BLACKDUCK
BY SYNOPSYS®

REPORT

2018 Open Source Security and Risk Analysis

Synopsys Center for Open Source Research & Innovation

Table of contents

About this report

The Black Duck by Synopsys Open Source Security and Risk Analysis (OSSRA) report provides an in-depth look at the state of open source security, license compliance, and code-quality risk in commercial software. Each year, the Black Duck On-Demand audit services group conducts open source audits on thousands of applications for its customers—primarily in conjunction with merger and acquisition transactions. This year's analysis was done by the Synopsys Center for Open Source Research & Innovation (COSRI) and examines findings from the anonymized data of over 1,100 commercial codebases audited in 2017. Industries represented in the report include the automotive, big data (predominantly artificial intelligence and business intelligence), cyber security, enterprise software, financial services, healthcare, Internet of Things (IoT), manufacturing, and mobile app markets.

The OSSRA report includes insights and recommendations intended to help organizations and security, risk, legal, development, and M&A teams better understand the open source security and license risk landscape as they strive to improve their application risk management processes.

This year's analysis examines findings from the anonymized data of over 1,100 commercial codebases audited in 2017.

8% of the audited codebases were found to contain Apache Struts, and of those, **33%** still contained the Struts vulnerability.

2017 in review—Application security in the headlines

The need for open source security management became front-page news in 2017 thanks to a major data breach at one of the world's largest credit reporting agencies, Equifax. Equifax maintains a vast amount of sensitive personal and financial information for residents of North America and the United Kingdom. The breach was reported in September 2017 to have compromised the information of over 148 million U.S. consumers, nearly 700,000 U.K. residents, and more than 19,000 Canadian customers.

Leaving the door open to exploit at Equifax

On March 8, 2017, the U.S. Department of Homeland Security sent out a notice of the need to patch a particular vulnerability in certain versions of Apache Struts, an open source framework for creating web applications. Struts is widely used by Fortune 100 companies to build corporate websites in sectors including education, government, financial services, retail, and media.

The need for open source security management became front-page news in 2017 owing to a major data breach at Equifax.

Equifax used Struts in its online disputes portal web application, and for reasons still unclear, a vulnerable version of Apache Struts in that portal was not identified or patched, even though Equifax knew of the vulnerability (CVE-2017-5638).

By mid-May attackers had accessed sensitive information at Equifax by exploiting the Struts vulnerability. The company was not aware of the breach at the time, and the unauthorized access continued into late July before finally being discovered. On Sept. 7, 2017, Equifax publicly announced the breach, noting that it compromised such consumer personal information as names, Social Security numbers, birth dates, addresses, and in some instances, driver's license numbers.

Equifax placed blame for the breach on a combination of human error and technical failure. "Both the human deployment and scanning deployment did not work," a former Equifax executive said in testimony to Congress.[1]

The negative publicity associated with the breach and resulting impact on Equifax and its leadership seemed to have little effect on prompting other organizations to investigate their applications for the Struts vulnerability. Eight percent of the audited codebases were found to contain Apache Struts, and of those, 33% still contained the Struts vulnerability that resulted in the Equifax breach.

As GDPR arrives in 2018, data privacy will require data security

Created by the European Parliament, the General Data Protection Regulation (GDPR) mandates that all companies processing and holding the personal data of European citizens must protect that information—regardless of where it is sent, processed, or stored—and proof of protection must be verified. Personal data can be anything from a name, a photo, or an email address to bank details, posts on social networking websites, medical information, or even a computer IP address.

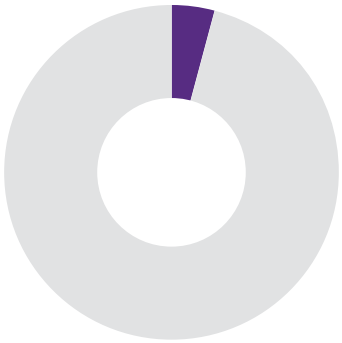
The new regulation applies to any organization that holds or processes the personal information of any European citizen, regardless of where the organization itself is based or where the data processing takes place. All companies that process, hold, or own European data must comply with the law's provisions, including U.S. businesses.

Once this regulation goes into effect, the penalties for noncompliance can be severe. Organizations can be fined up to 4% of annual global revenue or up to €20 million (approximately $22.3 million) for breaching GDPR, whichever figure is higher.

Although on a much smaller scale than the Equifax breach, and virtually unreported by the media, a harbinger of things to come with GDPR was the six-figure fine issued in 2017 to the Gloucester City Council for a breach of U.K. data protection laws.

