

CONTRAST LABS APPLICATION SECURITY INTELLIGENCE BIMONTHLY REPORT

EXECUTIVE SUMMARY

The Contrast Labs Application Security Intelligence Report for January-February 2020 leverages aggregate data collected from applications monitored and protected by Contrast Security solutions. It provides insights around the vulnerabilities found in—and attacks targeting—the applications we monitor and protect. General findings include:

- A subset of applications has a **large number of vulnerabilities**, while the typical application receives **tens of thousands of attacks** per month.
- The vast majority of attacks **do not hit an existing vulnerability** and are thus unsuccessful, but the volume can create alert fatigue that might cause a successful attack to be missed.
- **Attacks on known vulnerabilities in open-source code** focus on vulnerabilities identified three to 10 years ago, highlighting the need to carefully vet such code before use. One specific vulnerability that was responsible for the biggest data breach of 2017 is repeatedly targeted.
- **Key vulnerabilities for development and security teams to watch** include cross-site scripting (XSS), path traversal, and SQL injection.

These findings accentuate that application security (AppSec) continues to be a struggle for many organizations. To deploy secure applications in a timely manner, the best approach is comprehensive: incorporating security testing and response into **every step of the application life cycle**. Such a strategy mitigates the weaknesses of legacy application security tools that slow development cycles, produce numerous false positives and negatives, and require significant security expertise and staff time.

KEY FINDINGS

31%
6
20,000
99%
94%

of Java apps have a cross-site scripting (XSS) vulnerability

Applications have an average of **6** cross-site scripting (XSS) vulnerabilities

The average app was attacked **20,000** times in January-February 2020

of attacks do not reach a targeted vulnerability and should not sound alarms

of attacks come from a U.S. IP address

Contrast Labs' bimonthly Application Security Intelligence Reports provide an update on the status of AppSec as observed by vulnerabilities pinpointed by telemetry from customer applications. The dataset includes vulnerabilities identified by Contrast Assess and attacks detected by Contrast Protect. Every two months, Contrast Labs analyzes this data to determine which types of vulnerabilities and attacks are most prevalent in protected applications, and identifies actionable insights that can aid developers and security teams as they refine their application security strategy. It is the **only report in the industry** that combines insights about vulnerabilities, library issues, and attacks in a single report.



APPLICATION VULNERABILITY TRENDS

FOR JANUARY-FEBRUARY 2020, CONTRAST LABS IDENTIFIED SEVERAL VULNERABILITY TRENDS FROM ANALYSIS OF ITS AGGREGATE DATA:

TREND: A SUBSET OF APPLICATIONS HAS A LARGE NUMBER OF SERIOUS VULNERABILITIES.

Vulnerabilities are quite common in internally developed enterprise applications. In fact, Contrast Assess identified serious vulnerabilities in **32 percent** of applications it monitored during January-February 2020. Identifying and remediating these vulnerabilities as early in the development cycle as possible saves significant human and financial resources. This, in turn, minimizes the amount of work that must be redone and ensures the fastest time to market for the application. Regardless, if vulnerabilities are addressed before the application goes into production, they cannot be exploited by cyber criminals and do not require developers to go back and fix security issues in future sprints.

The number of vulnerabilities detected varies widely from application to application. One indicator of this trend: An average (mean) of **6 cross-site scripting (XSS) vulnerabilities** were found per application (Figure 1). But when one considers that **only 19 percent** of applications contain this type of vulnerability (Figure 2), it is clear that a subset of applications has a large number of them. Similarly, an average of **3 SQL injection vulnerabilities** exist per application, but such vulnerabilities are only found in **7 percent** of applications. Overall, **10 percent had more than 10 serious vulnerabilities** in January-February 2020.

For applications that have a large number of vulnerabilities, the noise created by alerts can cause significant bottlenecks—especially for companies that rely on static application security testing (SAST) tools that require line-by-line scanning of code. Furthermore, SAST tools and dynamic application security testing (DAST) solutions are notorious for producing a significant number of both false positives and false negatives, potentially resulting in both alert fatigue and missed vulnerabilities. Even when alerts are legitimate, SAST and DAST tools do not rank them according to the risk they pose to a particular application.

When security and development teams must power through thousands of alerts with brute force, this can also significantly delay an application's deployment in production. At the same time, risk to an organization remains high due to false negatives.

