

# Web Application Vulnerabilities: Statistics for 2018

2019

## Contents

Introduction.....	2
Executive summary.....	2
Trends.....	3
Assessment of web application security.....	4
Most common vulnerabilities .....	4
Analysis of threats and security levels.....	6
Vulnerabilities in test and production applications.....	8
Comparison of test techniques used.....	8
Conclusions.....	10
Client snapshot .....	11
Materials and methods .....	12

## Introduction

Small businesses, banks, and major industrial concerns all depend on web applications. When seeking a product or service, we tend to do so by visiting the company's website. Thanks to the development of web technologies, government services have improved in both availability and quality. Websites are the public face of both business and government, so any issues with a website can damage reputation. For this and other reasons, web application owners are motivated to improve and support their sites. Proper attention to cybersecurity is a key part of any such strategy.

To meet a high standard of security, web applications must be regularly tested for vulnerabilities. This report provides statistics gathered by Positive Technologies while performing web application security assessments throughout 2018. Data from previous years is provided for comparison purposes.

## Executive summary

In 19 percent of tested web applications, vulnerabilities allow an attacker to take control of the application and server OS. If such a server is on the network perimeter, the attacker can penetrate the internal corporate network. As shown in our [report on vulnerabilities in corporate information systems](#), 75 percent of LAN penetration vectors involve weaknesses in web application protection.

In most cases, web application vulnerabilities are caused by coding errors. Configuration changes suffice to fix only 17 percent of vulnerabilities, most of which are of low severity. Generally speaking, remediating critical vulnerabilities requires making modifications to code.

Around half of leaks may lead to disclosure of account credentials, including for third-party resources. One example is configuration files (with passwords stored inside) that are accessible to all users.

An attacker can obtain personal data from 18 percent of web applications handling such data. Almost all tested web applications (91%) store and process personal data.

On average, each web application contained 33 vulnerabilities, of which 6 were of high severity. The number of critical vulnerabilities per web application grew by 3 times compared to 2017.

Production applications contain fewer vulnerabilities than test applications, but this does not necessarily make them more secure. The percentage of production applications containing at least one high-severity vulnerability was higher than the equivalent percentage for test applications. And as practice shows, a successful attack on a web application often requires only a single high-severity vulnerability.

Analysis of source code makes assessment more effective. When testers have access to source code, the average number of high-severity vulnerabilities found more than doubles, according to our statistics.

## Trends

### Web applications are tending to contain more critical vulnerabilities.

A two-year dip in the percentage of web applications with high-severity vulnerabilities has stopped, rising to 67 percent in 2018. The most common vulnerabilities include Insufficient Authorization, Arbitrary File Upload, Path Traversal, and SQL Injection.

### Unauthorized access continues to be a menace.

The percentage of web applications with vulnerabilities enabling unauthorized access (72%) increased compared to 2017, nearly returning to the levels of 2016 (75%).

### Fingerprinting opportunities decreasing for attackers.

In 2018, only 42 percent of web applications disclosed the versions of software in use. In 2017, 61 percent of web applications did so. Wide coverage of the topic, as well as the relative ease of fixing such vulnerabilities, were likely contributing factors.

### More applications are vulnerable to information exposure.

Access to configuration and debug information, source code, session identifiers, and other sensitive information is possible in 79 percent of web applications. This is concerning when compared to past years such as 2016 (60%) and 2017 (70%).