The council failed to ensure open source software it was using was updated to fix the Heartbleed vulnerability, a critical security flaw that can expose secure communications. Even though Heartbleed was discovered over four years ago, and IT staff at the council flagged the need to update the software, a patch that had been issued for the software was never applied.

An attacker was able to download over 30,000 emails from a senior officer's mailbox; they contained financial and sensitive personal information on past and current employees.



4% of the codebases audited still contained Heartbleed, 4 years after its disclosure.

"Gloucester appears to have overlooked the need to ensure that it had robust measures in place to ensure that the patch was applied," said the entity imposing the £100,000 fine, the U.K. Information Commissioner's Office.[2]

Over four years after its disclosure, a number of organizations across all industries are still vulnerable to exploitation of the Heartbleed bug. Four percent of the codebases audited contained Heartbleed.

All companies processing and holding the personal data of European citizens must protect that information—regardless of where it is sent, processed, or stored.

The £100,000 fine imposed on the Gloucester City Council should serve as a reminder to all organizations of the need to manage the security risks of open source software, which often goes unnoticed and unpatched.

If your organization needs to comply with the General Data Protection Regulation, you'll need to examine the software ecosystem you're using and include open source identification and management in your GDPR security program. In addition to examining custom source code for vulnerabilities, ensure that the open source you or your vendor companies use is not introducing hidden security vulnerabilities.

Interesting developments in GPL enforcement

GPL, or GNU General Public License, is the most commonly used free software license and allows software to be freely used, modified, and redistributed by anyone. In early 2017, the U.S. District Court for the Northern District of California (in the case of *Artifex Software, Inc. v. Hancom, Inc*., which was later settled out of court) found that a breach of the GPL license may be a breach of contract.

In the same matter, the court allowed for a plaintiff who dual-licenses software under both commercial and open source terms (a common business model) to seek monetary relief for breach of the open source license based on the value of the commercial license fees it would have received from the defendant had the defendant's use of the software been licensed under the commercial license.

74% of audited codebases contained components with license conflicts.

While the courts have found that the breach of an open source license can result in IP infringement, they have not definitively ruled whether it is a breach of contract. Judge Corley, in her denial of Hancom's motion to dismiss, set a precedent for treating open source licenses like contracts, and with Artifex and Hancom having reached a confidential settlement, this precedent stands.

The court also addressed an important issue in open source law—whether an open source licensor can obtain an order requiring the licensee to distribute the source code to a derivative work it created. Although the court did not give a ruling on the issue, it expressly refused to dismiss Artifex's request for such an order.

Seventy-four percent of codebases audited for the 2018 OSSRA report contained components with license conflicts, the most common of which were GPL license violations, with 44% of all applications having GPL conflicts. Identifying exactly what open source code is in your codebase is crucial for properly managing its use and reuse, as well as key to ensuring compliance with software licenses, an essential step in reducing business risk.

"Attack of the Connected Car Wash" and other disturbing tales of IoT

As systems become increasingly connected, additional security exposures are created. More connections mean more pathways and back doors that could be exploited by a hacker—especially when a system's own designers are not aware that those pathways and back doors even exist, as is often the case with use of vulnerable open source components.

As products with the ability to connect to the internet become available, hackers have learned how to access data through new—and sometimes unexpected—ways. In July 2017, the FBI warned that conversations with some internet-connected toys could let hackers harvest a child's name, school, likes, dislikes, and location.

In another instance, a North American casino installed a high-tech aquarium as a new attraction, with advanced sensors that automated feeding and reported temperature and salinity to other casino devices. Hackers managed to compromise this high-tech connected fish tank, and then moved on to exploit vulnerabilities in the casino's network, eventually sending over 10 gigabytes of company data to a server in Finland before being discovered.

In the first example of an IoT device being used in a physical attack, two security researchers revealed at Black Hat 2017

that they could hack internet-connected car washes to close entry and exit doors, locking a vehicle and its inhabitants inside the wash chamber while causing mechanical arms to strike the vehicle. If the driver tried to escape, the attackers could repeatedly open and close the wash bay doors as the car attempted to exit, damaging the vehicle and potentially injuring its occupants.

The hack wasn't through a vulnerability. The researchers found an easy back door to access the online system they broke into—the default admin password (would you believe "12345"?). That security lapse is a good reminder for consumers to practice cyber hygiene and change default passwords immediately when setting up a new system, especially one that will be joining the wild kingdom of the Internet of Things. Before taking a device to market, IoT manufacturers should require that its default password be changed at setup, and should put processes in place to scan for information leakage in a device's software to identify information that shouldn't be let outside the organization or its supply chain.