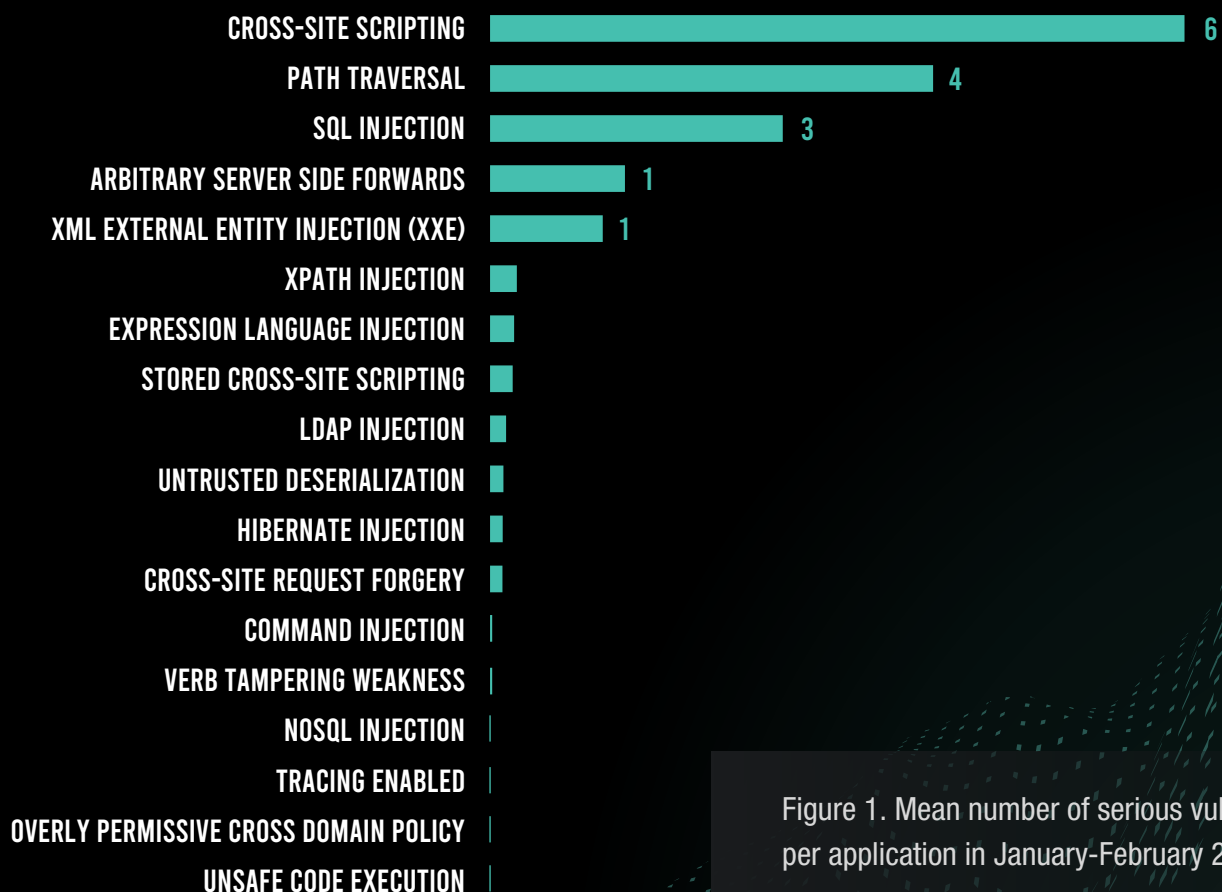


Figure 1. Mean number of serious vulnerabilities per application in January-February 2020 by type.

TREND: THE MOST COMMON TYPE OF SERIOUS VULNERABILITY IS CROSS-SITE SCRIPTING (XSS).

For January and February, XSS is by far the most common serious vulnerability detected by Contrast Assess, occurring in **19 percent** of overall applications and **31 percent** of Java applications (Figure 2). As noted above, applications that have XSS vulnerabilities are likely to have a lot of them.

Other common serious vulnerabilities include path traversal, cross-site request forgery, and SQL injection. For Java applications specifically, XML external entity (XXE) injection vulnerabilities are found in **13 percent** of applications. This makes sense since many Java applications still heavily utilize XML as a way to transport and process data, while other languages do not. Notably, almost every vulnerability type is more common with Java applications than with .NET ones.

