2018 STATE OF THE SOFTWARE SUPPLY CHAIN

Sonatype's 4th annual report on managing
open source components to accelerate innovation.

Sonatype

Introduction

From banking, to manufacturing, to healthcare, to entertainment; companies capable of delivering innovative software applications are disrupting established players and gaining share in every industry.

To survive and compete effectively, CEOs and shareholders are placing intense pressure upon IT leaders to accelerate the pace of software innovation. In response, organizations are hiring armies of software developers, consuming unprecedented amounts of open source components, and equipping teams with next generation and cloud-native tools designed to automate and optimize the entire software development lifecycle.

In this world, speed is critical, open source is everywhere, and security concerns are sometimes relegated to the back seat — which is why we're once again examining the state of the open source software supply chain. Like previous reports, the 2018 State of the Software Supply Chain Report blends a broad set of public and proprietary data with expert research and analysis. Key findings in this year's report include:

- Open source vulnerabilities increased 120% YoY and their mean time to exploit compressed 400%.
- Public vulnerability databases lack information on more than 1.3 million open source security advisories.
- Suspected or known open source breaches increased 55% YoY.
- DevOps teams are 90% more likely to comply with open source governance when policies are automated.
- Managing software supply chains through automated OSS governance reduces the presence of vulnerabilities by 50%.
- Government regulation across the United States and Europe hints at software liability on the horizon.

Different from the 2017 report, this year's report highlights new methods cyber-criminals are employing to infiltrate software supply chains, offers expanded analysis across languages and ecosystems, and more deeply explores how government regulations are likely to impact the future of software development.

At Sonatype we have a long history of partnership with the world of open source software development. From our humble beginning as core contributors to Apache Maven, to supporting the world's largest repository of open source components (Maven Central), to distributing the world's most popular repository manager (Nexus Repository Manager), we exist for one simple reason: to help accelerate software innovation.

The sole purpose of this report is to share with you the things that we observe from our unique vantage point within the open source community. We hope you find it useful, and we welcome your feedback.

Sincerely,

WAYNE JACKSON

Chief Executive Officer, Sonatype

OSS BREACHES

55%

increase in OSS **breaches** YoY

P.5

8

Struts **breaches** made headlines over past year

P.6

45 to 3

decline in number of days between **vulnerability disclosure and exploit**

P.5

TIDAL WAVE OF INNOVATION & AUTOMATION IN SOFTWARE DEVELOPMENT

87B

download requests (java), representing a **68%** year over year **increase**

P.13

170,000

average enterprise downloads of open source components per year

P.17

26K

average number of npm packages downloaded annually per JavaScript developer

P.13

KNOWN VULNERABILITIES

1-in-8 OSS component downloads contain a known **security vulnerability**

P.17

USE OF AUTOMATION

50%

Reduction in use of vulnerable OSS components when applying automation

P.27

57%

DevOps teams automate security across the software development lifecycle

P.23

90%

DevOps teams are 90% more compliant when OSS security policies are automated

P.24

OSS & GOVERNMENT

19

number of government organizations calling for improved OSS governance

P.29

11

Reports of hackers injecting vulnerabilities into open source releases

P.8

Table of Content

Chapter 1: The Wake Up Call "We Are All Equifax"

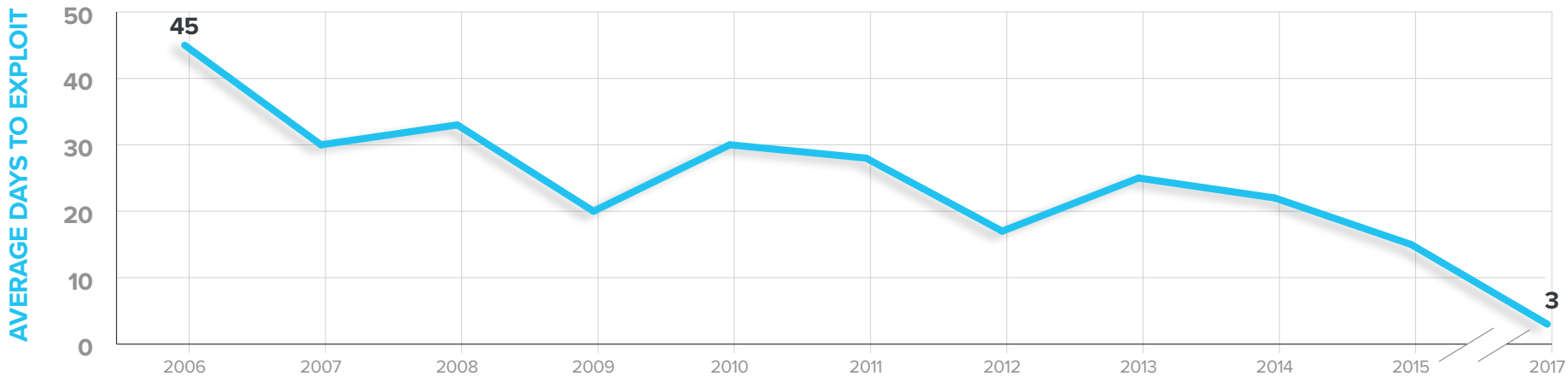
Open source software breaches are on the rise

Exploits of vulnerable open source software components are on the rise. The 2018 DevSecOps Community Survey of 2,076 IT professionals found that 30% of respondents claimed a breach stemming from the use of vulnerable open source components — up 55% over the 2017 survey of the same name. Since 2014 — the year that OpenSSL's Heartbleed vulnerability made headlines — open source related breaches are up 121%.[1]

The window to respond is shrinking rapidly

A series of high profile and devastating cyber attacks last year demonstrated that adversaries have the intent and ability to exploit security vulnerabilities in the software supply chain. Never was that so apparent than in the massive breach at Equifax. While news of the breach traveled far and wide, **one detail that did not receive sufficient attention was the three days between the Apache Struts vulnerability being announced (March 7) and the initial breach at Equifax (March 10).**

Average Days Before Vulnerability is Exploited



Sources: Gartner, IBM, Sonatype

Equifax was not alone. Hackers quickly attempted to exploit the Struts vulnerability elsewhere. According to David Hogue, a senior technical director for the NSA's Cybersecurity Threat Operations Center (NCTOC), "We had a nation-state actor within 24 hours scanning for unpatched [Struts] servers within the DoD."[2] **Struts related breaches were also recorded at Alaska Airlines, the Canada Revenue Agency, Okinawa Power, the Japanese Post, the India Post, AADHAAR (India's social security system), and the GMO Payment Gateway**.[3] [4] [5] [6] [7] [8] [9]

According to a recent Gartner/IBM study, the average time between a vulnerability being reported and then exploited shrank from 45 days to 15 days between 2006 and 2015.[10] Another study from Tenable calculated the vulnerability exploit window to be 5.5 days.[11]

The time required for hackers to exploit a newly disclosed open source vulnerability has shrunk 400% in the last decade. This harsh reality establishes a new normal for software supply chain management and demands that organizations are prepared to do three things within 48 hours of a new public disclosure: (1) assess which, if any, of their production applications are exploitable, (2) establish a comprehensive plan to remediate potential exposure, and (3) implement necessary fixes in production. In October 2017, Josh Corman echoed this reality to Congressional investigators saying, "Once a vulnerability is known, there is a gold rush effect where adversaries create methods of finding and exploiting them — fairly quickly."[12]

Real time proof: a new critical vulnerability in Apache Struts

On August 22, 2018 — on the very day of this writing — the Apache Software Foundation announced yet another critical remote code execution vulnerability in Apache Struts.[13] The vulnerability (CVE-2018-11776) was identified and reported by Man Yue Mo from the Semmle Security Research team.[14]

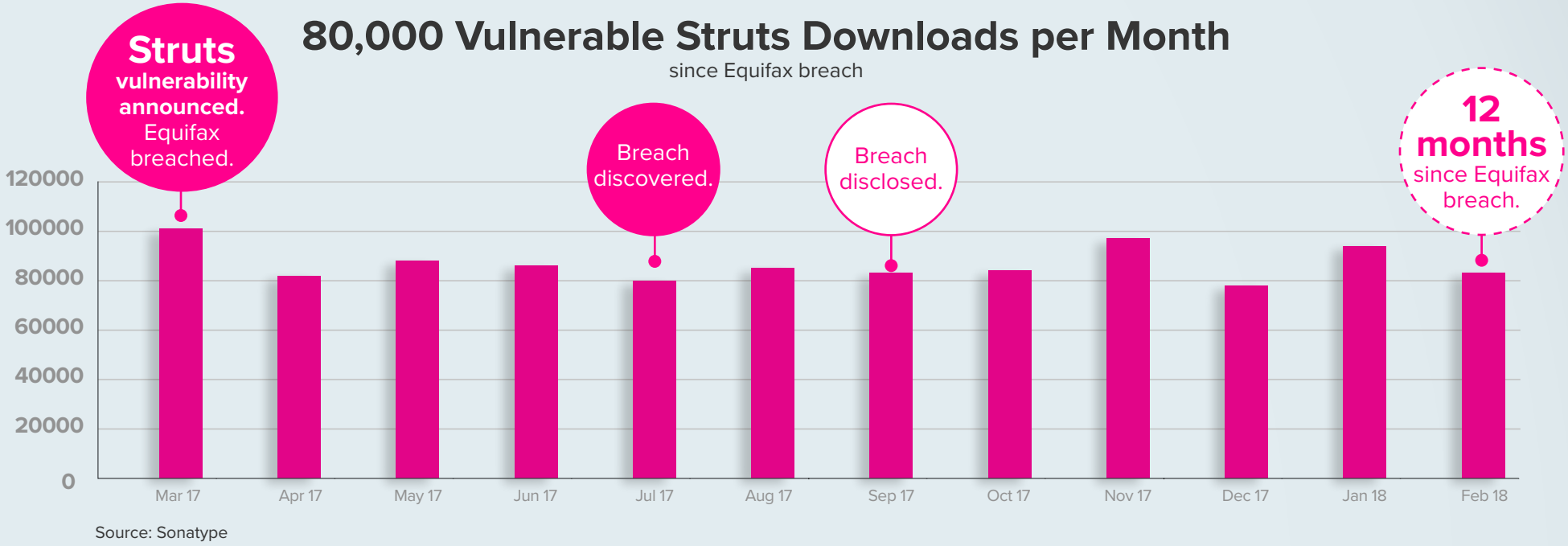
The public disclosure urgently advised organizations and developers using Struts to upgrade their components immediately to versions 2.3.35 and 2.5.17. As outlined in the section above, previous public disclosures of similarly critical vulnerabilities have resulted in exploits being published within days — attacks in the wild within three days, and devastating damage to critical infrastructure and massive theft of customer data over time. In this specific case, **automated exploits were found on GitHub and chatter was heard in Russian and Chinese underground forums within two days of the vulnerability disclosure.**[15] The result being that organizations around the world were scrambling to respond to a brand new threat that they just learned about within the last 48 hours. The race between well-equipped hackers and enterprise remediation paths had begun.

In these situations, DevOps-native organizations with the ability to continuously deploy software releases have an automation advantage that allows them to stay one step ahead of the hackers. According to a recent Forrester survey, 8% of organizations deploy once per day, 25% deploy once per week, and 68% of organizations deploy less than once per month on average.[16] Similarly, the most recent *Accelerate: State of DevOps Report* from DORA reveals that 7% of elite DevOps teams can deploy changes in less than an hour, 48% deploy changes between one day and one week, and 52% deploy between one week and six months.[17] In this new normal, organizations that release software faster face significantly less risk than their peers.

Unbridled consumption of vulnerable Struts and Spring versions

According to Sonatype's analysis of the Central Repository, the volume of monthly vulnerable downloads tied to Struts lies between 80,000 and 100,000.[18] Thousands of organizations indiscriminately download vulnerable versions, including 57% of the Fortune Global 100.[19] "Troublingly, the fallout from Equifax has not seemed to dissuade corporations from pulling unsafe code into their networks. **As many as 8,780 organizations have continued to download known, vulnerable versions of the Struts software** since Equifax's breach disclosure on September 7, 2017," according to Robert Hackett at Fortune.[20]

Sonatype's research also points to poor cybersecurity hygiene around the use of another popular web application framework called Spring. When a critical vulnerability was discovered in the framework in September 2017, vulnerable downloads had been averaging about 85,000 per month. In the months following the discovery of the vulnerability, downloads diminished by only 15% to 72,000 per month.[21]



Source: Sonatype

A new battlefront: software supply chain attacks

Five years ago, large and small enterprises alike witnessed the first prominent Apache Struts vulnerability.[22] In this case, Apache responsibly and publicly disclosed the vulnerability at the same time they offered a new version to fix the vulnerability. Despite Apache doing their best to alert the public and prevent attacks from happening — many organizations were either not listening, or did not act in a timely fashion — and, therefore, exploits were widespread.[23] Indeed, when companies fail to follow disclosures, it creates a perfect storm: a well publicized exploit, millions of targets, and no one paying attention.[24]

Since that mainstream Struts vulnerability in 2013, the development community has witnessed Shellshock, Heartbleed, Commons Collection and others all following the pattern of widespread exploit post-disclosure.[25] [26] [27] The 2017 Equifax attack documented an exploit in the wild just three days post-disclosure.