Of the IoT applications scanned, on average **77%** of the codebase was comprised of open source components, with an average **677** vulnerabilities per application.

The Internet of Things encompasses a wide range of devices, from smart refrigerators to insulin pumps. Obviously, there is a big difference between the security issues of low-priority devices like internet-connected toasters and the security of things such as automobiles and medical devices.

Autonomous vehicles are the next-level thing to worry about in hacking cars.

When you put new technology into cars, you run into security challenges. For example:

- When security researchers demonstrated that they could hack a Jeep over the internet to hijack its brakes and transmission, it posed a security risk serious enough that Chrysler recalled 1.4 million vehicles to fix the bug that enabled the attack.
- For nearly half a decade, millions of GM cars and trucks were vulnerable to a remote exploit that was capable of everything from tracking vehicles to engaging their brakes at high speed to disabling the brakes altogether.
- The Tesla Model S's infotainment system contained a four-year-old vulnerability that could potentially let an attacker conduct a fully remote hack to start the car or cut the motor.

Autonomous vehicles will present an especially dangerous challenge. "Autonomous vehicles are the next-level thing to worry about in hacking cars," security researchers Charlie Miller and Chris Valasek said in their keynote at Black Duck's FLIGHT 2017 open source security conference.

"Autonomous vehicles are being specifically designed for outside input," Miller noted. "In 2014 it was an accident our Jeep's CAN-BUS had so much access to the car's functions and that Sprint gave us access to the car's head unit. With self-driving cars, everyone already knows there's a pathway in."

The researchers who demonstrated the car wash hack mentioned earlier in this report had also turned their attention to medical pacemakers in 2017. They acquired hardware and supporting software for four brands of pacemakers and looked for weaknesses in architecture and execution. One of the chief issues noted in the paper they published in 2017 was one the Black Duck On-Demand team sees frequently—unpatched software libraries.

All four pacemakers the researchers examined contained open source components, and roughly 50% of all those components included vulnerabilities. These numbers are in line with the results of the Black Duck On-Demand audits, which found that of the IoT applications scanned, on average 77% of the codebase was comprised of open source components, with an average 677 vulnerabilities per application.

The numbers make it strikingly clear that any organization planning to use IoT technology needs to examine the software ecosystem it uses to deliver a device's features, and account for open source identification and management in its overall security program. Besides examining custom source code for vulnerabilities, companies need to ensure that open source code being used in the Internet of Things does not introduce hidden security vulnerabilities.

Open source and secure voting

A *New York Times* op-ed piece was published in 2017 on the security benefits of moving from proprietary to open sourcing election software. Open source voting applications already play a role in elections in New Hampshire, and districts in San Francisco, Los Angeles, and Texas have allocated funds to move toward open source voting systems as well.

If the open source model for voting systems gains traction, as the *Times* editorial advocated, effective management of open source security will become extremely important. At the 2017 convention DEF CON 25, it took only a few hours for white hat hackers to break into five different voting machines, one via a vulnerability in OpenSSL. Unlike the Gloucester City Council breach, which was enabled by the Heartbleed bug, this exploit took advantage of an unnamed vulnerability in OpenSSL, CVE-2011-4109, first disclosed in 2012.

In another example, a hacker used a 14-year-old exploit in a Microsoft operating system to gain access to an unpatched voting machine. Whether in open source or proprietary code, most known vulnerabilities have patches available on the

If the open source model for voting systems gains traction, effective management of open source security will become extremely important.

date of their disclosure. Both the OpenSSL and Microsoft vulnerabilities had patches available for years that could have prevented the respective DEF CON voting machine attacks. The open source community generally does a good job of discovering and reporting vulnerabilities (over 4,800 of them were reported in 2017 alone), as well as issuing patches. But an alarming number of companies simply do not apply patches, sometimes owing to lack of time, money, and resources or concerns that the patch might break a currently working system.

Yet another voter registration machine was found to use an unencrypted SQLite database (SQLite is one of the world's most popular open source database engines) containing literally all the information that had been captured by the system. A real attacker could have easily obtained the machine's entire voter database, including names, addresses, the last four digits of voters' Social Security numbers, and an electronic facsimile of each voter's signature.