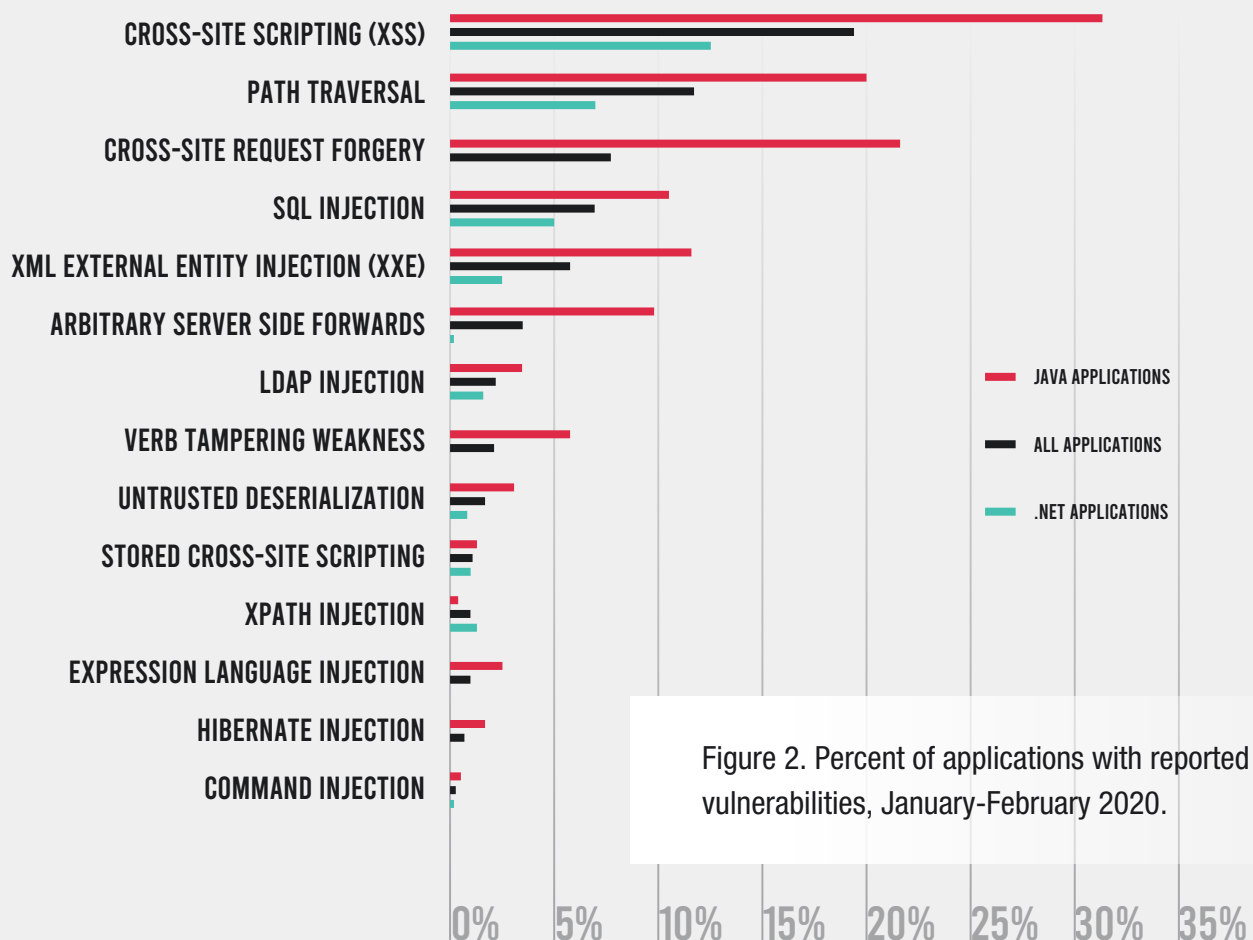


Figure 2. Percent of applications with reported serious vulnerabilities, January-February 2020.

ATTACK TRENDS

FOR JANUARY-FEBRUARY 2020, CONTRAST LABS IDENTIFIED SEVERAL ATTACK TRENDS IN ITS AGGREGATE DATA:

TREND: THE VAST MAJORITY OF ATTACKS ARE PROBES THAT DO NOT REACH A TARGETED VULNERABILITY.

Contrast Labs also analyzed aggregate data from customers that use Contrast Protect to block attacks on applications. Not surprisingly, the examination found that the volume of attacks is quite high, with the average application being affected by **more than 20,000 attacks** in January and February. Overall, large majorities of applications received incoming attacks **targeting path traversal, XSS, and SQL injection vulnerabilities**, and **nearly half** experienced **command injection attacks** (Figure 3).