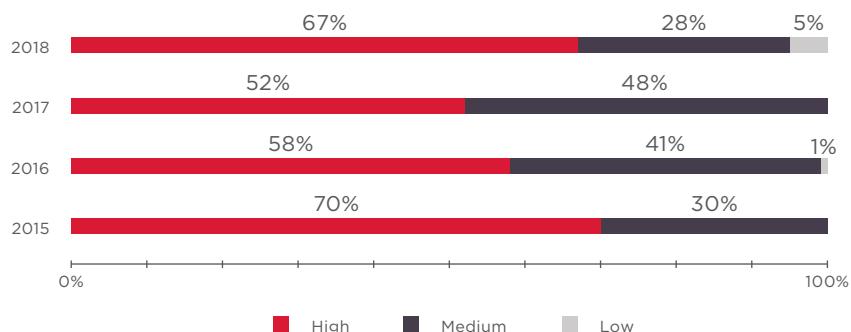


Figure 1. Websites by maximum severity of vulnerabilities found

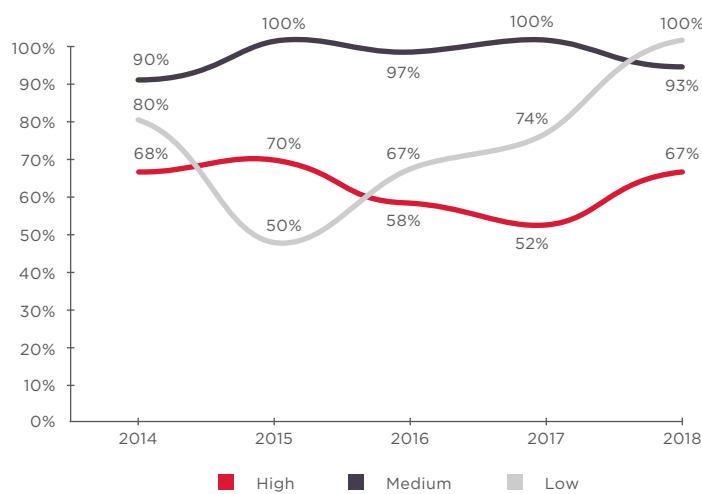


Figure 2. Websites by vulnerability severity

## Assessment of web application security



Code vulnerabilities

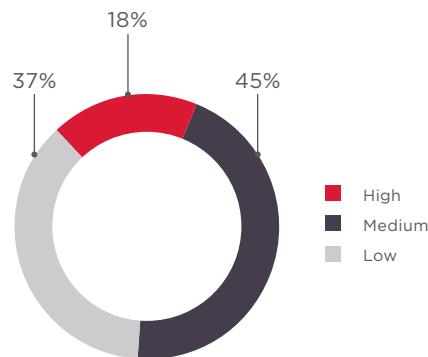


Figure 3. Vulnerabilities by severity

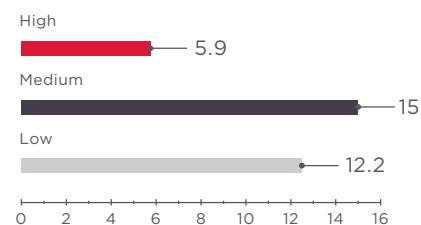


Figure 4. Average number of vulnerabilities per application



Properly configure web application components, web servers, and database servers. Do not leave factory settings and change all default passwords



Disable external entities and DTDs in XML parsers if not needed for web application functionality

## Most common vulnerabilities

In 2018, our testers found around 70 types of weaknesses in web applications. As always, Cross-Site Scripting (XSS) vulnerabilities are present in many web applications. Four out of five web applications contained configuration errors such as default settings, standard passwords, error reporting, full path disclosure, and other information leaks that might have value for potential intruders.

In 2018, there was a fall in the percentage of web applications vulnerable to XML External Entities (XXE). Unfortunately, this is more likely to be a statistical fluctuation than a lasting trend. In the context of web vulnerabilities overall, XXE is as relevant as ever. XXE entered the OWASP Top 10 in 2017, immediately taking fourth place.