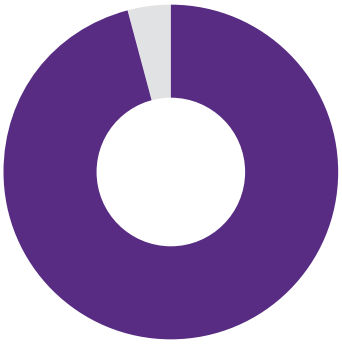
In the first two DEF CON hacking examples, policies should have been in place to ensure that all open source components in use were up-to-date, all patches were applied, and the entity responsible for the system's maintenance was monitoring for new vulnerabilities on a regular basis. In the third DEF CON hacking, a voting registration machine was easily compromised owing to an unencrypted SQLite database. Had open source policies been in place to mitigate risks, they likely would have required that the database be encrypted when first placed on the machine. The lesson from all three hacks is that only with secure systems and policies in place will election officials be able to maximize the benefits of open source while effectively managing its risks.

The 2018 Open Source Security and Risk Analysis report

The concept of keeping software "open" was introduced more than 30 years ago, and the adoption of open source software has been accelerating ever since. Today, open source use is pervasive across every industry and is used by organizations of all sizes. The reasons are straightforward—open source lowers development costs, speeds time to market, and accelerates innovation and developer productivity. In The Forrester Wave™: Software Composition Analysis, Q1 2017, analyst Amy DeMartine notes that "developers use open source components as their foundation, creating applications using only 10% to 20% new code." She goes on to say, "Unfortunately, many of these components come with liabilities in their license agreements, and one out of every 16 open source download requests is for a component with a known vulnerability."[3]

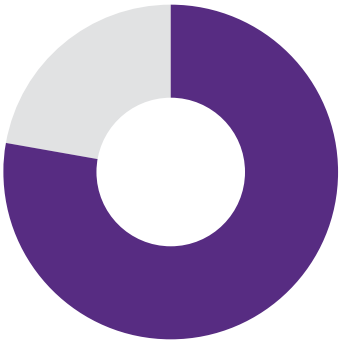
Open source is neither more nor less secure than custom code. However, there are certain characteristics of open source that make vulnerabilities in popular components very attractive to attackers. As results from Black Duck audits show, open source is now ubiquitous in both commercial and internal applications, providing attackers with a target-rich environment when vulnerabilities are disclosed. And vulnerabilities—as well as exploits—are regularly disclosed through sources like the National Vulnerability Database (NVD), mailing lists, and project home pages.

Unlike commercial software, where updates are automatically pushed to users, open source has a pull support model—users are responsible for keeping track of vulnerabilities, fixes, and updates for the open source they use. Open source can enter codebases through a variety of ways, not only through third-party vendors and external development teams but also through in-house developers. If an organization is not aware of all the open source it has in use, it can't defend against common attacks targeting known vulnerabilities in those components, and it exposes itself to license compliance risk.



Black Duck On-Demand audits found open source components in **96%** of the applications scanned, with an average **257** components per application.

The average percentage of codebase that was open source was **57%** vs. **36%** last year. Many applications now contain more open source than proprietary code.



78% of the codebases examined contained at least one vulnerability, with an average **64** vulnerabilities per codebase.

Open source composition of scanned applications

As part of Synopsys' software composition analysis (SCA) offerings, the Black Duck On-Demand audit services group conducts open source audits for organizations looking to assess license compliance and security risks of a particular application or codebase. One of the outputs of these audits is a complete listing (referred to as a bill of materials, or BoM) of the open source components in use. The BoM is cross-referenced with data contained in the Black Duck KnowledgeBase™ to identify potential license compliance and security risks.

On average, the Black Duck On-Demand audits identified 257 open source components per codebase in 2017. Altogether, the number of open source components found per codebase grew by about 75% between the 2017 and 2018 reports.

The audits found open source components in 96% of the applications scanned, a percentage similar to last year's report. Illustrating the ongoing dramatic growth in open source use, the average percentage of open source in the codebases of the applications scanned grew from 36% last year to 57%, suggesting that a large number of applications now contain much more open source than proprietary code.

Some open source components are so important to developers that those components are found in a significant share of applications. This year, Bootstrap, an open source toolkit for developing with HTML, CSS, and JavaScript, was present in 40% of all applications scanned, followed closely by jQuery, with 36% of applications including that open source component.

Notable among the components common across industries was Lodash, a JavaScript library that provides utility functions for programming tasks. Lodash appeared as the most common open source component used in applications employed by such industries as healthcare, IoT, internet, marketing, eCommerce, and telecommunications.

Over **4,800** open source vulnerabilities were reported in 2017.