Fortunately, **99 percent** of these attacks did not reach a targeted vulnerability, and represent adversaries' attempts to generate large volumes of various attack types in the hope that a small percentage of them will succeed. While the large number of unsuccessful attacks might provide some comfort, they also result in **excessive noise** for security and development teams. Traditional application security tools make no distinction between hits and misses, forcing team members to comb through 99 worthless alerts for every one that needs to be addressed.

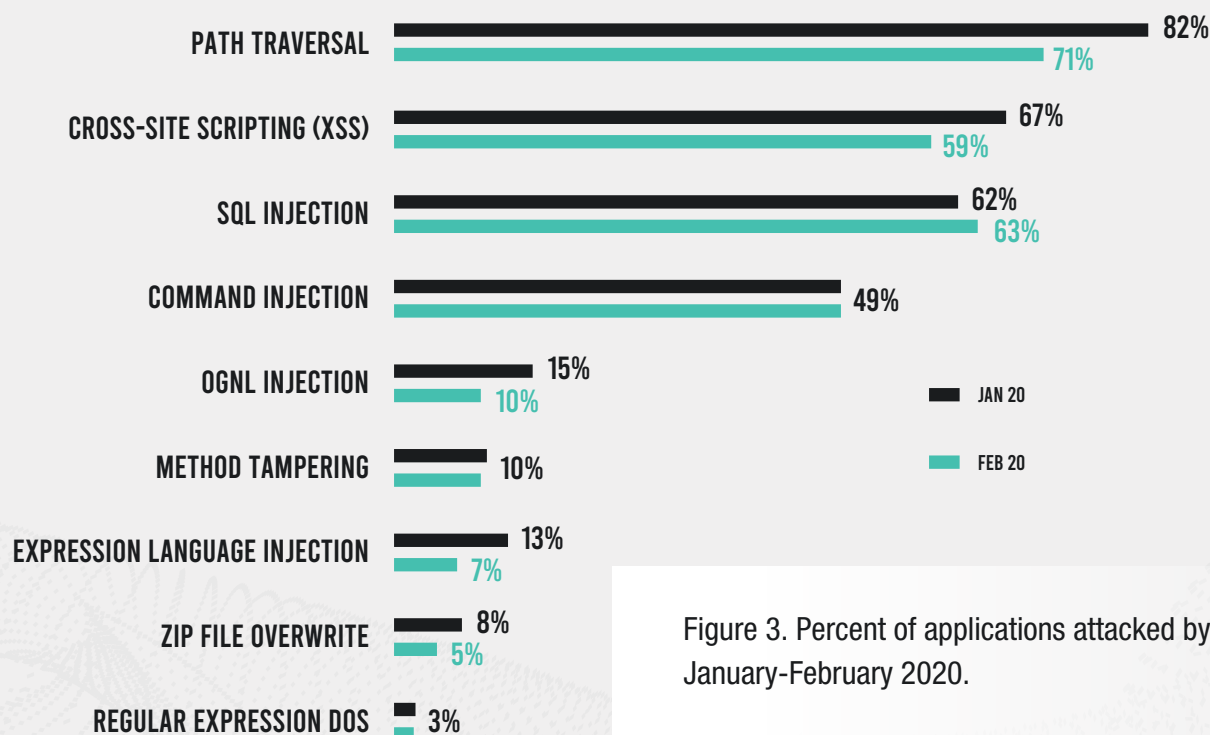


Figure 3. Percent of applications attacked by type, January-February 2020.

TREND: VIRTUALLY ALL ATTACKS WERE INITIATED OR DEFLECTED FROM A U.S. IP ADDRESS.

Many analyses of the threat landscape pinpoint locations such as Russia, China, and certain eastern European countries as the ultimate source of a large percentage of cyberattacks.¹ We have no reason to doubt these conclusions. However, Contrast Protect records the most recent IP address through which an attack originates. Using this metric, Contrast Labs determined that the U.S. is the source of **94 percent** of attacks on protected applications aimed at organizations in that country.

Notwithstanding, we operate on the assumption that the majority of these attacks were launched by bad actors based outside of the U.S. Because of the latency involved with long distance, an attacker can produce a larger volume of reliable attacks from locations closer to their targets. As a result, many now take advantage of U.S.-based server assets (including cloud-based infrastructure from major providers) to launch attacks. Others use a U.S. location as a “**reflecting point**” for an attack, or simply use **masking techniques** to make a foreign attack appear to originate from a U.S. IP address to prevent a red flag from being raised.²

TREND: NON-LIBRARY CODE ATTACKS FOCUSED ON COMMAND INJECTION AND SQL INJECTION.