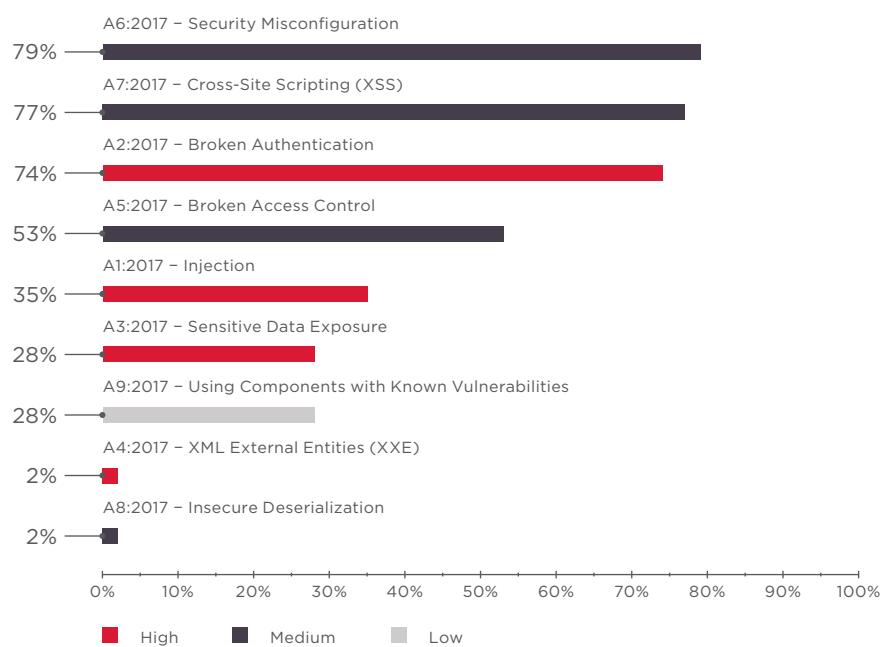


Figure 5. OWASP Top 10–2017 vulnerabilities (percentage of web applications)

In April 2018, reports indicated that the site of American eatery chain Panera Bread stored personal data and card information for 37 million clients in a publicly accessible way



Do not store sensitive data in cleartext. Use strong encryption methods. Set an effective policy for restricting access to sensitive data

Data leaks afflicted systems around the world in 2018. Many infamous incidents have stemmed from failures in administration and access control. As found in our testing, secure storage of sensitive data in web applications is often problematic. Credentials are at stake in 46 percent of leaks. Personal data is processed by 91 percent of the web applications we tested, with 18 percent of them vulnerable to a leak (19% of all leaks).

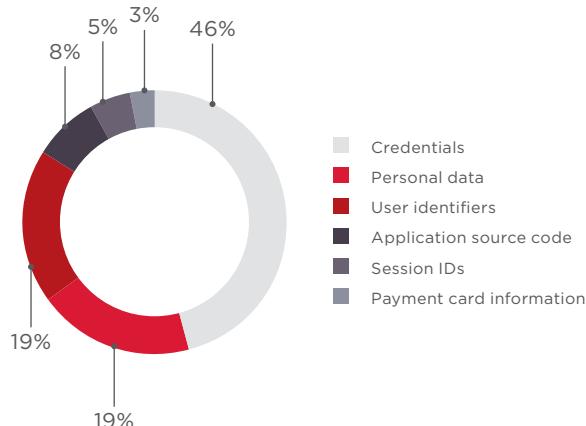


Figure 6. Sensitive data exposure

An attacker was able to steal information regarding 21 million users of Timehop after obtaining administrator credentials. Absence of two-factor authentication was a critical factor in the success of the attacker's subsequent actions



Use multifactor authentication to prevent credential stuffing

Broken authentication can lead to unauthorized access to web application functionality or content.

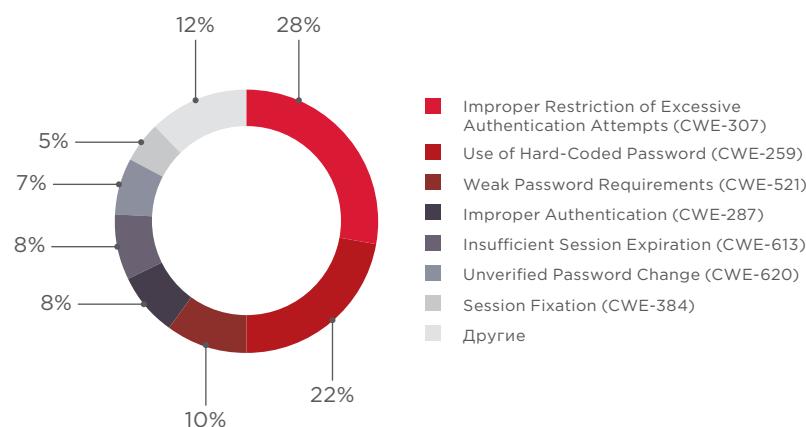


Figure 7. Broken authentication vulnerabilities

In 2018, media outlets reported on fines totaling over \$149 million against Uber, which failed to stop theft of personal data regarding 50 million passengers and 7 million drivers. According to the company's official statement, an attacker discovered credentials for database access in source code stored in a company repository on GitHub

Automated analysis of web applications often revealed hard-coded passwords, such as for access to databases or third-party APIs. An attacker with access to source code could use these credentials for unauthorized access to such systems and their data. In addition, a number of the hard-coded passwords found by our experts did not even meet minimal strength requirements. Therefore, the passwords could fall victim to a brute-force attack. And because they are hard-coded, changing the compromised passwords requires making code modifications.