Over the last 18 months a dangerous new trend has emerged. Specifically, a series of **11 events triangulate a serious escalation of software supply chain attacks. Attackers directly inject vulnerabilities into the supplier networks i.e. the Shifting Battlefront of Attacks: Hackers Inject Malicious Code into Software Supply Chain OSS projects themselves.**

A Shifting Battlefront of Attacks "Hackers Inject Malicious Code into Supply Chain"

March 2016 - August 2018

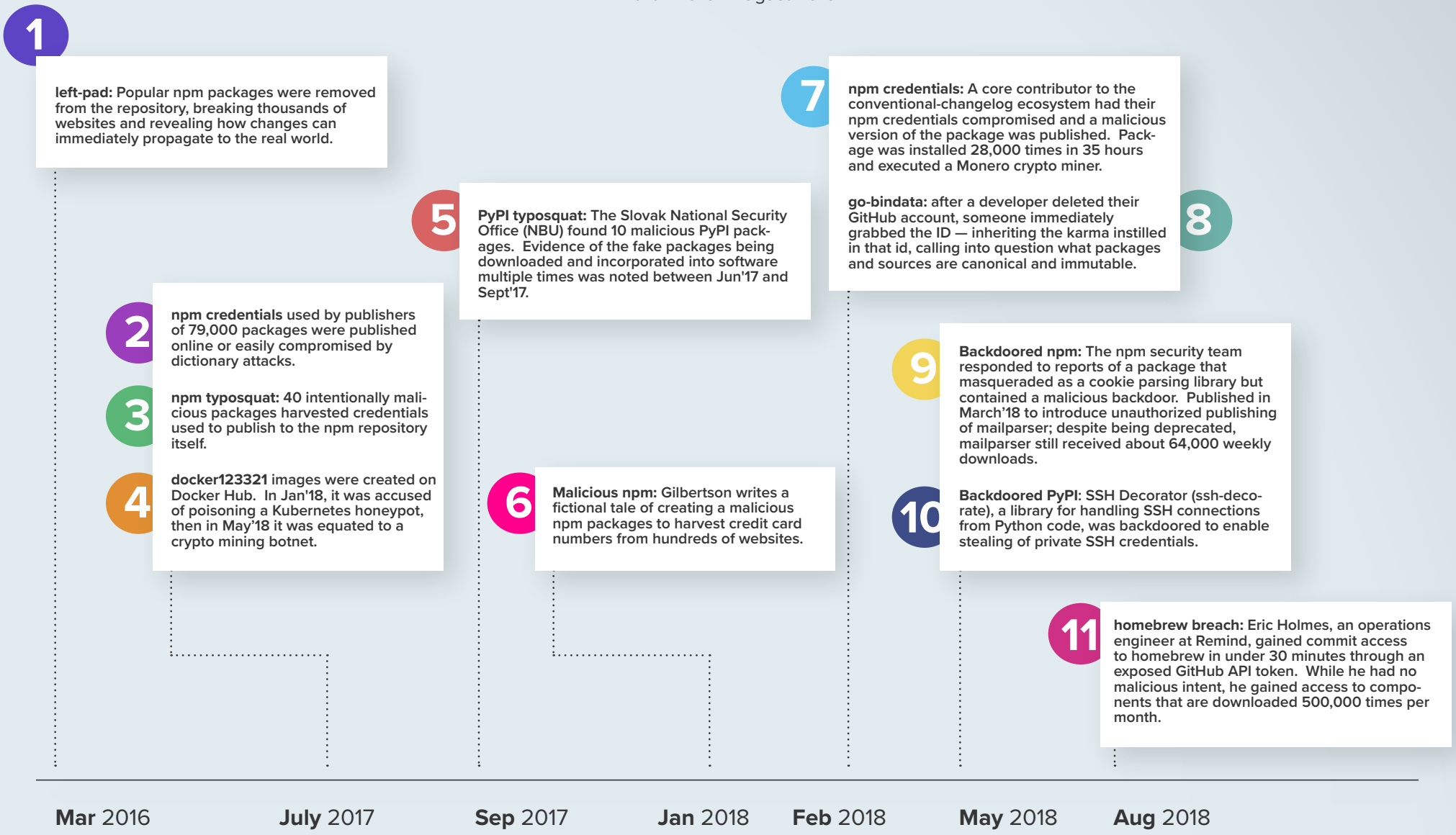


Image by Sonatype

March 2016 — left-pad: Trivial but popular npm packages were removed from the repository, instantly breaking thousands of websites. This incident showed how widespread even small modules are and how immediate changes in the repository propagate to the real world.[28]

July 2017 — npm study: The credentials used by publishers of more than 79,000 releases were already published online or were easily compromised by dictionary attacks. From this, we observe that not all publishers are diligent with their own security.[29]

July 2017 — npm typosquat: Forty intentionally malicious releases collected environment variables and focused on harvesting the very credentials used to publish to the npm repository itself. These sat unnoticed for two weeks. Fortunately, the effect of this attack was limited because the releases themselves were not popular. The attackers instead tried to trick consumers using a technique made popular by dns scammers called typosquatting.[30] [31]

July 2017 — The first bunch of malicious docker123321 images were created on Docker Hub. Three more image groups appeared on Docker Hub between October 2017 and February 2018. In January 2018, docker123321 was accused of poisoning a Kubernetes honeypot and then in May 2018 was equated to a crypto mining botnet. Docker Hub deleted 123321 on 10 May 2018.[32]

September 2017 — PyPI Typosquat: Similar to the npm attack, these components collected information and streamedit to an IP based in China.[33] [34] The Slovak National Security Office (NBU) found and reported the ten malicious Python releases (e.g., urllib-1.21.1.tar.gz, based upon a well known release urllib3-1.21.1.tar.gz) on PyPI, which were promptly removed from the public repository.[35] [36] Evidence of the fake releases being downloaded and incorporated into software multiple times was noted between June 2017 and September 2017.[37]

January 2018 — David Gilbertson writes a fictional tale of creating a malicious npm release to harvest credit card numbers from hundreds of websites. He does so contributing several hundred pull requests to various front-end releases and their dependencies and brags, "I'm now getting about 120,000 downloads a month, and I'm proud to announce, my nasty code is executing daily on thousands of sites, including a handful of Alexa-top-1000 sites, sending me torrents of usernames, passwords and credit card details." While fictional, his tale was very plausible and was applauded by over 200,000 readers in the development community.[38]

February 2018 — A core contributor to the conventional-changelog ecosystem had his npm credentials compromised. Using these credentials, a malicious version of conventional-changelog (version 1.2.0) was published to npm. While the release was only available for 35 hours, some estimated that it was downloaded and installed over 28,000 times during that period. The release executed a Monero crypto miner.[39]

February 2018 — go-bindata hijack: The developer of a popular component deleted his Github account and someone immediately grabbed the ID — effectively inheriting the karma instilled in that id, with the ability to publish new authoritative versions of its releases.[40] The ability to delete and reinstate projects from GitHub took many developers by surprise, as it pulled into question what releases and sources are canonical and immutable.[41]

May 2018 — The npm security team received and responded to reports of a release that masqueraded as a cookie parsing library but contained a malicious backdoor. The backdoored version had been published in March 2018 to introduce unauthorized publishing of mailparser; despite being deprecated, mailparser still received about 64,000 weekly downloads.[42] The three components and three versions of a fourth component were then unpublished from the npm Registry.[43]

May 2018 — SSH Decorator (ssh-decorate), a library for handling SSH connections from Python code, was backdoored to enable stealing of private SSH credentials.[44]

August 2018 — Eric Holmes, an operations engineer at Remind, gained commit access to homebrew in under 30 minutes through an exposed GitHub API token. While he had no malicious intent, he gained access to components of homebrew that are downloaded 500,000 times per month.[45]

Just a few years ago, organizations were concerned about the possibility that they might be attacked within a few months of a new vulnerability being publicly disclosed. In the past 18 months, organizations have been increasingly concerned about being exploited within a few days of a new vulnerability being disclosed. **Today, organizations must contend with the fact that hackers are intentionally planting vulnerabilities directly into the supply of open source components.**

Open source components are being tainted at their origin to enable immediate attacks once they are deployed into production. **This shifts adversary behavior from a wait-then-prey to a design-in-then-exploit style of attack.**

Not only is the attack landscape changing, but adversaries are automating their operations and becoming smarter and faster. According to a Ponemon Institute / ServiceNow survey, the majority of respondents (54%) agreed that attackers are outpacing enterprises with technology such as machine learning and artificial intelligence (AI). The same survey reports, "53% of respondents said that the time window for patching—the time between patch release and hacker attack—has decreased an average of 29% over the last two years. As AI-fueled attacks become more prevalent, we expect that window to shrink even further."[46]

Cryptomining using popular open source components

Historically, most successful breaches involved hackers taking control of applications and stealing data. While data theft continues to be lucrative, it's also risky. A successful thief must find someone willing to buy the data — which increases their risk of being caught. In light of this risk variable and the rapidly rising value of cryptocurrency, some cyber criminals have shifted gears and are now exploiting open source to steal computing resources to actively mine cryptocurrency.

Starting in March 2017, attackers had compromised systems running applications built with Apache Struts and installing backdoors, DDoS bots, cryptocurrency miners, or ransomware. While fixed versions of Struts were available, the hackers were targeting systems that had not yet received an update. It's estimated that the hackers walked away with at least $100,000 in cryptocurrency.[47]

A second instance of crypto hacking was discovered in December 2017. Researchers at F5 identified a sophisticated multi-staged attack that targeted internal networks with the NSA-attributed EternalBlue and EternalSynergy exploits. The attack appeared to be one of many campaigns exploiting servers vulnerable to the Apache Struts Jakarta Multipart Parser attack (CVE-2017-5638) that had been widespread since it was discovered in March 2017. The same attack exploited the DotNetNuke (DNN) vulnerability (CVE-2017-9822), disclosed in July 2017.[48] Successful breaches resulted in crypto mining malware being installed on corporate servers for purposes of mining (printing) new digital currency.

Yet another large scale crypto hacking operation inserted malware on vulnerable versions of the popular Jenkins continuous integration platform. The campaign netted an estimated $3.4 million in cryptocurrency for the hacking group.[49] Through basic queries to Shodan.com, researchers following this campaign identified over 25,000 servers on the internet that were susceptible to this attack. Hacks that result in the production of cryptocurrencies offer untraceable and easily converted windfalls.

A more sophisticated form of cryptojacking was discovered in August 2018. Members of the hacker community were targeting a serious Struts2 vulnerability with cryptomining malware injectors, but they added a twist. The new mining software scanned the vulnerable web application for existing cryptominers, removed them, and then proceeded to mine currency without interference or competition.[50] The behavior is nefarious, but points to the efficiency and competitiveness of hacker operations in 2018.

Chapter 2: Accelerating Innovation Drives Automated Consumption of OSS Components

Every organization depends on a software supply chain

To accelerate software innovation, every company — whether they know it or not — depends on a software supply chain. Software supply chains are comprised of thousands of open source suppliers (projects) that produce millions of parts (components and versions) each year. Supplier parts are then consumed hundreds of billions of times each year by software development teams. Teams then assemble their parts into applications. Finished applications are then deployed into production environments and managed by IT operations. Once in production, applications deliver value to customers and users in the form of a product or service.

Today 80% to 90% of every modern application is comprised of open source components.[51] Consumption of open source is so vast, that most organizations can not identify how many components are entering their software supply chains, where they are flowing, or where they might exist in production applications.

Echoing this reality, an April 2018 letter from the US Congress' Energy and

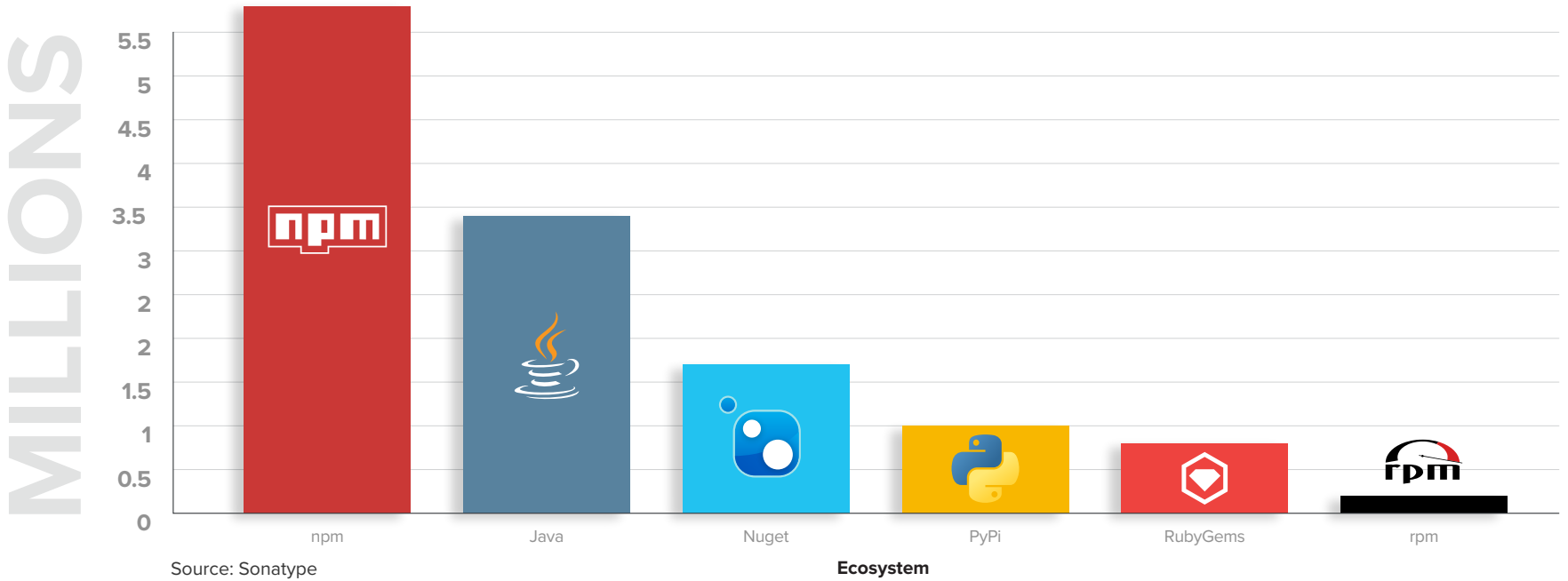
Commerce Committee to the Linux Foundation stated, "The widespread impact of the Heartbleed vulnerability through the deployment of a piece of OSS forced individuals and organizations outside of the information technology community to recognize what members within the community had long-known: **software is no longer written, but assembled."**[52]

This year's report finds that open source hygiene is inconsistent across supplier projects and individual components. Therefore, organizations cannot afford to blindly trust the quality of open source components flowing into development lifecycles. Also, as we will show in Chapter 4, **by actively governing the flow of open source components across the software supply chain, organizations can reduce the use of vulnerabile open source components by 50%.**[53]

Supply of open source is growing exponentially

The supply of open source components is growing at an extraordinary rate. The inventory of unique Java open source components grew from 2.0 to 3.5 million in the past year alone.[54] The availability of JavaScript releases grew from 3.0 to 5.5 million during the same period.[55] Python components grew from 870,000 to 1.4 million, NuGet releases for the .NET community grew from 900,000 to 1.7 million, and nearly one million RubyGems releases are now available.[56] [57] **This equates to almost 15,000 new or updated open source releases being made available to developers every day** of the year — up from 10,000 in our 2017 report.

