 020 N 260 207 217 314 s s e
0114520 262 33
0114540 e 00
0114560 H 20
0114600 \r 00
0114620 001 \
0114640 e 0 z z u + 006 \v * 206 h 206 367 \c 001 \c
0114660 020 002 \f 1 034 0 032 0 030 0 026 004 024 6 r h
0114700 0 242 i h 371 353 E 226 361 331 232 223 016 247 m
0114720 372
0114740 004
0114760 270 34
0115000 265
0115020 \0 00
0115040 001 03
0115060 273 20
0115100 275 365 253 265 o 1 243 * 004 k 342 362 342 \f / 177
0115120 ) K 354 * & 232 371 355 K 260 327 v 266 340 371 Y
0115140 - 31
0115160 374 26
0115200 300 26
0115220 372
0115240 303 \
0115260 277 24
    
```

Salt Lake City

Invalid SCAL chunk ignored: non-positive width

libpng version 1.5.10 - March 29, 2012

22

Example: Bill of Materials



Scan of a single file:
Qt5Gui.dll

qwe > qweqwe > Qt5Gui.dll: Scan Done, Scan Depth: Shallow

Bill of Materials Vulnerabilities Scan Status

Search component name

Pass/Fail	Component Name	Component Version	Match	CVSS Distribution	Target
⊗	libpng	1.5.10	High	0 0 2 5 1	Qt5Gui.dll
⊙	libiconv	1.15	High	0 0 0 0 0	Qt5Gui.dll

Items per page: 25 1 - 2 of 2

libiconv detected,
but safe

libpng detected, with
several vulnerabilities

Example: Vulnerabilities View



Pass/Fail	Component Name	Component Version	Match	CVSS Distribution	Target
⊗	libpng	1.5.10	High	0 0 2 5 1	Qt5Gui.dll

File path:
Qt5Gui.dll

Vulnerability Information:

- CVSS Unassigned
- CVSS Low
- CVSS Medium
[CVE-2013-7353](#)
[CVE-2013-7354](#)
- CVSS High
[CVE-2015-0973](#)
[CVE-2015-8126](#)
[CVE-2015-8472](#)
[CVE-2016-10087](#)
[CVE-2016-3751](#)
- CVSS Critical
[CVE-2017-12692](#)

Link to CVEs associated with the
detected version of libpng

Conclusion



- N-day vulnerabilities are abundant
- Particularly pernicious for Supply Chain Risk Management
- Software Composition Analysis tools can help close the door on large classes of serious security weaknesses
- Tools capable of analyzing binaries are now available