= 100 vulnerabilities

The number of open source vulnerabilities per codebase grew by **134%**.

85% of the codebases audited for this report had either license conflicts or unknown licenses.

On average, vulnerabilities identified in the audits were disclosed nearly **6** years ago.

Security issues in open source components

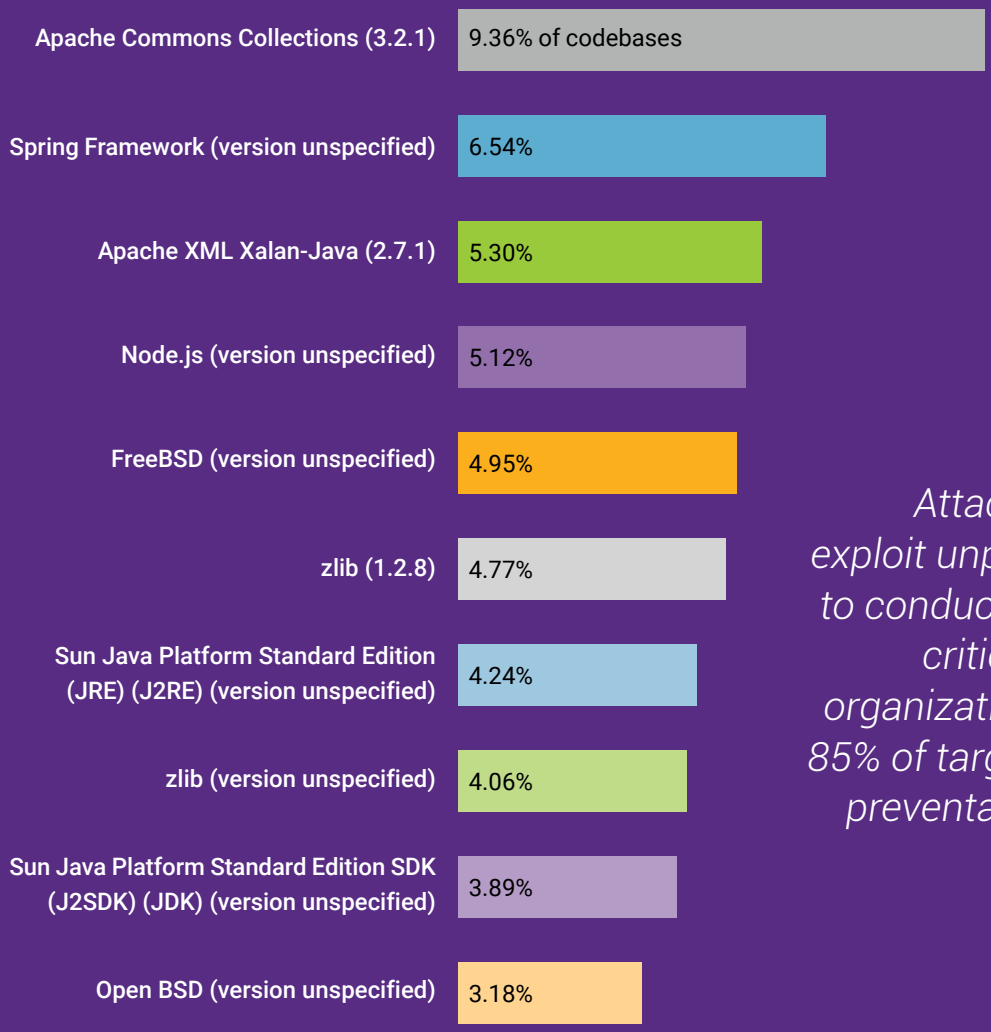
The number of open source vulnerabilities per codebase grew by 134%, with 78% of the codebases examined containing at least one vulnerability, and an average 64 vulnerabilities per codebase.

This high growth rate is partially due to the record number of vulnerabilities reported in 2017. The National Vulnerability Database (NVD) alone listed over 14,700 vulnerabilities versus only 6,400 in 2016. Other reports gave vulnerability totals of over 20,000, with nearly 8,000 of those flying under the NVD radar.

The figures address *all* known vulnerabilities reported in 2017, but more than 4,800 of those were open source vulnerabilities, continuing a five-year growth trend in known open source vulnerabilities. Over 40,000 open source vulnerabilities have been reported in the past 17 years.

Another important datapoint found by the scans was that the average age of the vulnerabilities discovered is increasing. On average, vulnerabilities identified in the audits were disclosed nearly six years ago—versus the four years reported in 2017—suggesting that those responsible for remediation are taking longer to remediate, if they're remediating at all, allowing a growing number of vulnerabilities to accumulate in codebases.