Number of Releases per OSS Ecosystem



Source: Sonatype

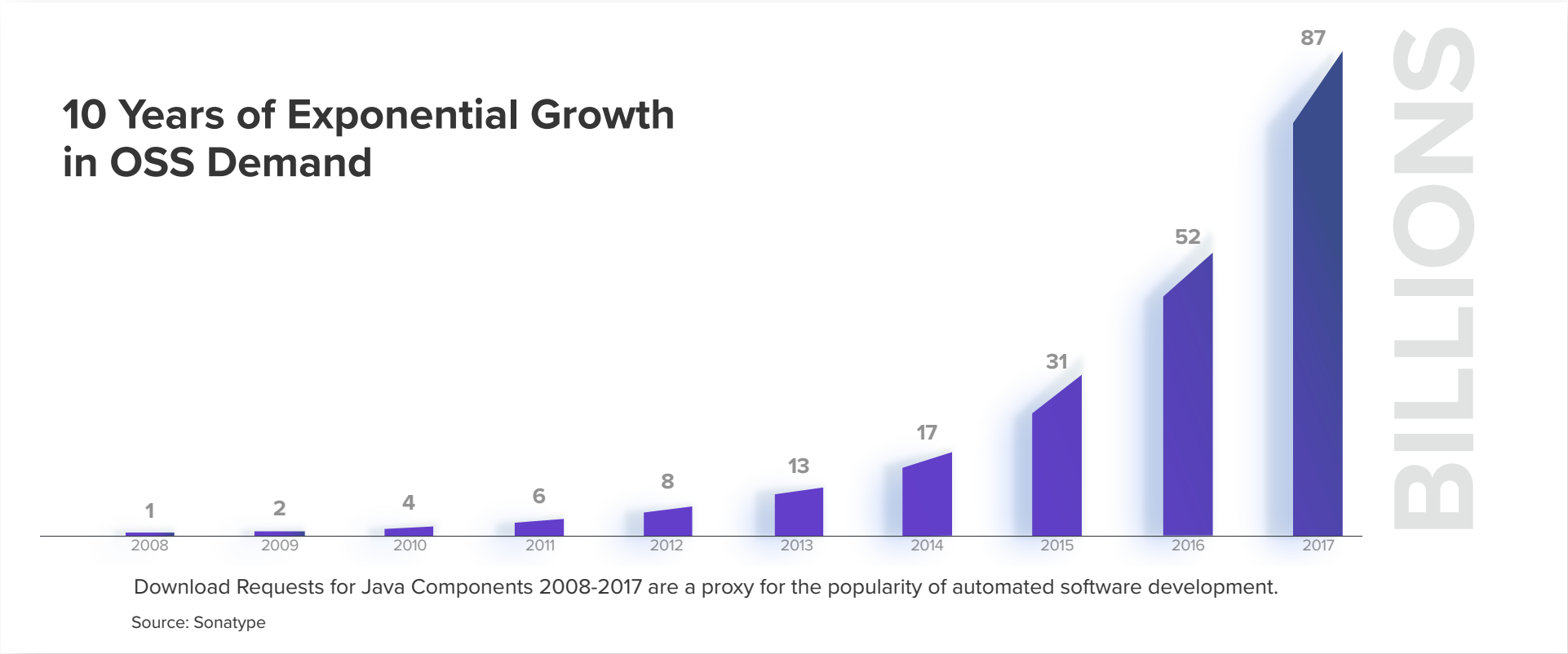
Automation accelerates the demand for open source

Fifty percent of the current S&P 500 will be replaced over the next ten years, according to Innosight's biennial corporate longevity forecast.[58] Turnover will be driven by a number of different circumstances, but highly disruptive technology startups will continue to be a leading factor. For those leading the innovation race, speed and efficiency are imperative and open source plays center stage.

The growing demand for innovation has accelerated implementations of automated software development pipelines while also driving open source consumption to new heights. In 2017, the number of **download requests for Java components from the Central Repository grew 68% year over year to 87 billion.**[59] While that growth is remarkable, a view into JavaScript component downloads demonstrates an even greater rush toward developer efficiencies.

In February 2017, npm COO and co-founder, Laurie Voss (@seldo) tweeted that they had witnessed 1.8 billion component downloads a week from the npm repository.[60] **By August 2018 downloads reached 6 billion per week — equating to a 235% increase in 16 months.**[61] To add further context, there are an estimated 12 million JavaScript developers in the world— meaning the average JavaScript developer is sourcing 500 open source components a week or 26,000 per year.[62]

Analysis of PyPI component downloads **by Python developers in 2017** was also significant. In late 2017, monthly downloads from the PyPI repository were averaging between 4.3 and 4.7 billion per month — or 52 billion on an annualized basis.[63]



Download Requests for Java Components 2008-2017 are a proxy for the popularity of automated software development.

Source: Sonatype

GitHub also plays home to a number of open source projects. According to TheHill.com, GitHub "hosts roughly 70 million open source projects. There are nearly 600,000 open source components on the repository, which are downloaded a combined 14 billion times a month".[64] [65]

Containers are also an active and growing part of an organization's software supply chain. In June 2018, Docker revealed that **development and operations teams pulled one billion containers from Docker Hub every two weeks**. This represents a run rate of 26 billion per year — a 334% increase over pull volume recorded in 2016.[66]

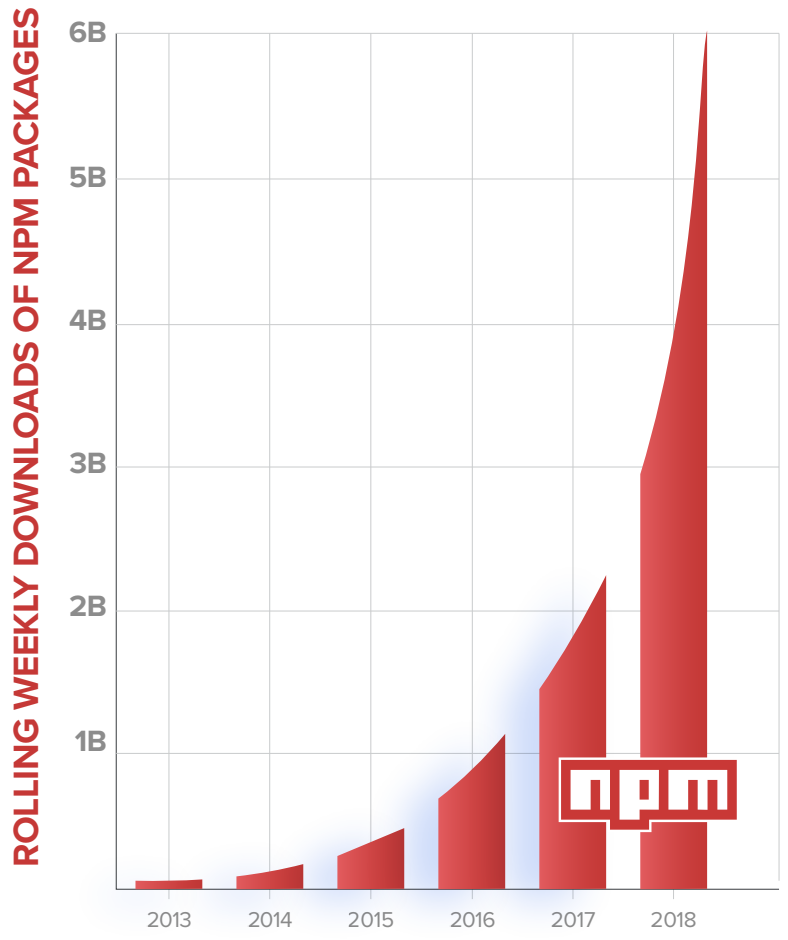
Exponential growth in the consumption of open source components and containers is *a* clear proxy for the adoption of automated software development tools and DevOps pipelines. The growth witnessed across these ecosystems are not simply a reflection of individual developer requests for components and containers, but are evidence of automated tooling that can generate hundreds or thousands of download requests per build.

Open source vulnerabilities are pervasive

Sonatype's analysis of various open source component ecosystems reveals a high percentage contain known security vulnerabilities. For example, analysis of 3.5 million Java components in the Central Repository reveals that 351,000 (10%) had at least one known vulnerability. In fact, Sonatype researchers identified over 3 million vulnerabilities across those components.[67]

By comparison, younger open source ecosystems also contain significant numbers of vulnerabilities. **There are over 509,000 npm, 158,000 PyPI, 118,000 RubyGems, and 57,000 NuGet releases with known vulnerabilities.**[68] As seen in the figure "Distribution of Vulnerability Scores", analysis by Sonatype reveals the severity of risk and exploitability for components across different ecosystems based on the Common Vulnerability Scoring System (CVSS).

According to researchers at Snyk, "One method of reducing the number of vulnerabilities included in an application is to conduct regular security audits...43.7% of open source maintainers say they have never had a security audit done on



Source: npm Inc., Laurie Voss (@seldo)

their code, and another 31.8% say they've only audited their code once or twice in the lifetime of the project".[69]

Extending the analysis to JavaScript and RubyGems open source projects hosted on GitHub, researchers identified 4 million known vulnerabilities in over half a million repositories in November 2017. Working with project owners, GitHub says that it remediated 450,000 of the 4 million vulnerabilities by December 2017. Still, 89% of the vulnerable projects were available to download, unchecked by developers, into software supply chains.[70]

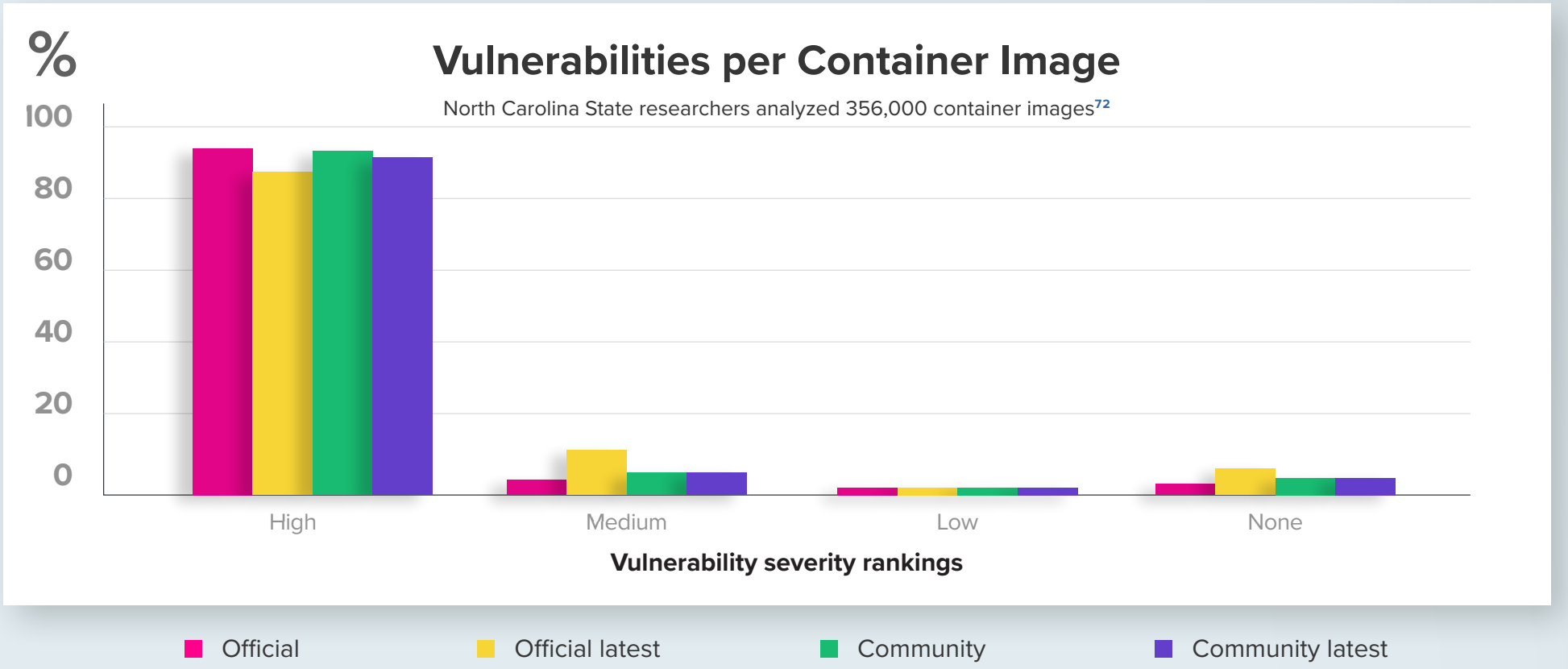
Distribution of Vulnerability Scores Per Ecosystem

Analysis of 19 million OSS components

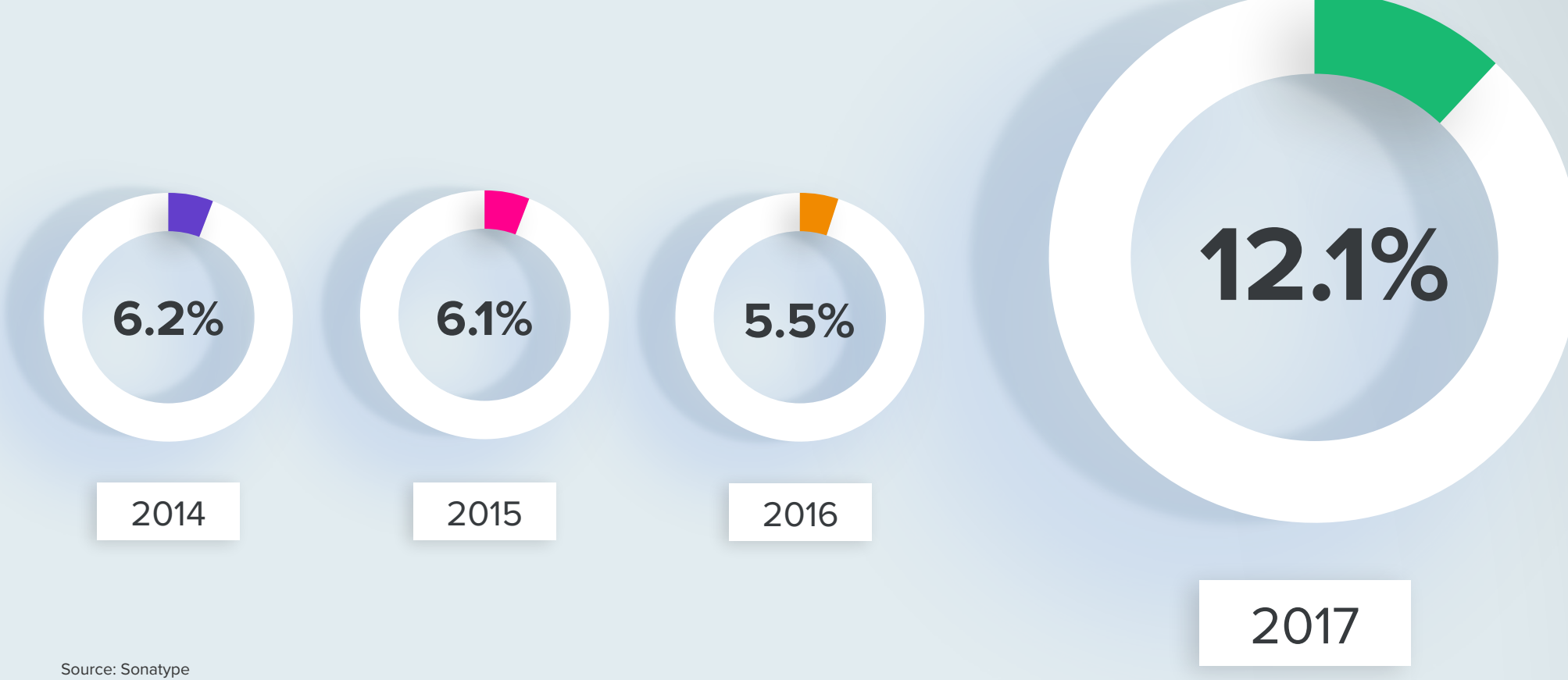
Source: Sonatype

Containers are not immune

The presence of vulnerabilities also applies to containers. According to the 2017 Study of Security Vulnerabilities on Docker Hub, from researchers at North Carolina State, **an analysis of 356,000 container images found: (1) both official and community images contain more than 180 vulnerabilities on average when considering all versions;** (2) many images have not been updated for hundreds of days; and (3) vulnerabilities commonly propagate from parent images to child images.[71]