While the majority of code found in most applications comes from open-source libraries, non-library code has seen more attacks, according to data from Contrast Labs. The vast majority of these attacks in January and February came in the form of **SQL injection** and **command injection** (Figure 4). As noted, SQL injections have a higher potential payout than some others in terms of potential data exfiltrated, so it makes sense that attackers would seek to exploit any such vulnerabilities in fresh code that they encounter.

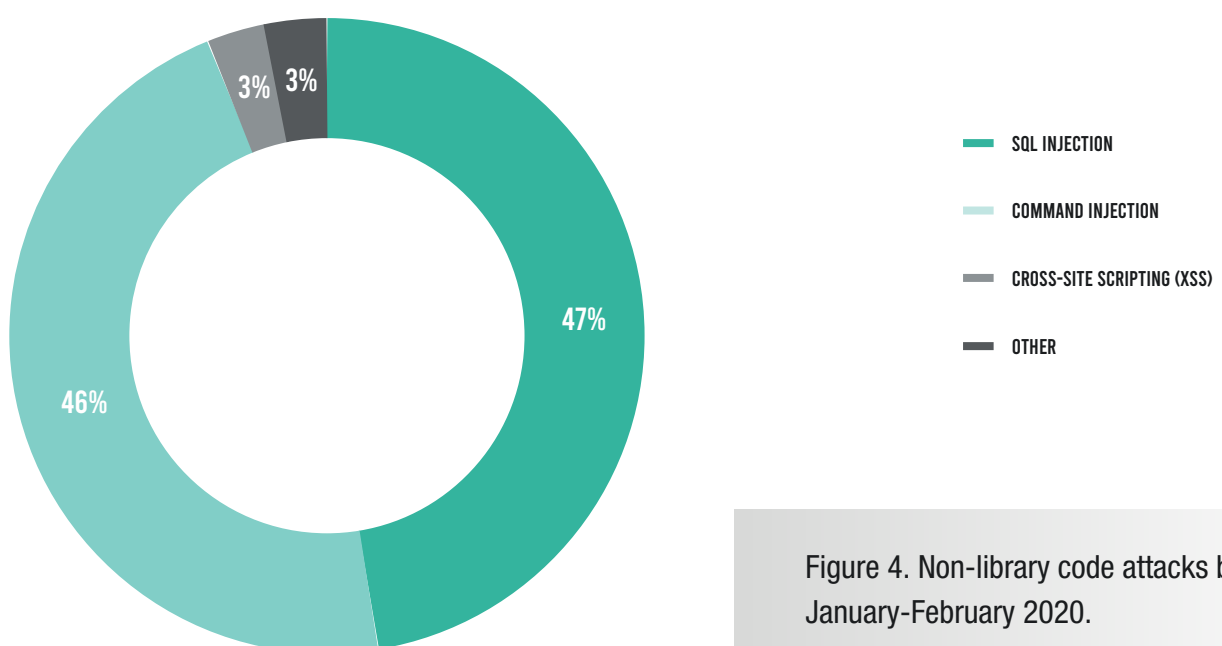


Figure 4. Non-library code attacks by type, January-February 2020.



TREND: ATTACKS ON OPEN-SOURCE CODE FOCUSED ON A HANDFUL OF CVEs

For off-the-shelf code from open-source libraries, the database of Common Vulnerabilities and Exposures (CVEs) catalogs all known vulnerabilities—more than 133,000 of them.³ Fortunately, a large majority of these are never targeted, but they can **create noise** that makes it difficult to find the CVE vulnerabilities that truly pose risk. Contrast Labs analyzed which CVEs are currently being targeted, which can help security and development teams prioritize fixes and understand the methods used by attackers targeting them.

All of the **top four CVEs** attacked in January-February 2020 were vulnerabilities in the **Apache Struts open-source Java web application framework**, and three of the four were in the top four in December 2019 (Figure 5). This is consistent with another study that found that just two major application frameworks, including Apache Struts, account for **55 percent** of vulnerabilities that are weaponized and exploited.⁴

CVE-2017-5638, an improper input validation vulnerability in Apache Struts that was famously blamed for the biggest data breach of 2017,⁵ has been among the top four since at least June 2019.

CVE-2017-9791, an Apache Struts remote code execution vulnerability, has cycled in and out of the top four over the past nine months. *CVE-2014-0112* and *CVE-2014-0114* saw residual activity in January after spiking in December.