Top 10 high-risk components found



Attackers continue to exploit unpatched software to conduct attacks against critical infrastructure organizations. As many as 85% of targeted attacks are preventable, according to US-CERT.

Vulnerability breakdown

Over 54% of the vulnerabilities found in audited codebases are considered high-risk vulnerabilities, meaning they are more easily exploited. The most common vulnerability found was CVE-2016-9878, a vulnerability in the Pivotal Spring Framework, which appeared in 13% of the codebases audited.

Seventeen percent of the codebases contained a named vulnerability, such as Heartbleed, Logjam, or Poodle, a remarkable figure given that named vulnerabilities generally receive a high level of publicity. The most common named vulnerability discovered—Logjam—was found in 11% of the codebases. Logjam makes supposedly secure TLS-using services vulnerable to being decrypted by an attacker, thus allowing HTTPS sessions to be cracked. Websites, mail servers, and virtual private networks are among the services vulnerable to these attacks. Other notable named vulnerabilities included Poodle, found in 8% of the codebases scanned, Freak and Drown, found in 5% and—discouragingly—Heartbleed, found in 4% of the scanned codebases, even more than four years after its disclosure and several well-publicized exploits.

While they may not receive the same media attention as their "branded" brethren, CVEs are often more pervasive and just as dangerous. Other risky CVEs found in the scanned codebases included CVE-2016-7103, a cross-site scripting vulnerability (in 13% of the codebases); CVE-2014-3625, a path traversal vulnerability (12%); CVE-2014-0050, a permissions vulnerability (12%); and CVE-2015-6420, a remote code execution vulnerability (11%).

Over **54%** of the vulnerabilities found in audited codebases are considered high-risk vulnerabilities.

17% of the codebases contained a highly publicized vulnerability such as Heartbleed, Logjam, Freak, Drown, and Poodle.

Those responsible for remediation are taking longer to remediate, if they remediate at all, allowing a growing number of vulnerabilities to accumulate in codebases.

Top 10 CVEs found

CVEs	Percent of apps containing	Vulnerability type
CVE-2016-9878	13.49%	Path traversal
CVE-2016-7103	12.59%	Cross-site scripting
CVE-2014-3625	12.05%	Path traversal
CVE-2014-0050	11.51%	Permissions, privileges, and access control
CVE-2015-6420	11.33%	Deserialization of untrusted data
CVE-2014-3578	11.33%	Path traversal
CVE-2013-6429	11.15%	Permissions, privileges, and access control
CVE-2016-6303	10.97%	Out-of-bounds write
CVE-2009-1190	10.97%	Resource management errors
CVE-2016-5007	10.97%	Permissions, privileges, and access control

License issues in open source components

While open source security risk gets much of the media attention because of highly publicized exploits such as Heartbleed and the Apache Struts vulnerability, lack of license compliance is the other major risk associated with open source.

An open source code audit is an automated process conducted by human experts to discover the open source components in a codebase using a set of techniques collectively known as software composition analysis, or SCA. The audit identifies all the legal compliance issues related to those open source components, prioritizing any issues based on their severity.

Open source components are governed by one of about 2,500 known open source licenses, many with obligations and varying levels of restriction. Failure to comply with open source licenses can put businesses at significant risk of litigation and compromise of IP.

The audit will also discover known security vulnerabilities related to the open source components, as well as operational risks such as versioning and duplications. An audit report can be invaluable for companies wanting to better understand the composition of their code and software development processes. From an M&A perspective, the code audit enables a buyer to understand risks in the codebase that could affect the value of the IP. Sellers may also employ an audit proactively to avoid surprises in due diligence.

Open source components are governed by one of about 2,500 known open source licenses, many with obligations and varying levels of restriction. Failure to comply with open source licenses can put businesses at significant risk of litigation and compromise of IP. These license requirements can be managed and complied with only if the open source components governed by those licenses are identified. Just as with security vulnerabilities, it is impossible to manage license compliance risk if the components used aren't identified.