Source: North Carolina State University, A Study of Security Vulnerabilities on Docker Hub



Source: Sonatype

Consumption of vulnerable open source hits record high

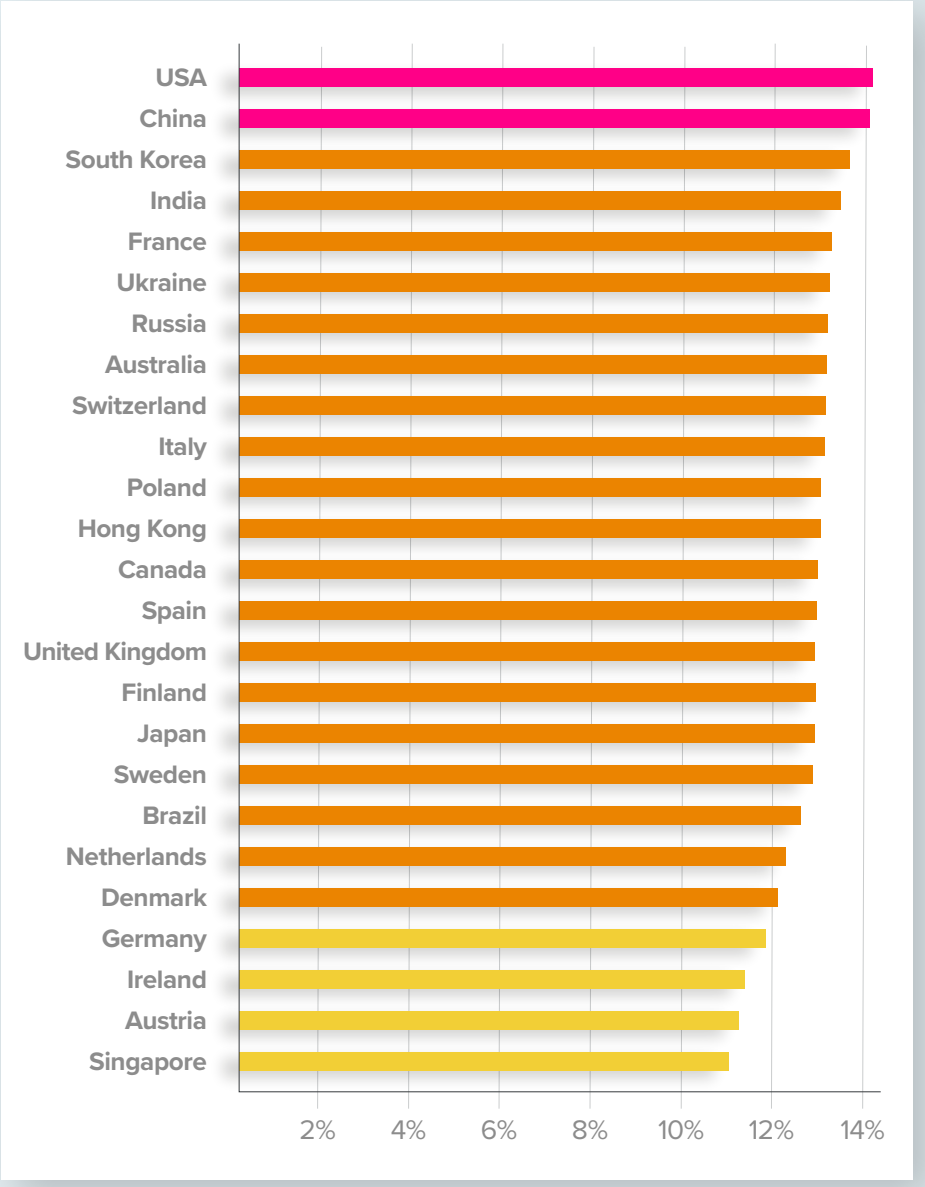
Because public repositories are immutable by design, vulnerable components are not proactively removed from inventory. Instead, it is incumbent upon development organizations to practice good hygiene when consuming open source components. Teams with suboptimal hygiene inevitably consume open source releases with critical vulnerabilities and risky license types. Over the past four years that Sonatype has been reporting the percentage of known vulnerable downloads from the Central Repository, 2017 was the worst, by far.

Between 2014 and 2016, the percentage of vulnerable components downloaded dropped from 6.2% to 5.5%. **For 2017, Sonatype researchers noted 12.1% (1 in 8) of Java downloads contained known vulnerabilities -- a 120% year over year increase.**[73] This just four years after the notorious OpenSSL Heartbleed vulnerability was discovered -- and in the same year the Equifax breach was announced.

This year's report, again, studied the open source consumption patterns of 7,500 organizations. **The average enterprise downloaded 170,000 Java components in 2017, up 36% year over year.** From this set, over 150 organizations -- across 32 countries -- topped the charts at well over 1,000,000 Java components downloads annually.

The 7,500 organizations consumed an average of 3,500 unique components -- up 10% year over year. Deeper analysis of the downloads revealed 11.1% were known vulnerable, up an alarming 91% year over year.[74]

Vulnerable Downloads per Country



Source: Sonatype

Downloads of vulnerable components were assessed by country for the first time in this year's report. No country fared well in Sonatype's analysis of the top 25 countries consuming open source components. Singapore had the lowest percentage of vulnerable component downloads at 10.9% — representing 1 in 9 downloads with a security defect. At the high end, the **United States and China were at 14.1% and 14.0% respectively — representing 1 in 7 downloads in those countries were vulnerable.**[75]

Chapter 3: Dirty Rivers Flow Downstream Leading to Dirty Reservoirs

The race to out-innovate one's competition has led to high performing organizations chasing increased deployment velocities but often ignoring the quality of parts being used to manufacture their applications. It was 2003 when Bruce Schneier (@schneierblog) penned, "Today there are no real consequences for having bad security, or having low-quality software of any kind. Even worse, the marketplace often rewards low quality. More precisely, it rewards additional features and timely release dates, even if they come at the expense of quality."[76]

As nimble organizations deliver new innovations, adversaries are also upping their game. **They have increased scale through automation, improved breach success through precision targeting, and — as noted in Chapter 1 — have shifted battlefield tactics to inject known vulnerabilities, defects, and malware into software supply chains at the point of inception.** Today, the stakes are higher than ever. Cybersecurity Ventures predicted that the cybercrime "industry" will exceed $6 trillion by 2021, making it more profitable than global drug cartels.[77]

Roads and bridges are falling down: NVD and CVE

The U.S. government maintains a repository of standards based vulnerability management data: the National Vulnerability Database. For years, the data has enabled the automation of vulnerability management, security measurement, and compliance for security and open source governance teams. But as more vulnerabilities are discovered, the database has not been able to keep pace. Significant numbers of known vulnerabilities are missing or the information provided is incomplete or published long after a disclosure.[78] Furthermore, the database that has long served security professionals does not cater to the needs of the developer community who are tasked with understanding and remediating the vulnerabilities that impact code in their applications.

In-depth research led by **Sonatype uncovered 1.3 million vulnerabilities in open source components that do not have a corresponding CVE advisory in the public NVD database.** Along similar lines, Risk Based Security recently reported that, "By the numbers, despite CVE/NVD making efforts to address coverage issues after industry and Congressional pressure, 2017 shows that they are actually falling further behind. Along with the drop in quality of CVE entries, this firmly demonstrates that CVE/NVD is no longer 'good enough' for your organization's vulnerability management." RBS claims that its own database contains more than 57,000 publicly disclosed advisories that are not present in the National Vulnerability Database.[79]

Further evidence of the NVD's inability to keep pace with vulnerability updates was recognized by Bill Ladd in October 2017 when he noted the Chinese National Vulnerability Database (CNNVD) had over 1,700 more records. Ladd also noted average time between vulnerability disclosure and inclusion was 33 days for the NVD and 13 days for the CNNVD.[80] With exploits now occurring in under three days post-disclosure, **organizations relying solely on NVD and CNNVD inclusion details will leave 30-day and 10-day windows, respectively, for adversaries to disrupt their operations.**

Reports on the responsiveness of the Chinese efforts vary. In March 2018, it was reported that the CNNVD was being manipulated: "the delays in CNNVD publishing have an impact in the greater digital ecosystem. Last year, publication of the Microsoft Office vulnerability CVE-2017-0199 came out 57 days late on the Chinese database. In the meantime, a Chinese advanced persistent threat group exploited the vulnerability in cyber operations against Russian and Central Asian financial firms."[81]

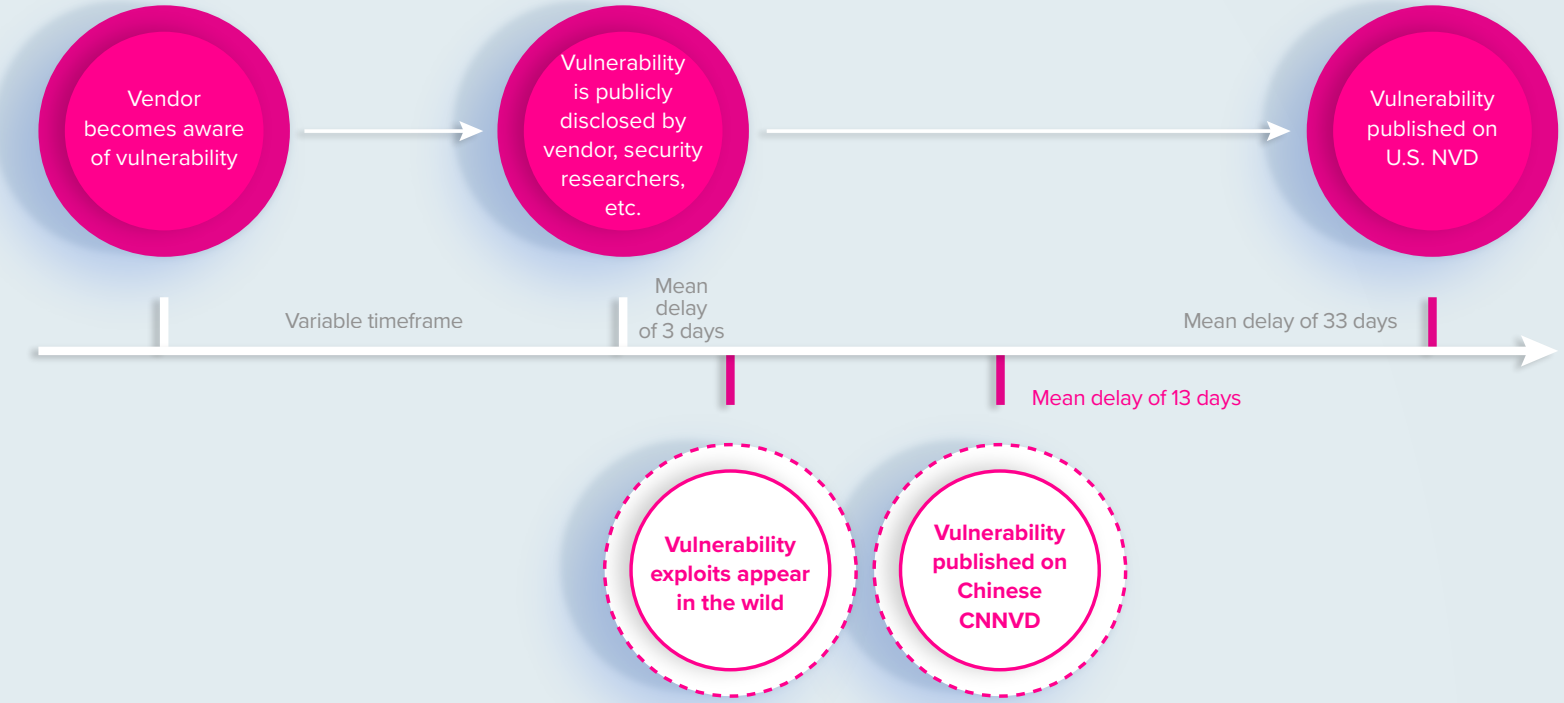
As open source software continues to accelerate to its zenith of value, the underlying fundamentals of the ecosystem and the infrastructure supporting it

are at risk. Open source researcher, Nadia Eghbal remarked, **"Digital infrastructure has a free rider problem. Resources are offered for free, and everybody (whether individual developer or large software company) uses them, so nobody is incentivized to contribute back, figuring that somebody else will step in."**[83] The world Eric Raymond imagined — that is, "with enough eyeballs all bugs are shallow" — is straining under its own success.[84]

Industry lacks meaningful open source controls

The most common way to introduce controls is through the application of open source governance policies across a software supply chain. When the 2,076 participants of the 2018 DevSecOps Community survey were asked if their organization employed open source governance policies, 64% responded positively. But that percentage degraded further when participants were asked if they followed the policy. Only 65% with an OSS governance policy admitted to following it, leaving the effective rate at 41%. This means 4-in-10 organizations have control over vulnerable components flowing downstream.

Further evidence of the lack of cybersecurity hygiene was revealed by **62% of survey participants who admitted to not having meaningful controls over what open source components are used in their applications.** That is, they did not record a complete software bill of materials of open source components used in applications.[85]



Source: www.recordedfuture.com/chinese-vulnerability-reporting/, Sonatype[82]

Repository managers host a larger percentage of vulnerable components

Within a software supply chain, repository managers are commonly used in DevOps toolchains to house open source components needed in application development. Where internet-based repositories (e.g., npmjs.org, NuGet Gallery, and PyPI.python.org) are the central warehouses of a software supply chain, repository managers (e.g., Sonatype Nexus Repository Manager, JFrog Artifactory) act as local, private warehouses for development teams. A single repository manager may contain multiple repositories, each of which manage components from a different open source ecosystem.