It is noteworthy that the top CVEs targeted are **all quite old**. Among the top eight, **the newest were discovered in 2017** and the **oldest go back to 2010**. When adversaries attack specific vulnerabilities repeatedly, they do so because they see potential profit in exploiting them. While another 2017-level breach would represent a huge jackpot for a cyber criminal, even smaller breaches can bring significant profit.

Another interesting insight: **Two of the top eight**, *CVE-2014-0112* and *CVE-2014-0114*, were created in an attempt to remediate another vulnerability, *CVE-2014-0094*. It is a reminder that code from open-source libraries must be vetted carefully.

Additionally, it is important to note that the top four CVEs targeted belong to a broader category known as **expression language (EL) injection vulnerabilities**. And even though EL injection vulnerabilities are the **ninth most common vulnerability** in custom code, they rank higher on the list of vulnerabilities attacked (see Figure 7). Further, the risk of EL injection vulnerabilities is high due to the fact that attackers can modify or invoke functionality on the application server and gain access to data and functionality—as well as hijack accounts or enact remote control execution.⁶

To avoid EL injection vulnerabilities, developers need to avoid incorporating user-controllable data into dynamically evaluated code. Instead, they should use safer alternative methods for implementing application functions that cannot be manipulated for malicious purposes.

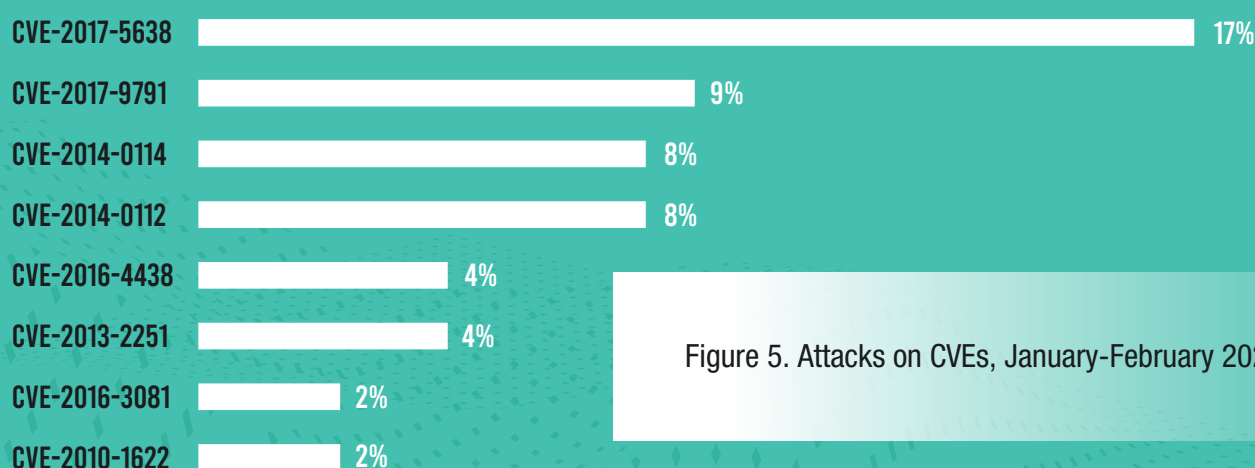


Figure 5. Attacks on CVEs, January-February 2020.



APPLICATION SECURITY WATCH LIST

BRINGING TOGETHER THE VULNERABILITY AND ATTACK DATA FROM CONTRAST LABS, WE PINPOINTED THE TOP VULNERABILITIES TO WHICH DEVELOPMENT AND SECURITY TEAMS SHOULD PAY ATTENTION.

For development and security teams, the most important takeaway when looking at vulnerability and attack data is how the two work together to impact risk. Contrast Labs did further analysis on these two data sets to compile the Application Security Watch list for 2020 (Figure 6). It is based on a comparison of the likelihood that a vulnerability will occur and the likelihood that the specific vulnerability will be attacked (see Figure 7). The top three vulnerabilities on the Watch List are as follows:

- *XSS vulnerabilities* are difficult to avoid, as there are many vectors through which such an attack can be launched. It is no wonder that web applications are impacted in **19 percent** of data breaches.⁷ XSS attacks are also difficult to detect by the owners of a website, as anomalies are typically only visible through the experience of a website user. While XSS may not be the direct cause of a data breach, it can be one of several vulnerabilities in an attack chain.
- *Path traversal*, considered by many to be less dangerous than SQL injection or XSS, can also pose a major security threat in some cases. The level of risk depends on what data is stored in files outside a website's root directory but accessible via this hacking technique. If cyber criminals

can obtain user login credentials for back-end systems through path traversal, they can potentially move laterally within the network for months without detection.