Frequency of high security risks by industry

Industry	Percent of open source in codebase	Percent of codebase with high security risks
Aerospace, Aviation, Automotive, Transportation Logistics	53%	30%
Big Data, AI, BI, Machine Learning	45%	25%
Computer Hardware & Semiconductors	74%	22%
Cyber Security	36%	41%
Ed Tech	45%	15%
Energy & Clean Tech	11%	33%
Enterprise Software/SaaS	46%	17%
Financial Services & FinTech	27%	34%
Healthcare, Health Tech, Life Science	48%	31%
Internet & Mobile Apps	57%	60%
Internet of Things	77%	15%
Internet & Software Infrastructure	65%	67%
Manufacturing, Industrials, Robotics	32%	9%
Marketing Tech	76%	23%
Retail & E-commerce	71%	32%
Telecommunications & Wireless	64%	38%
Virtual Reality, Gaming, Entertainment, Media	70%	50%

44% of audited codebases had GPL (GNU General Public License) conflicts.

Even so-called permissive open source licenses typically require acknowledgement of use and other obligations such as redistribution and documentation requirements. And components with no identifiable license terms also can be problematic. When software does not have a license, generally it means no one has permission from the copyright holders of the software to use, modify, or share the software. Creative work (which includes code) is under exclusive copyright by default. Unless a license specifies otherwise, nobody else can legally use, copy, distribute, or modify that work without being at risk of litigation.

The audits conducted by the Black Duck On-Demand team designate a component as "not licensed" when the component is made publicly available but with no clear grant of license or terms of use. Not all teams publishing free software assign a license to their projects, and GitHub, the most popular source of open source on the internet, introduced the ability to attach a license to a project only a few years ago. Stack Overflow and other developer forums where example code can be found are also a common source of "not licensed" software components.

As noted earlier, the audited codebases analyzed for this report contained 257 open source components on average. It's unlikely an organization could keep track of this number of license obligations with the traditional spreadsheet method, and probably impossible without automated processes in place.

Seventy-four percent of audited codebases contained components with license conflicts, the most common of which were GPL license violations, found in 44% of codebases. Eighty-five percent of the codebases audited for this report had either license conflicts or components without licenses.

Frequency of license conflicts by industry

Industry	Percent of open source in codebase	Percent of codebase with license conflicts
Aerospace, Aviation, Automotive, Transportation Logistics	53%	78%
Big Data, AI, BI, Machine Learning	45%	72%
Computer Hardware & Semiconductors	74%	72%
Cyber Security	36%	76%
Ed Tech	45%	77%
Energy & Clean Tech	11%	78%
Enterprise Software/SaaS	46%	83%
Financial Services & FinTech	27%	78%
Healthcare, Health Tech, Life Science	48%	71%
Internet & Mobile Apps	57%	64%
Internet of Things	77%	75%
Internet & Software Infrastructure	65%	78%
Manufacturing, Industrials, Robotics	32%	91%
Marketing Tech	76%	77%
Retail & E-commerce	71%	61%
Telecommunications & Wireless	64%	100%
Virtual Reality, Gaming, Entertainment, Media	70%	92%

Open source security and license risk across verticals

Unsurprisingly, given the prevalence of open source, its use is pervasive across every industry vertical. The Black Duck audit data shows open source components make up between 11% and 77% of commercial applications across a variety of industries.

Vulnerable components were found in applications in every industry. Of clear concern were the code audits of applications from industries the public entrusts with their sensitive personal, financial, and health information. Many of these organizations have apparently not learned the lesson taught by Equifax and may contain data breach time bombs relentlessly waiting to be triggered by an exploit.

The Internet and Software Infrastructure vertical had the highest proportion—67%—of applications containing high-risk open source vulnerabilities, followed by Internet and Mobile Applications, with 60%. Ironically, while lower than last year's 59%, 41% of the applications in the Cyber Security industry were found to have high-risk open source vulnerabilities, putting that vertical at fourth highest.

Thirty-four percent of applications scanned in the Financial Services and FinTech markets contained applications with high-risk vulnerabilities, with the Healthcare, Health Tech, and Life Science vertical closely following with 31%. Manufacturing, Industrials, and Robotics had the lowest percentage—9%—possibly due to the pressure that OEMs (manufacturers) put on suppliers across the software supply chain to deliver vetted, clean code. Conversely, the Manufacturing vertical also had the third highest percentage of license conflicts of verticals, at 91%.

Indeed, based on the Black Duck On-Demand audit data, organizations in all verticals should be concerned with open source licensing and the potential risk of litigation or compromise of their code's intellectual property because of failure to comply with an open source license. The percentage of applications with license conflicts ranged from the Retail and E-commerce industry's low of 61% to the high of the Telecommunications and Wireless industry—where 100% of the code scanned had some form of open source license conflict.