In 2017, 2.08% of the downloads from the Central Repository were triggered by repository managers.[86] While the percentage is small, keep in mind that a repository manager will only download a component once. Once cached, future downloads of that specific component are unnecessary.

For this year's report, Sonatype analyzed 161,000 repositories used to host components from open source ecosystems including Java, npm, NuGet, PyPI, RubyGems, etc. The average repository contained 955 open source components of which 172 (18%) were identified with at least one known security vulnerability — up from 7.2% as reported in last year's report.[87]

Container usage soars while vulnerabilities proliferate

The 2018 DevSecOps community survey also inquired about container security practices. While noted in Chapter 2 that **container downloads surged to an average of one billion every two weeks, only 42% of survey respondents said their organizations leverage security products to identify vulnerabilities in containers.**[88] As the volume of container use continues to surge, the need for more automated and systematic methods of applying security updates to container images will be critical to maintaining quality across software supply chains.

Chapter 4: Automation Eats the World

Automate faster than evil

In her 1983 paper, Ironies of Automation, Lisanne Bainbridge detailed where human and computer assisted approaches worked best or struggled. Bainbridge remarked that "overall control performance was better with manual [human] control of a single loop, but was also better with [computer assistance] in complex environments", suggesting that aid is best applied at higher workloads. She then went on to say, "humans working without time-pressure can be impressive problem solvers. The difficulty remains that they are less effective when under time pressure."[89]

Managing software supply chains today requires both human and computer based aid. Modern software supply chains can only operate safely when protected with automated security and quality assessments of these upstream open source components and containers.

This sentiment was echoed in Forrester's Top Recommendations For Your Security Program (March 2018) where analysts advised, "Automate faster than evil does. If you thought your security team struggled with alert volume — and alert fatigue — then you haven't prepared for the volume of attacks coming at you soon. Attackers will use the same building blocks of Artificial Intelligence (AI) that you are from natural language processing to Machine Learning (ML) and automation to improve the scale and efficacy of every kind of malicious activity. **Manual methods to detect, investigate, and respond to threats will guarantee failure in the near future.**"[90]

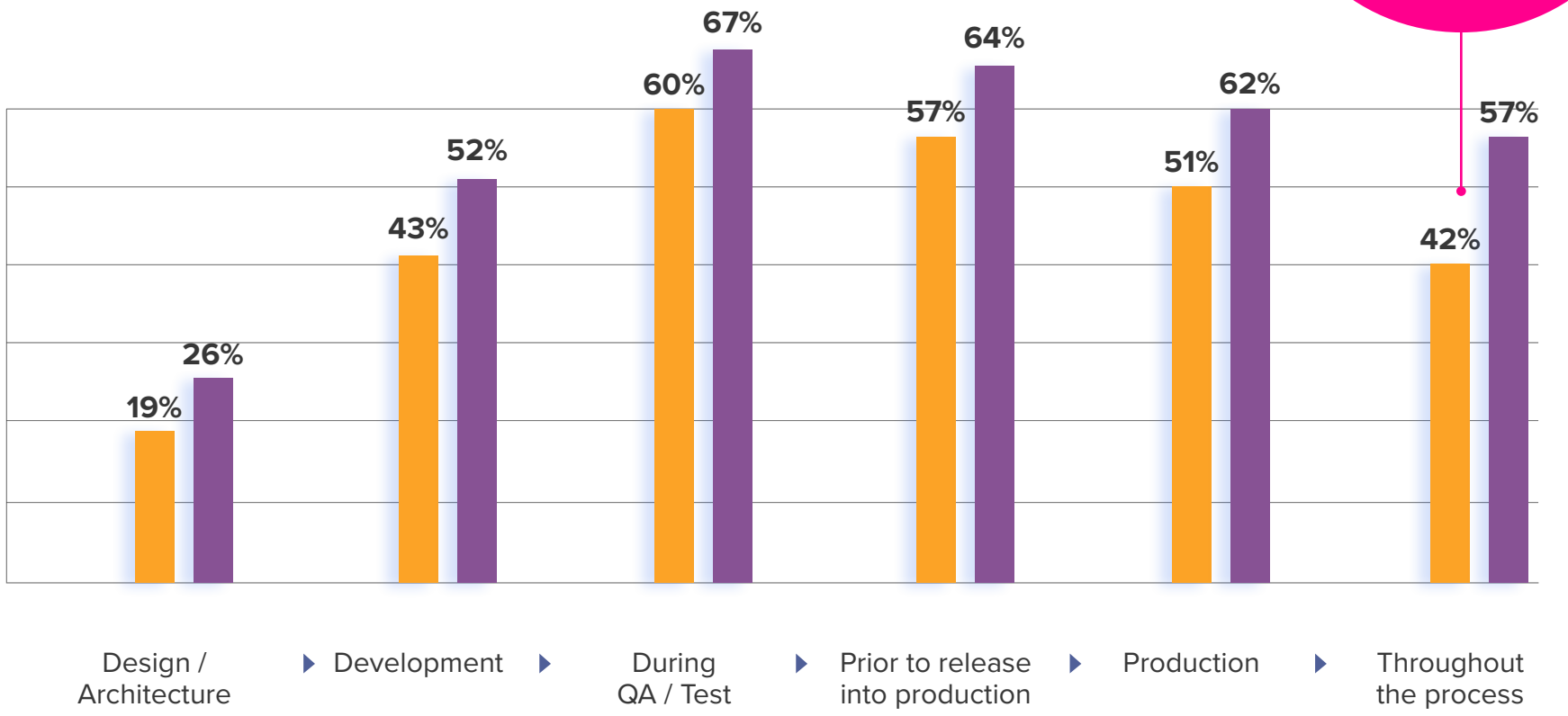
This chapter explored increased investments in automation, the value of precision when automating component analysis, and benefits of managed supply chains over unmanaged environments.

Increasing investments in automated security

The DevSecOps Community survey revealed that more organizations had introduced automated security checks throughout the software development lifecycle in 2018 compared to the previous year. For mature DevOps organizations, the survey noted a 15% year over year jump to 57% in automated security that was implemented throughout each stage of the software development lifecycle. The top three investments for automated security noted by survey participants were web application firewalls, container security, and open source governance.[91]

Where is Automated Application Security Performed?

Mature DevOps practices ramped their investment in automated security by 15%.



Source: Sonatype

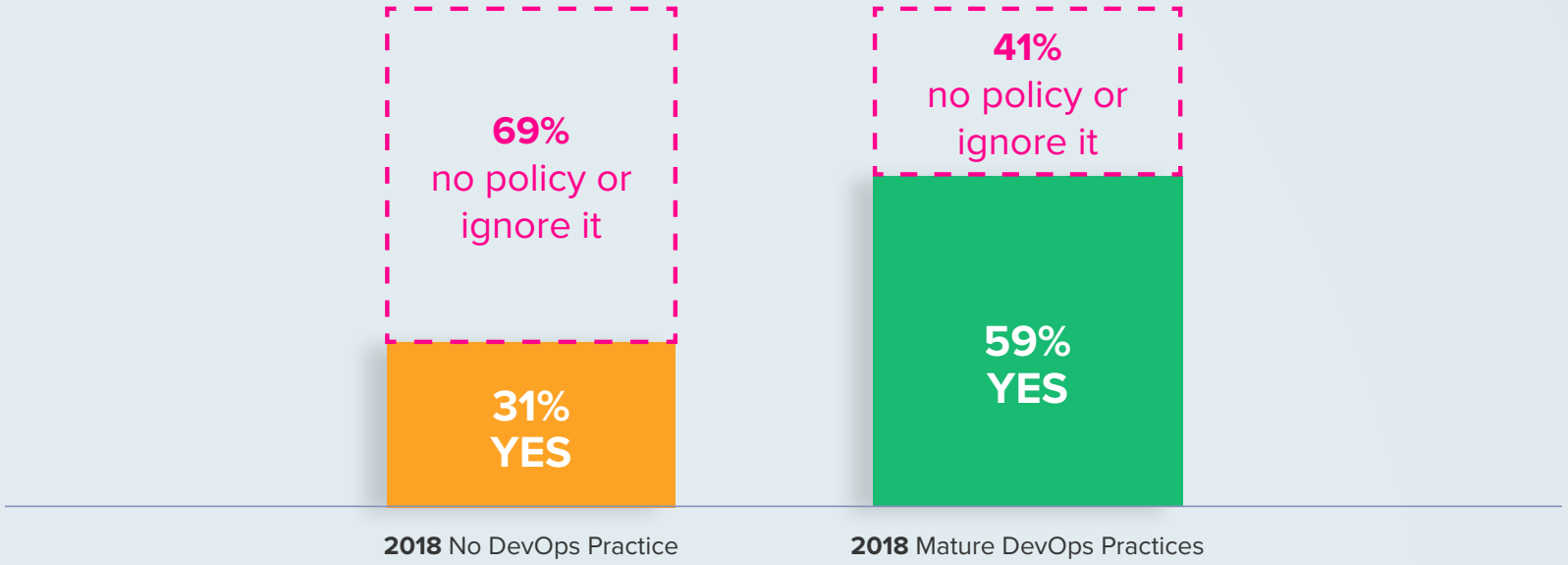
2017 Mature DevOps Practices

2018 Mature DevOps Practices

Automation of open source policies makes it harder to ignore

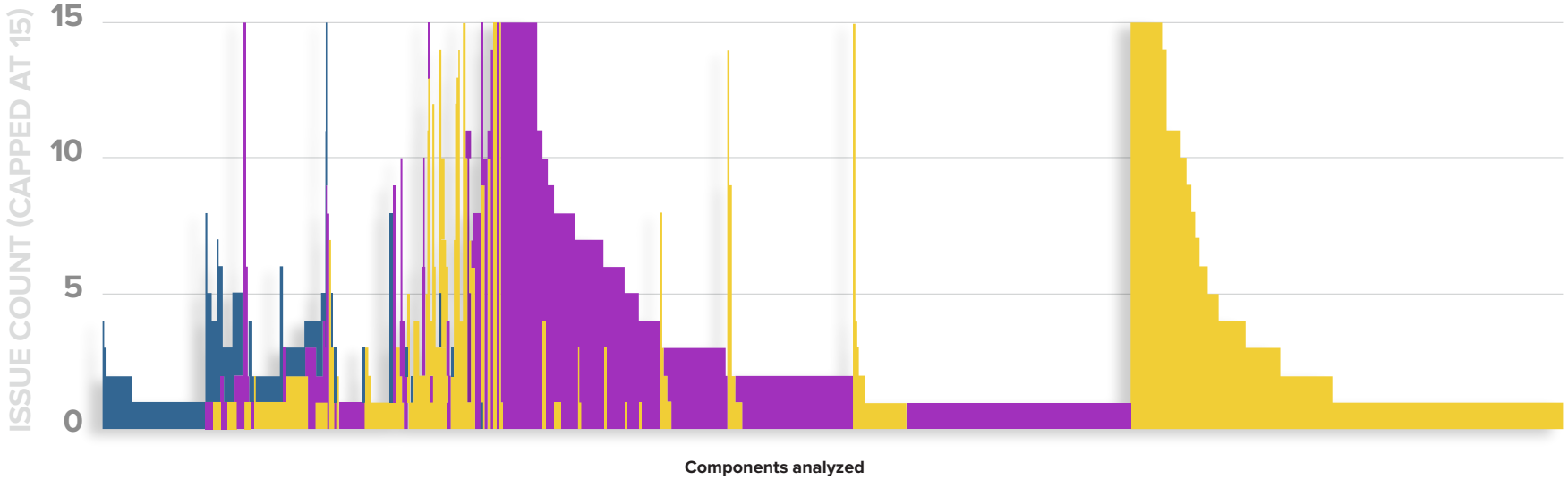
DevSecOps Community Survey participants also revealed that automation is difficult to ignore. As noted in Chapter 3, 64% of the survey participants employed an open source governance policy. Deeper analysis of this result showed that 58% of those with no DevOps practice and 77% of those with a mature DevOps practice had implemented OSS governance policies. When asked how many of the participants ignored the policy, 46% with no DevOps practice vs. 24% with mature DevOps practices chose to ignore their policies. **Therefore, effective compliance rates for OSS governance policies were nearly double for those with mature DevOps practices (59%) where more automation is present versus those with no DevOps practice (31%) with lower rates of automation.**[92]

Automated Policies are Difficult to Ignore



The 2018 DevSecOps Community Survey asked "do you follow your organization's open source policy?" OSS governance policies were nearly double for those with mature DevOps practices (59%) where more automation is present versus those with no DevOps practice (31%).

Automating OSS Policies Requires Precision



True Positive: CPE in coordinate name False Positive: CPE in coordinate name False Negative: CPE not in coordinate name

Source: Sonatype

Automation of open source governance requires precision

A number of free and open source tools rely on CPE (Common Platform Enumeration) based matching to discover vulnerabilities in components. CPEs are distributed with CVEs in the National Vulnerability Database, which to the novice researcher or developer adds an air of credibility. While CPE data originated as helpful information, reliance on this data today can lead to significant levels of false negatives and false positives.

Sonatype researchers analyzed 6,000 open source components to understand the efficacy of CPE based vulnerability matching. The researchers identified 1,034 true positives, 5,330 false positives (when CPE was part of the coordinate name), and 2,969 false negatives (when CPE was NOT in the coordinate name).

When it comes to using open source components to manufacture modern software, the bottom line is this — precise intelligence is critical. Tools that lack precision cannot scale to the needs of modern software development. Inaccurate and/or incomplete data will leave organizations to deal with vulnerabilities, licensing, and other quality issues in a manner that adds unnecessary research and unplanned rework.