- *SQL injection* attacks have been around for two decades, but they are still considered by cyber criminals to be the most profitable attack type. SQL databases contain all kinds of sensitive data, and applications with vulnerabilities can be queried for information like credit card data and login credentials.

VULNERABILITY RISK FACTOR

CROSS-SITE SCRIPTING (XSS)	1	LDAP INJECTION	11
PATH TRAVERSAL	2	UNTRUSTED DESERIALIZATION	12
SQL INJECTION	3	STORED CROSS-SITE SCRIPTING	13
CROSS-SITE REQUEST FORGERY	4	XPATH INJECTION	14
XML EXTERNAL ENTITY INJECTION (XXE)	5	HIBERNATE INJECTION	15
ARBITRARY SERVER SIDE FORWARDS	6	NOSQL INJECTION	16
HEADER INJECTION	7	TRACING ENABLED	17
COMMAND INJECTION	8	OVERLY PERMISSIVE CROSS DOMAIN POLICY	18
EXPRESSION LANGUAGE INJECTION	9	UNSAFE CODE EXECUTION	19
VERB TAMPERING	10		

Ranking list of vulnerabilities based on prevalence of vulnerabilities and likelihood of an attack.

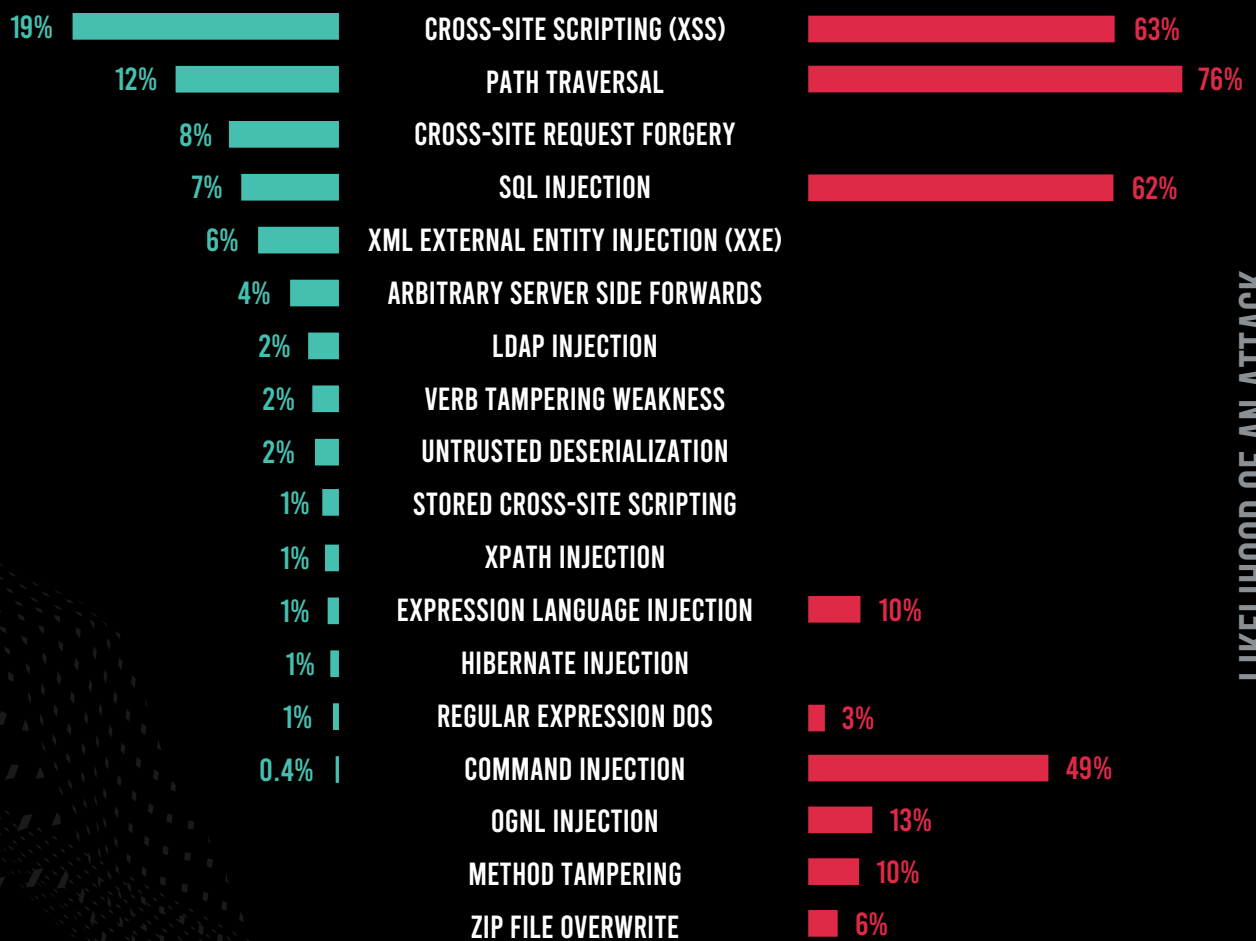
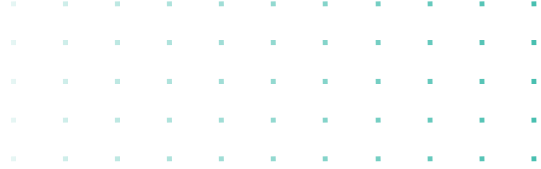