Most common high-risk component by industry

Industry	Most common high-risk component in apps	Percent of codebases containing that component
Aerospace, Aviation, Automotive, Transportation Logistics	zlib	17%
Big Data, AI, BI, Machine Learning	Spring Framework	9%
Computer Hardware & Semiconductors	libxml2, zlib	17%
Cyber Security	Apache Log4j	12%
Ed Tech	Zend Framework	12%
Energy & Clean Tech	Apache Xerces-C++ XML Parser	33%
Enterprise Software/SaaS	Spring Framework	5%
Financial Services & FinTech	Spring Framework	10%
Healthcare, Health Tech, Life Science	libtiff, libxml2	4%
Internet & Mobile Apps	Spring Framework	21%
Internet of Things	OpenSSL, Apache Tomcat	10%
Internet & Software Infrastructure	Node.js	28%
Manufacturing, Industrials, Robotics	Sun Java Platform Standard Edition (JRE) (J2RE), Apache Tomcat	9%
Marketing Tech	Symfony	15%
Retail & E-commerce	Apache Commons Collections	13%
Telecommunications & Wireless	Chromium Source	25%
Virtual Reality, Gaming, Entertainment, Media	zlib	25%

Conclusion and recommendations

Over 80% of all cyber attacks happen at the application level. Consistently, over the past two years, the audits of thousands of codebases conducted by the Black Duck On-Demand audit services group have revealed that 96% of the codebases studied contain open source code.

The debate over whether open source should be used is moot. Today, most application code demonstrably *is* open source. Of the codebases audited that contained open source, an average of 57% of those codebases were open source components, confirming that many applications now contain more open source than proprietary code.

With the growth of open source usage comes risk, primarily due to organizations lacking the proper tools to recognize what—and how much—open source is in their internal and public-facing applications. Nearly 5,000 open source vulnerabilities were discovered in 2017, and nearly 40,000 since the year 2000. Seventy-eight percent of the codebases audited contained at least one vulnerability, with an average 64 vulnerabilities per codebase found.

Most open source components are governed by one of about 2,500 known open source licenses, many with obligations and varying levels of restriction. Failure to comply with open source licenses can put businesses at significant risk of litigation and compromise of IP. The audits found 74% of audited applications contained components with license conflicts.

As the codebase landscape changes, an organization's application security program also needs to evolve to continue to be effective. The bottom line is that no one technique can find every vulnerability. Static analysis (SAST) is essential for detecting security bugs—SQL injection, cross-site scripting, buffer overflows—in proprietary code. Dynamic analysis (including DAST, IAST, and fuzz testing) is needed for detecting vulnerabilities stemming from application behavior and configuration issues in running applications.

But with the growth in open source use, organizations also need to ensure that software composition analysis (SCA) is in their application security toolbelt. With the addition of SCA, organizations can effectively detect vulnerabilities in open source components as they manage whatever license compliance their use of open source may require.

Know your code. By integrating policies, processes, and automated solutions into the software development life cycle to identify, manage, and secure open source, organizations can maximize the benefits of open source while effectively managing its vulnerability and license risks.

ABOUT COSRI

The Synopsys Center for Open Source Research & Innovation (COSRI) leverages comprehensive open source data-gathering expertise and skilled teams to conduct cutting-edge open source security, machine-learning, and data-mining research and promotes the secure use of open source that will enable continuous innovation.

Resources

1. Oversight of the Equifax Data Breach: Answers for Consumers, Digital Commerce and Consumer Protection Subcommittee, Energy and Commerce Committee, U.S. House of Representatives, 2017.
2. Monetary Penalty Notice to Gloucester City Council, U.K. Information Commissioner's Office, June 12, 2017.
3. Amy DeMartine, The Forrester Wave™: Software Composition Analysis, Q1 2017, Forrester, 2017.

The Synopsys difference

Synopsys Software Integrity Group helps organizations build secure, high-quality software, minimizing risks while maximizing speed and productivity. Synopsys, a recognized leader in application security, provides static analysis, software composition analysis, and dynamic analysis solutions that enable teams to quickly find and fix vulnerabilities and defects in proprietary code, open source components, and application behavior. With a combination of industry-leading tools, services, and expertise, only Synopsys helps organizations maximize security and quality in DevSecOps and throughout the software development life cycle.

For more information, go to www.synopsys.com/software.

Synopsys, Inc.
185 Berry Street, Suite 6500
San Francisco, CA 94107 USA

U.S. Sales: 800.873.8193
International Sales: +1 415.321.5237
Email: sig-info@synopsys.com

SYNOPSYS®
Silicon to Software™

05/10/18.2018OSSRA-Rep-UL.