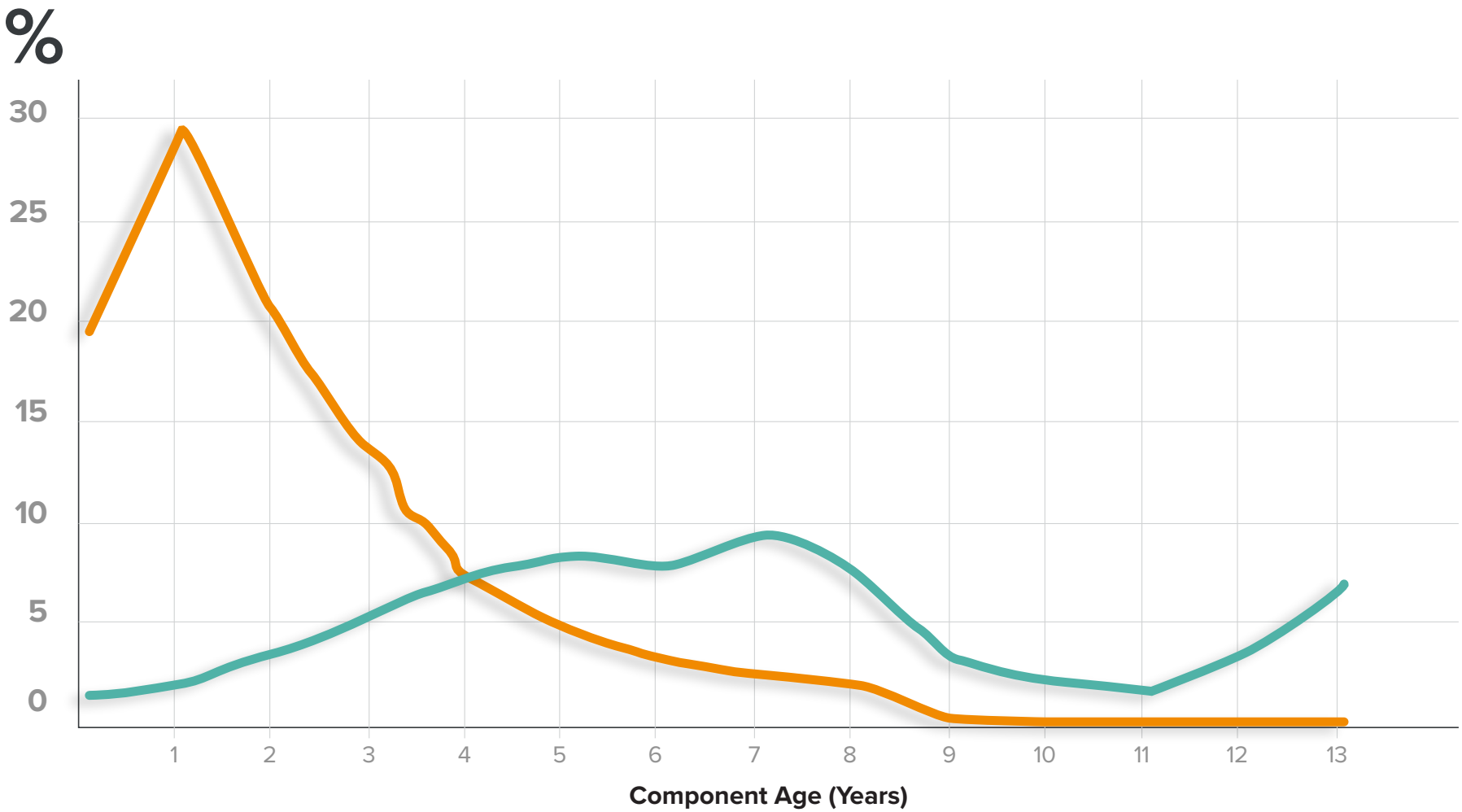
Automating open source governance at scale requires precise identification of vulnerable components and safer alternatives in order to accelerate remediation efforts. In all cases, once vulnerabilities are identified and remediation priorities are defined, human intervention [by software developers] is required to remediate issues. In the near future, automated self-healing techniques may further expedite remediation.

Guiding open source policies: newer components make better software

Analysis of 19.6 million open source components by Sonatype reveals that the latest versions of components have the lowest percentage of known security defects. Components under three years in age represented 68% of components across Java, NuGet, PyPI, Ruby, and npm ecosystems. Among these components, the average security defect rate was 2.3%. By comparison, **components between 3 and 8 years old (31% of the overall population) had an average security defect rate of 7.6% — representing 229% greater vulnerability levels.**[93]

Newer Components Make Better Software

Source: Sonatype analysis of 19 million open source components from Java, NuGet, PyPI, Ruby, and npm ecosystems

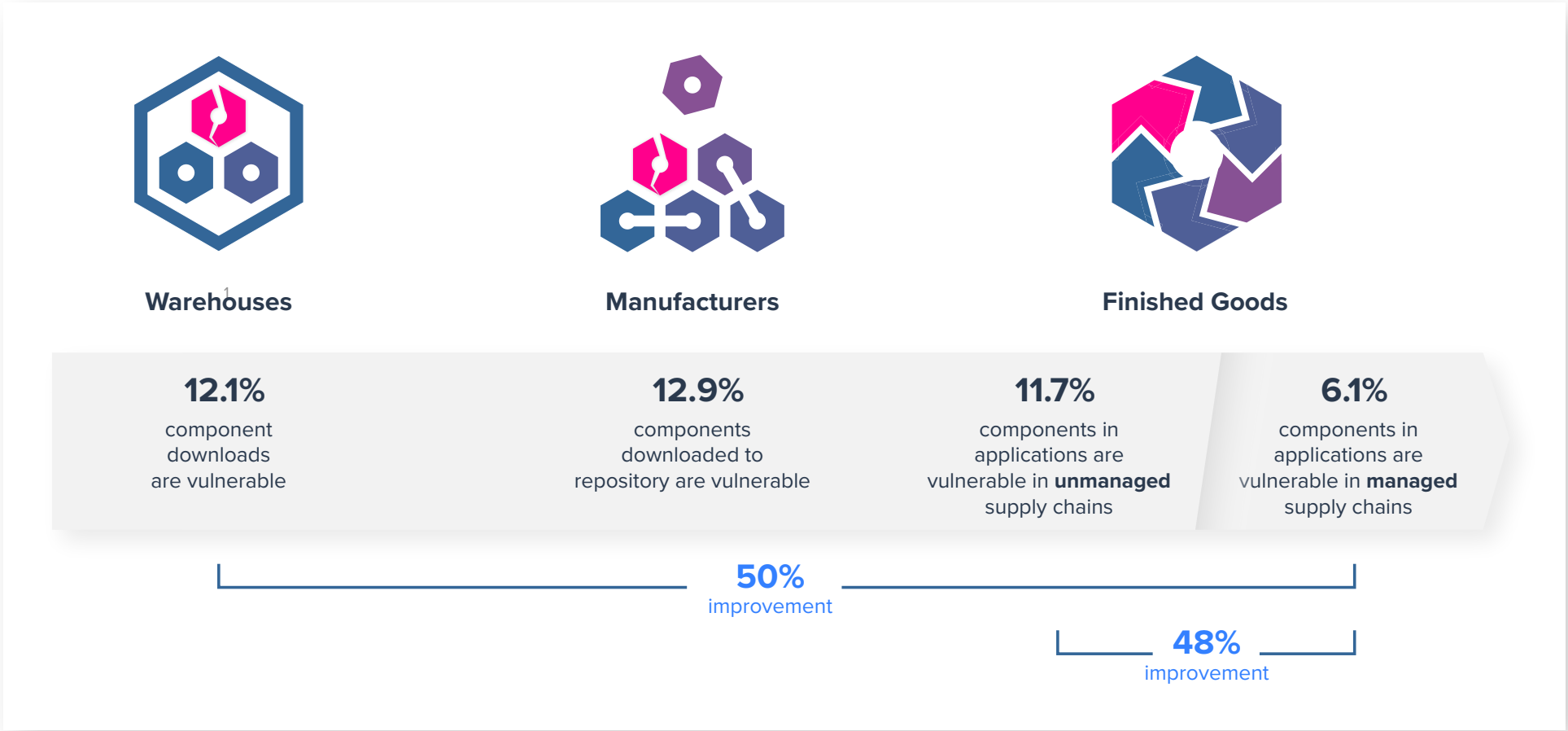


As components age and receive more attention from developers and security researchers, more vulnerabilities are discovered. No software is perfect forever. Therefore, better OSS component selection, due to analysis and governance processes, will improve the quality of a finished application, it can also reduce the number of break-fixes and unplanned work.

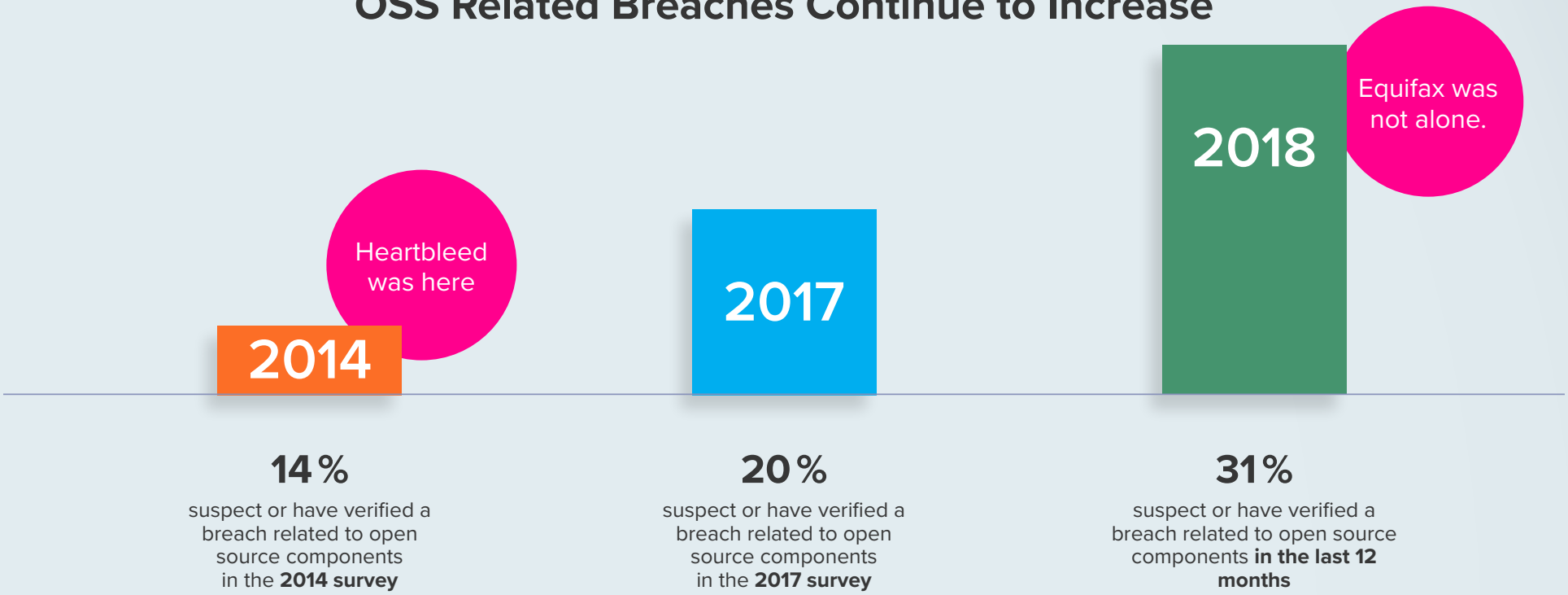
Vulnerabilities make their way into production applications

Sonatype's analysis of 500 applications in 2018 found the median number of open source components in an application was 127, while a few maxed out at over 5,000. From this set of applications, the average percentage of components with a known security vulnerability was 11.7%.[94] **The slight delta between the 12.1% of global component downloads and the 11.7% in this set of 500 applications demonstrates that vulnerable open source components are flowing freely through *unmanaged* software supply chains.**

According to another study, Automated Code Analysis: Web Application Vulnerabilities (2017), 94% of 33 web applications tested, contained a high-severity flaw and 85% carried an exploitable vulnerability.[95] A broader study of the 700 most popular applications in the Google Play Store revealed that 1 in 5 contained open source components with known security vulnerabilities.[96] While some have argued that downloads of known vulnerable components don't equate to



OSS Related Breaches Continue to Increase



their use, the evidence of their presence in production applications is undeniable and represents an increased likelihood of successful breaches.

Managed software supply chains are 2x more secure

A separate analysis by Sonatype of more than 25,000 applications in 2018 revealed over 137,209 distinct Java open source components being used within managed software supply chains. From this population, 8,350 (6.1%) of the Java components had at least one known vulnerability. The total number of vulnerabilities associated with these components totaled 21,407 or an average of 2.56 per vulnerable component.[97]

Globally, 12.1% of all components downloaded from the Central Repository had at least one known security vulnerability. As seen on the graphic on page 27, applications *analyzed* from unmanaged software supply chains showed only a slight reduction in the percentage of vulnerable OSS components downloaded compared to those used inside finished applications. By comparison, **organizations automating open source governance as part of a managed software supply chain practice reduced the percentage of vulnerable components used in finished applications by 50%.**

Open source related breaches grow to record levels

The DevSecOps Community Survey for 2018 revealed a **55% year over year increase in participants confirming or suspecting breaches tied to security vulnerabilities in open source components.** The survey also compared responses from 2014 — the year the OpenSSL Heartbleed vulnerability was disclosed — to find a 121% jump in confirmed or suspected open source breaches over the past four years.[98] As the number of vulnerable open source component downloads grow (Chapter 2), so have the breaches.

Chapter 5: The Rise of Regulating Software

The Rising Tide of Policies and Regulation

When it comes to software supply chains and cybersecurity hygiene, industry has failed to regulate itself. The incentives simply don't exist for self-governance in the face of pressures to innovate and maintain competitive differentiators.

More specifically to software supply chains, it is an immense challenge for any organization to self-regulate behavior for which they are barely aware. Attention was lax at 13 billion download requests witnessed five years ago and even **now as consumption volumes have increased 20x, vulnerability rates are up 2x, and breaches have increased 121%, insensitivity remains.**

As reliance on unreliable software grows, internationally renowned security technologist Bruce Schneier also believes there is no better alternative to ensuring cyber-security safety than government regulation. Rather than fall subject to regulations driven from the national level down, Schneier recently called for industry to participate in the definition of regulations. In November 2017, Schneier advised, "As internet security becomes everything security, all security has strong technological components. We'll never get policy right if policy makers get technology wrong."[99]

In the year since our last State of the Software Supply Chain Report, software regulation and liability discussions took center stage across the governments and global headlines.

United States Congress

In June 2017, U.S. Representative Anna Eschoo (D-CA) and Susan Brooks (R-IN) introduced the *Promoting Good Cyber Hygiene Act.* The Act called for "software updates to patch known vulnerabilities; using strong, secure passwords; and utilizing modern firewall and security techniques".[100] Patching known vulnerabilities has often been recognized as the fastest way to lower security risk.

Two months later, U.S. Senators Mark R. Warner (D-VA) and Cory Gardner (R-CO), co-chairs of the Senate Cybersecurity Caucus, along with Sens. Ron Wyden (D-OR) and Steve Daines (R-MT) introduced bipartisan legislation called the *Internet of Things Cybersecurity Improvement Act of 2017.*[101] According to a fact sheet released at the time, **"This legislation is aimed at addressing the market failure by establishing minimum security requirements for federal procurements of connected devices."**[102] The proposed legislation would require vendor commitments, including: (1) ensure devices don't contain known security vulnerabilities when shipped, (2) ensure proper disclosure of new security vulnerabilities discovered within their devices, and (3) prepare remediation plans for any IoT device where known vulnerabilities have been discovered.