Figure 7. Likelihood of a vulnerability vs. likelihood of an attack, January-February 2020.

CONCLUSION

The Contrast Labs Application Security Intelligence Report for January-February 2020 finds that application security continues to be a critical issue for enterprises, with some applications containing multiple vulnerabilities, and the typical application being hit with tens of thousands of attacks over a two-month period.



Given this volume, it is important for development and security teams to find ways to prioritize vulnerabilities and attacks according to the risk they pose to the organization. Based on data from January and February 2020, areas of special focus should include:

- Vulnerabilities targeting Java applications, especially those developed with Apache Struts
- XSS vulnerabilities, the most common type
- Exploitable vulnerabilities like expression language injection and zip file overwrite
- Command injection vulnerabilities, uncommon but commonly targeted

To mitigate risks such as these, organizations need to “shift left” with their security processes.⁸ In other words, organizations must incorporate security testing into every link of the DevOps toolchain, from initial planning to deployment. Companies must also “shift right” and monitor applications that are running in production.⁹ Such an approach incorporates security testing and protection into the underlying structure of every application, freeing up developers to do their jobs and security professionals to focus on strategic risk management.

¹ E.g., “China, Russia Biggest Cyber Offenders,” U.S. News & World Report, February 1, 2019.

² “How Do Hackers Hide Their IP Address?” Cyber Security Intelligence, May 18, 2018.

³ Common Vulnerabilities and Exposures, MITRE, accessed March 23, 2020.

⁴ Catalin Cimpanu, “WordPress and Apache Struts Account for 55% of All Weaponized Vulnerabilities,” ZDNet, March 17, 2020.

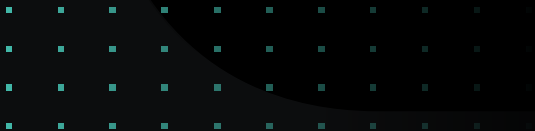
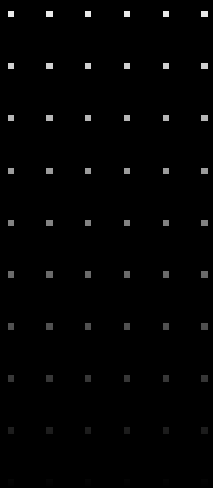
⁵ Thomas Brewster, “How Hackers Broke Equifax: Exploiting a Patchable Vulnerability,” Forbes, September 14, 2017.

⁶ “Expression Language Injection Description,” OWASP, accessed March 30, 2020.

⁷ “2019 Data Breach Investigations Report,” Verizon, accessed March 23, 2020.

⁸ Jakob Pennington, “Shifting Left: DevSecOps as an Approach to Building Secure Applications,” Medium, July 18, 2019.

⁹ Alan Shimel, “DevOps Chat: Shifting Security Left and Right, with Contrast Security,” Security Boulevard, October 7, 2019.



CONTRAST SECURITY

240 3rd Street
Los Altos, CA 94022
888.371.1333

Contrast Security is the world's leading provider of security technology that enables software applications to protect themselves against cyberattacks, heralding the new era of self-protecting software. Contrast's patented deep security instrumentation is the breakthrough technology that enables highly accurate assessment and always-on protection of an entire application portfolio, without disruptive scanning or expensive security experts. Only Contrast has sensors that work actively inside applications to uncover vulnerabilities, prevent data breaches, and secure the entire enterprise from development, to operations, to production.