While legislation from the Senators was clearly aimed at consumer protections and privacy, it also focuses on quality, safety, and regulatory standards applied to every other major manufacturing industry (i.e., do not ship products with known defects). The legislation specifically calls for vendors selling IoT devices "to provide written certification that the device does not contain, at the time of submitting the proposal, any hardware, software, or firmware component with any known security vulnerabilities or defects."

U.S. Food and Drug Administration

Guidance issued by the FDA to medical device manufacturers in 2005 foreshadowed problems related to the need for a software bill of materials and cybersecurity hygiene. The report identified Software of Unknown Pedigree

(SOUP) where "some or all of the software contained in a Software Device may have been obtained by the submitter from a third party. The type and quality of documentation that accompanies this software can vary considerably. Software for which adequate documentation may be difficult to obtain is referred to as SOUP."[103]

More recently, in October 2017, Dr. Suzanne Schwartz of the FDA penned a blog warning of cybersecurity risks implanted in medical devices.[104] Schwartz called for "a need to balance protecting patient safety [while] promoting the development of innovative technologies and improved device performance." She then went on to recommend that medical device manufacturers take **"a total product lifecycle approach, starting at the product design phase when we build in security to help foil potential risks, followed by having a plan in place for managing any risks that might emerge, and planning for how to reduce the likelihood of future risks."**[105]

Next, the FDA introduced the Medical Device Safety Action Plan (April 2018) where they called for considerations of "potential new premarket authorities to require firms, on the front end, to: (i) build the capability to update and patch device security into a product's design and to provide appropriate data regarding this capability to FDA as part of the device's premarket submission; and, (ii) **develop a 'Software Bill of Materials' that must be provided to FDA as part of a premarket submission** and made available to medical device customers and users, so that they can better manage their networked assets and be aware of which devices in their inventory or use may be subject to vulnerabilities. In addition, availability of a 'Software Bill of Materials' will enable streamlining of timely postmarket mitigations."[106]

U.S. Department of Defense (DoD)

Software supply chain attacks are an increasing concern for government procurement officers in the Pentagon. According to ExecutiveGov.com, Pentagon spokesperson Maj. Audricia Harris said **the Defense Department "is examining ways to designate security as a metric within the acquisition process**" adding that "[the] department's goal is to elevate security to be on par with cost, schedule and performance."[107]

In August 2018, MITRE delivered supply chain and cybersecurity guidance to the U.S. Department of Defense in a report titled, Deliver Uncompromised. The report explored the composition of modern software, new software attack vectors, and the introduction of software liability. The MITRE authors advised, "Much of the software used in contemporary systems has open source components with uncertain pedigree or provenance. **Should adversaries insert malicious functionality into open source components of software code or exploit latent vulnerabilities, the resulting corruption of the software tool chain can have pervasive and durable effects;** these may not result in immediate harm but can be activated at the time chosen by an adversary. Hence, static assessment or static certification by itself is insufficient to ensure protection."[108]

In order to accelerate industry cooperation with the DoD and strengthen responses to malicious open source components being used, MITRE also recommended that the **"U.S. Department of Defense should consider when to require a Software Bill of Materials (SBOM) and can encourage Congress to hold hearings on whether to change the law on software immunity**—perhaps for certain areas of commerce related to national security and industry and key infrastructure. **DoD can lead efforts at litigation reform to manage liability risks** and therefore to encourage positive industry behavior and facilitate timely government actions."[109]

U.S. Department of Health and Human Services (HHS)

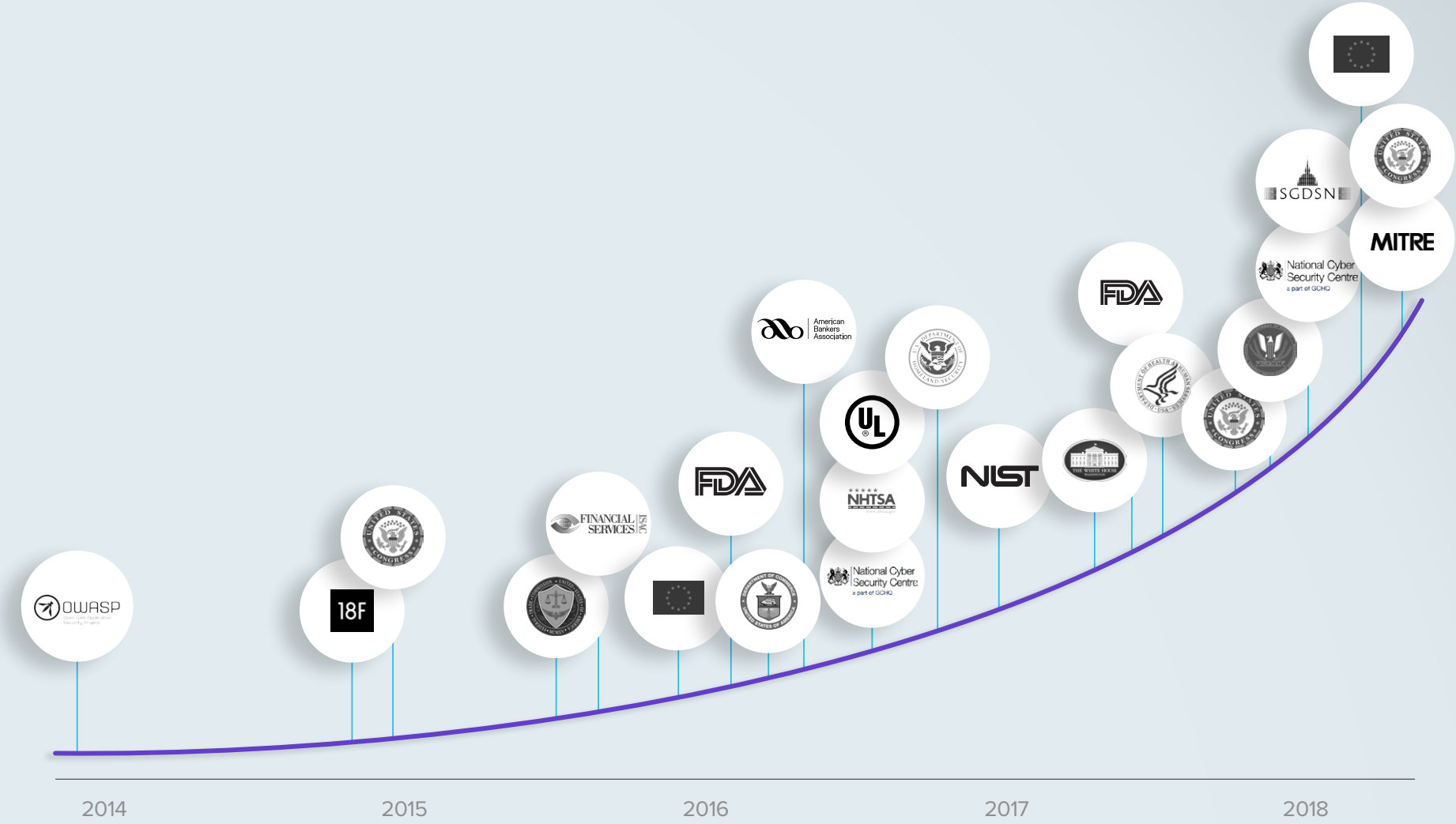
In November 2017, U.S. Congressman Greg Walden (R-OR) sent a letter to the U.S. Department of Health and Human Services (HHS) requesting they also **convene a sector-wide effort to establish a plan of action for creating, deploying, and leveraging software bill of materials (SBOM)** — similar to the guidance offered by the U.S. Food and Drug Administration and to the Department of Defense.[110]

U.S. Department of Commerce

The National Telecommunications and Information Administration (NTIA) is an agency of the U.S. Department of Commerce. In July 2018, Allan Friedman, director of cybersecurity for the NTIA, kicked off a multi-stakeholder initiative to **"scope out the idea of software transparency and the problems it seeks to solve, including how Software Bill of Materials data might be shared."**[111]

The Rising Tide of Regulation and Software Liability

Governments are increasing their attention on software supply chain risks



Source: Sonatype

Note: organizations listed multiple times introduced new or strengthen policies at different points along the time line.

Speaking at Black Hat in August 2018 Friedman shared more about this initiative. He explained that the goal of the Software Bill of Materials initiative is "for software and IoT vendors to share details on the underlying components, libraries, and dependencies with enterprise customers...This transparency can catalyze a more efficient market for security by allowing vendors to signal quality and giving enterprise customers key knowledge — you can't defend what you don't know about."[112]

Mandating a Software Bill of Materials not only benefits enterprise customers and consumers, it also aids vendors in expediting remediation efforts for products where vulnerability defects are discovered. Vendors who maintain knowledge of what software components were shipped or deployed can quickly update them or advise on security risks when defects are discovered.

Federal Energy Regulatory Commission (FERC)

The Federal Energy Regulatory Commission (FERC) proposed new rules in January 2018 to mitigate cybersecurity risks associated with the supply chain. According to the FERC, "the global supply chain provides the opportunity for significant benefits to customers, including lower cost, interoperability, rapid innovation, and a variety of product features, but that it also enables opportunities for adversaries to directly or indirectly affect the management or operations of generation and transmission companies in a manner that may result in risks to end users, such as insertion of malicious software".[113]

These rules followed October 2017 comments delivered by Devon Streit, the Energy Department's deputy assistant secretary for infrastructure security and energy restoration, she recommended utilities assume "that either you are going to be compromised, or if you really are going to be honest about this, that you already have been compromised, and take it from there."[114]

European Union

When the innovation race is being run without proper oversight, getting to the finish line safely will require greater (and faster) care. That's set to be a major challenge for organizations developing software under the EU General Data Protection Regulation (GDPR) that went into effect in May 2018. **GDPR's Article 25, which mandates data protection measures be implemented "by design and by default"**, make it clear that privacy and security must become ingrained in every element of software being developed today. Organizations failing to follow the rules and who design in known software vulnerabilities that end up helping hackers steal sensitive consumer data, will be on the hook for seriously big fines: up to €20 million, or 4% of global annual turnover — the greater of the two.[115]

France

In February 2018 **French lawmakers proposed putting security liability in hands of product suppliers and making those companies liable for the security of products for as long as they are commercially available.**[116] According to Lukasz Olejnik , one of the solutions proposed by the French government to reduce cyber security risks related to poor cybersecurity hygiene would be "to release source code and documentation after an end of support date."[117]

United Kingdom

As Europe geared up for GDPR compliance, the UK government announced in January 2018 that organizations working in Britain's most critical services like energy, transport, water, health, and digital infrastructure **could be fined up to £17 million (USD $22M) if they fail to demonstrate that their cyber security readiness — reflecting increasing support for software liability linked to cybersecurity hygiene.**[118]

Two months later, in March 2018, the UK government's Department for Digital, Culture Media and Sport, released the *Secure by Design: Improving the cyber security of consumer Internet of Things* report. The report advised development teams at IoT device manufacturers to **embed security in the software design process rather than bolt them on as an afterthought.** In her introduction to the report, Margot James, UK Minister for Digital and Creative Industries advocated, "we must also reduce the burden on end users by embedding effective cyber security practices at every stage of a connected product's life cycle." The report advises that "knowing about a security vulnerability allows companies

to respond. Companies should also continually monitor for, identify and rectify security vulnerabilities within their own products and services." The Department's report then recommends **"All software components in internet-connected devices should be securely updateable.** Updates must be timely and not impact on the functioning of the device."[119]

Germany

In July 2017, a consumer advocacy group filed a lawsuit against a retailer that sold an inexpensive smartphone made by Mobistel. **The phone's software came with 15 critical and known security vulnerabilities which were not disclosed to the consumer at the time of purchase** but were later identified by investigators from the Federal Office for Information Security (BSI).[120]

While still working its way through the judicial system, this lawsuit points to the possibility that companies manufacturing software applications could be held liable for selling defective products to consumers — in exactly the same way that automakers have long been held liable for producing cars that have parts known to be defective.

When software kills people

In March 2018, a car killed pedestrian Elaine Herzberg in Tempe, Arizona. The crash resulted from a bug in the software controlling an Uber self-driving car. The software and sensors did detect Herzberg, but ruled her presence as a "false positive" not requiring the brakes to be applied.[121] The result was tragic.

For some, the expectation for any software is perfection, but the reality is that bug and vulnerability free software will never exist. That said, bugs and vulnerabilities in software can be minimized. **Where social normalizations of deviance in development teams over the past decade have led to persistent and pervasive use of open source components with known vulnerabilities, there is a growing argument for liability.** The deviant behavior is reflective of where reasonable care is not being applied to known vulnerabilities — defects that are relatively easy to identify. To be clear, this same argument does not apply to unknown vulnerabilities — code defects that can be costly to identify. For too long, speed of innovation has been favored over adherence to minimum quality standards.

The dawn of software liability

In 2003, Bruce Schneier wisely wrote, "Liability enforcement is essential. Remember that I said the costs of bad security are not borne by the software vendors that produce the bad security. In economics this is known as an externality: a cost of a decision that is borne by people other than those making the decision. **Today there are no real consequences for having bad security, or having low-quality software of any kind. Even worse, the marketplace often rewards low quality.** More precisely, it rewards additional features and timely release dates, even if they come at the expense of quality."[122] Many would agree that innovation is still heavily favored over quality today.

Paul Rosenzweig, Senior Fellow at the R Street Institute recently wrote, **"I see no prospect in the long run for avoiding liability for insecure code."** Rosenzweig also advises software development organizations to involve themselves in definitions of liability being considered by governments, writing, "Industry owes itself the obligation of trying to get ahead of the curve. If they don't help to design the liability system now, someone will design it for them and I suspect they will like it a lot less than if they had built it themselves."[123]

The first major regulation to hold organizations liable for not building security in by design went into effect in May 2018; it is called GDPR. Where GDPR is applied to organizations operating in the European Union, the U.S. government is also considering new liability standards. In early 2018, U.S. Senator Mark Warner called for increased software liability for organizations developing software. While software makers have long been exempt from liability lawsuits, Warner was stepping up the rhetoric during a speech at SXSW, saying "We need a cybersecurity doctrine, and that raises questions, including on software liability."[124] [125]

Warner, Rosenzweig, and Schneier — among many others — understand that known vulnerabilities baked into software weaken national defenses, healthcare systems, the energy grid, and financial systems. They understand the issue at hand as not one of technology, but one of social and economic well being.

Summary

Decades ago, W. Edwards Deming taught automobile manufacturers the critical importance of building quality into their products by more effectively managing suppliers, sourcing parts, and tracking the precise location of every part assembled in every vehicle. Today, these same lessons are being applied to optimize the performance of modern software supply chains.

As caretakers of the Central Repository, we support millions of software developers from around the world. From this unique vantage point, we strive to do two things everyday; cultivate a deep understanding with respect to the quality of open source components, and study the patterns and practices exhibited by high-performance software development organizations that consume these components to build applications.

The sole purpose of this report has been to share with you the things that we observe, including:

- Open source vulnerabilities increased 120% YoY and mean time to exploit compressed 400%.
- Global supply of open source components across all ecosystems increased 123% YoY.
- Developers continue to gorge on an ever-expanding supply of open source components.
- Open source software components vary widely in terms of quality and security.
- Public vulnerability databases lack information on more than 1.3 million open source security advisories.
- DevOps teams are 90% more likely to comply with open source governance when policies are automated.
- Managing software supply chains through automated OSS governance reduces the presence of vulnerabilities by 50%.
- Government regulations across the United States and Europe hint at software liability on the horizon.

Thank you for reading this year's report. We hope you found it useful. And, we welcome your feedback.

Sincerely,

Team Sonatype

Our Sources

[1] https://www.sonatype.com/2018survey

[2] https://www.washingtontimes.com/news/2018/apr/18/nation-state-sought-to-hack-pentagon-with-same-vul/

[3] https://www.aviationtoday.com/2018/08/28/alaska-hack/

[4] https://securityaffairs.co/wordpress/57130/hacking/cra-apache-struts-hack.html

[5] http://tech.nikkeibp.co.jp/it/atcl/news/17/031600847/

[6] http://exci.to/2mqMAwU

[7] http://www.newindianexpress.com/nation/2018/mar/11/tnie-exclusive-ever-used-india-post-well-your-information-is-vulnerable-to-hacking-1785426.html

[8] https://www.huffingtonpost.in/2018/05/02/hacker-attack-on-epfo-reveals-vulnerability-of-aadhaar-seeding-platform_a_23425089/

[9] https://financefeeds.com/gmo-payment-gateway-confirms-data-leakage-two-client-websites/

[10] https://www.foerderland.de/fileadmin/pdf/IBM_XForce_Report_2016.pdf

[11] https://www.wsj.com/articles/cyber-matters-heed-the-window-of-opportunity-1527115463

[12] https://oversight.house.gov/wp-content/uploads/2017/10/Corman_Testimony_IOT_10032017.pdf

[13] https://cwiki.apache.org/confluence/display/WW/S2-057

[14] https://semmle.com/news/apache-struts-CVE-2018-11776

[15] https://www.scmagazine.com/proof-of-concept-exploit-published-shortly-after-disclosure-of-critical-apache-struts-2-flaw/article/791343/

[16] Forrester: Six Trends That Will Shape DevOps Adoption In 2017 And Beyond, August 2017

[17] https://cloudplatformonline.com/2018-state-of-devops.html

[18] https://www.youtube.com/watch?v=IvcTAOdWnQg

[19] http://fortune.com/global500/list/

[20] http://fortune.com/2018/05/07/security-equifax-vulnerability-download/

[21] https://www.youtube.com/watch?v=IvcTAOdWnQg

[22] https://www.pcworld.com/article/2046695/hackers-targeting-servers-running-apache-struts-applications-researchers-say.html

[23] https://soundcloud.com/owasp-podcast/security-processes-at-the-apache-software-foundation-w-mark-thomas-and-brian-fox

[24] https://www.theregister.co.uk/2018/03/05/rest_vuln/

[25] https://www.nytimes.com/2014/09/26/technology/security-experts-expect-shellshock-software-bug-to-be-significant.html

[26] http://heartbleed.com/

[27] https://www.infoworld.com/article/3003197/security/library-misuse-exposes-leading-java-platforms-to-attack.html

[28] http://www.businessinsider.com/npm-left-pad-controversy-explained-2016-3

[29] https://github.com/ChALkeR/notes/blob/master/Gathering-weak-npm-credentials.md

[30] https://threatpost.com/attackers-use-typo-squatting-to-steal-npm-credentials/127235/

[31] https://en.wikipedia.org/wiki/Typosquatting

[32] https://kromtech.com/blog/security-center/cryptojacking-invades-cloud-how-modern-containerization-trend-is-exploited-by-attackers

[33] https://www.theregister.co.uk/2017/09/15/pretend_python_packages_prey_on_poor_typing/

[34] https://www.bleepingcomputer.com/news/security/ten-malicious-libraries-found-on-pypi-python-package-index/

[35] https://www.bleepingcomputer.com/news/security/ten-malicious-libraries-found-on-pypi-python-package-index/

[36] https://www.bleepingcomputer.com/news/security/somebody-tried-to-hide-a-backdoor-in-a-popular-javascript-npm-package/

[37] http://www.nbu.gov.sk/skcsirt-sa-20170909-pypi/index.html

[38] https://hackernoon.com/im-harvesting-credit-card-numbers-and-passwords-from-your-site-here-s-how-9a8cb347c5b5

[39] https://github.com/conventional-changelog/conventional-changelog/issues/282

[40] https://www.theregister.co.uk/2018/02/10/github_account_name_reuse/

[41] https://www.reddit.com/r/golang/comments/7vv9zz/popular_lib_gobindata_removed_from_github_or_why/

[42] https://blog.npmjs.org/post/173526807575/reported-malicious-module-getcookies

[43] https://www.bleepingcomputer.com/news/security/somebody-tried-to-hide-a-backdoor-in-a-popular-javascript-npm-package/

[44] https://www.bleepingcomputer.com/news/security/backdoored-python-library-caught-stealing-ssh-credentials/

[45] https://medium.com/@vesirin/how-i-gained-commit-access-to-homebrew-in-30-minutes-2ae314df03ab

[46] https://www.servicenow.com/lpayr/ponemon-vulnerability-survey.html/lpayr/ponemon-vulnerability-survey.html

[47] https://www.bleepingcomputer.com/news/security/ransomware-gang-made-over-100-000-by-exploiting-apache-struts-zero-day/

[48] https://www.f5.com/labs/articles/threat-intelligence/zealot-new-apache-struts-campaign-uses-eternal-blue-and-eternalsynergy-to-mine-monero-on-internal-networks

[49] https://www.digitaltrends.com/computing/crypto-mining-malware-monero-jenkins/

[50] https://www.zdnet.com/article/apache-2-vulnerability-exploited-in-linux-cryptojacking-campaign/

[51] 2017 DevSecOps Community Survey, www.sonatype.com/2017survey

[52] https://energycommerce.house.gov/wp-content/uploads/2018/04/040218-Linux-Evaluation-of-OSS-Ecosystem.pdf

[53] Sonatype analysis of over 17,000 software applications, 2016 - 2017.

[54] Sonatype proprietary research.

[55] Sonatype proprietary research.

[56] Sonatype proprietary research.

[57] Sonatype proprietary research.

[58] https://www.innosight.com/insight/creative-destruction/

[59] Sonatype proprietary research.

[60] https://twitter.com/seldo/status/831307104241659905

[61] https://twitter.com/seldo/status/961415623694729216

[62] https://twitter.com/seldo/status/990101697413373952

[63] https://twitter.com/dstufft/status/917703274966536192

[64] https://octoverse.github.com/

[65] http://thehill.com/opinion/cybersecurity/361855-pentagons-move-toward-open-source-software-isnt-going-to-enhance

Our Sources

[66] https://www.youtube.com/watch?v=SGBEq4A8ZHQ

[67] Sonatype proprietary research.

[68] Sonatype proprietary research.

[69] https://snyk.io/stateofossecurity/pdf/The%20State%20of%20Open%20Source.pdf

[70] https://www.zdnet.com/article/github-our-dependency-scan-has-found-four-million-security-bugs-in-public-repos/

[71] https://dl.acm.org/citation.cfm?id=3029832

[72] http://dance.csc.ncsu.edu/papers/codaspy17.pdf

[73] Sonatype proprietary research.

[74] Sonatype proprietary research.

[75] Sonatype proprietary research. Downloads from the Central Repository, H1'2018.

[76] https://www.schneier.com/essays/archives/2003/11/liability_changes_ev.html

[77] https://cybersecurityventures.com/hackerpocalypse-cybercrime-report-2016/

[78] https://www.csoonline.com/article/3122460/technology-business/over-6000-vulnerabilities-went-unassigned-by-mitres-cve-project-in-2015.html

[79] https://pages.riskbasedsecurity.com/2017-ye-breach-quickview-report

[80] https://www.recordedfuture.com/chinese-vulnerability-reporting/

[81] https://www.cyberscoop.com/china-national-vulnerability-database-mss-recorded-future/

[82] https://www.recordedfuture.com/chinese-vulnerability-reporting/

[83] https://www.fordfoundation.org/library/reports-and-studies/roads-and-bridges-the-unseen-labor-behind-our-digital-infrastructure/

[84] https://en.wikiquote.org/wiki/Eric_S._Raymond

[85] https://www.sonatype.com/2018survey

[86] Sonatype proprietary research.

[87] Sonatype proprietary research.

[88] https://www.sonatype.com/2018survey

[89] International Federation of Automatic Control, Ironies of Automation, Lisanne Bainbridge (1983)

[90] https://www.forrester.com/report/Top+Recommendations+For+Your+Security+Program+2018/-/E-RES143074

[91] 2018 DevSecOps Community Survey, Sonatype

[92] 2018 DevSecOps Community Survey, Sonatype

[93] Sonatype proprietary research.

[94] Sonatype research from applications submitted to publicly available open source analysis tool.

[95] https://www.ptsecurity.com/upload/corporate/ww-en/analytics/PT-AI-Statistics-2018-eng.pdf

[96] https://www.insignary.com/android-binary-scans

[97] Sonatype research from applications managed using automated OSS governance

[98] 2018 DevSecOps Community Survey, Sonatype

[99] http://www.eweek.com/security/ibm-s-schneier-it-s-time-to-regulate-iot-to-improve-cyber-security

[100] https://www.congress.gov/bill/115th-congress/house-bill/3010/text

[101] https://www.congress.gov/bill/115th-congress/senate-bill/1691/text?format=txt

[102] https://www.scribd.com/document/355273144/IoT-Cybesecurity-Improvement-Act-Fact-Sheet#-from_embed

[103] https://www.fda.gov/downloads/MedicalDevices/DeviceRegulationandGuidance/GuidanceDocuments/ucm089593.pdf

[104] https://blogs.fda.gov/fdavoice/index.php/2017/10/fdas-role-in-medical-device-cybersecurity/

[105] https://blogs.fda.gov/fdavoice/index.php/2017/10/fdas-role-in-medical-device-cybersecurity/

[106] https://www.fda.gov/downloads/AboutFDA/CentersOffices/OfficeofMedicalProductsandTobacco/CDRH/CDRHReports/UCM604690.pdf

[107] http://www.executivegov.com/2018/08/dod-plans-to-include-product-service-security-as-acquisition-metric/

[108] https://www.mitre.org/sites/default/files/publications/pr-18-2417-deliver-uncompromised-MITRE-study-8AUG2018.pdf

[109] https://www.mitre.org/sites/default/files/publications/pr-18-2417-deliver-uncompromised-MITRE-study-8AUG2018.pdf

[110] https://energycommerce.house.gov/wp-content/uploads/2017/11/20171116HHS.pdf

[111] https://www.ntia.doc.gov/SoftwareTransparency

[112] https://gcn.com/articles/2018/08/24/software-bill-of-materials.aspx

[113] https://www.ferc.gov/industries/electric/indus-act/reliability/01-18-18-E-2-Presentation.pdf

[114] https://www.eenews.net/stories/1060062889

[115] https://www.privacy-regulation.eu/en/article-83-general-conditions-for-imposing-administrative-fines-GDPR.htm

[116] http://www.sgdsn.gouv.fr/uploads/2018/02/20180206-np-revue-cyber-public-v3.3-publication.pdf

[117] https://blog.lukaszolejnik.com/highlights-of-french-cybersecurity-strategy/

[118] https://www.gov.uk/government/news/government-acts-to-protect-essential-services-from-cyber-attack

[119] https://assets.publishing.service.gov.uk/government/uploads/system/uploads/attachment_data/file/686089/Secure_by_Design_Report_.pdf

[120] http://www.sueddeutsche.de/digital/it-sicherheit-unsicheres-smartphone-verbraucherzentrale-klagt-gegen-media-markt-1.3592816!amp

[121] https://www.theinformation.com/articles/uber-finds-deadly-accident-likely-caused-by-software-set-to-ignore-objects-on-road

[122] https://www.schneier.com/essays/archives/2003/11/liability_changes_ev.html

[123] https://chertoffgroup.com/point-of-view/109-the-chertoff-group-point-of-view/677-the-evolving-landscape-of-cybersecurity-liability

[124] https://www.washingtonexaminer.com/policy/technology/mark-warner-eyes-liability-for-software-developers-as-key-way-to-shore-up-cybersecurity

[125] https://schedule.sxsw.com/2018/events/PP99655

Appendix

Acknowledgments

Each year, the State of the Software Supply Chain report is produced to shed light on the patterns and practices associated with open source software development. This year, we began collecting data for our 2018 report from the moment our 2017 report was published. The report is made possible thanks to a tremendous effort put forth by many team members at Sonatype, including: Derek Weeks, Matt Howard, Joel Orlina, Bruce Mayhew, Gazi Mahmud, Dariush Griffin, Mike Hansen, Brian Fox, Ilkka Turunen, Elissa Walters, Daniel Sauble, Melanie Latin, Adam Cazolla, Alvin Gunkel, and Cameron Townshend. Additionally, we could not have produced this report without contributions - big and small - from the DevOps and Open Source Software Community, including: Josh Corman (PTC), Hasan Yasar (CMU), John Willis (SJ Technologies), Gene Kim (IT Revolution), David Blevins (Tomitribe), Robert Hackett (Fortune), Kevin Greene (MITRE), DJ Schleen (Aetna), Matthew Barker (Twistlock), Donald Stufft (Python Software Foundation), Laurie Voss (npm), and Veronique Pomerleau (Half Serious).

A very special thanks goes out to Clara Charbonneau (Half Serious) who created the awesome design for this year's report.

About the Analysis

The authors have taken great care to present statistically significant sample sizes with regard to component versions, downloads, vulnerability counts, and other data surfaced in this year's report. While Sonatype has direct access to primary data for Java, JavaScript, Python, .NET and other component formats, we also reference third-party data sources as documented.