Do not hard-code passwords. Choose passwords of sufficient length and complexity



Figure 8. Example of a hard-coded password discovered in automated testing of a web application

OWASP [recommends checking for certain vulnerabilities](#) even though they are not in the OWASP Top 10 list for 2017. These include Arbitrary File Upload, a critical vulnerability that allows an attacker to upload executable files and run code on a server, potentially leading to full control over the web application and server.

## One out of four

tested web applications  
allows upload of arbitrary files



Filter uploaded files by extension based on a whitelist. Set a policy for restricting access to directories that contain executable files

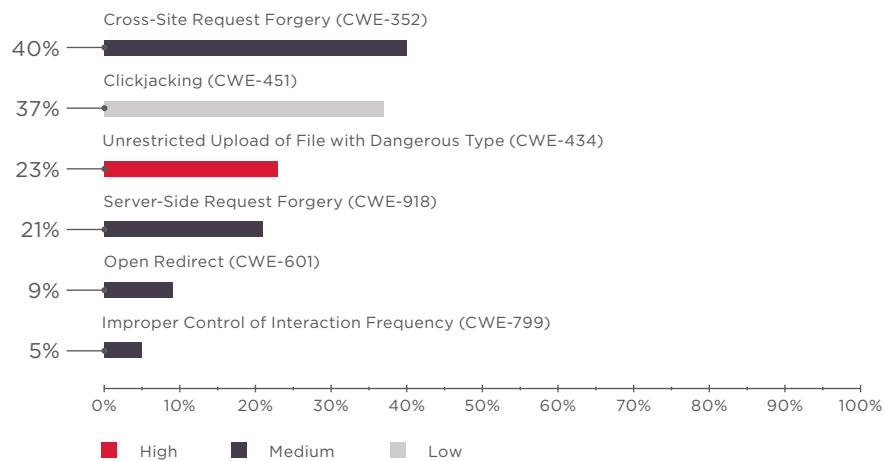


Figure 9. Common vulnerabilities not in the OWASP Top 10-2017 (percentage of applications)

# One out of three web applications

was graded as having an extremely poor level of security.

This represents an increase  
of 15 percent over 2017

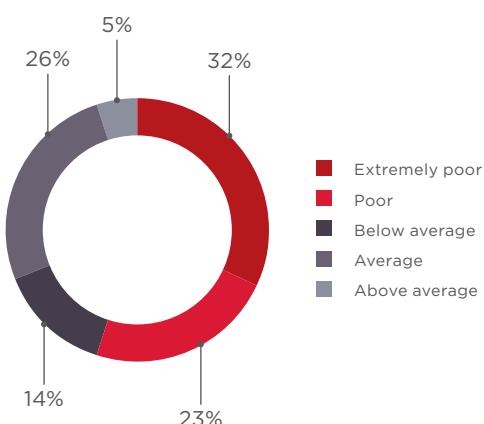


Figure 10. Security level (percentage of web applications)

## Besides data theft,

unauthorized access to a web application may also damage the reputation of site owners

In 2017, nearly half (48%) of web applications were at risk of unauthorized access. But this percentage jumped by half in 2018 (to 72%). In 19 percent of web applications, critical vulnerabilities were present that could allow taking control of both the application and the server OS. If the server is on the network perimeter, it could be compromised as a springboard to attack internal resources. But attacking the LAN is possible even without full control of a web application server. For instance, Server-Side Request Forgery (SSRF) enables scanning the LAN and accessing internal resources.

The attacker responsible for [hacking Ticketfly](#) did not stop with stealing information for 27 million accounts, leaving a [message on the main page](#) so that any visitor would immediately know about the site's security breach

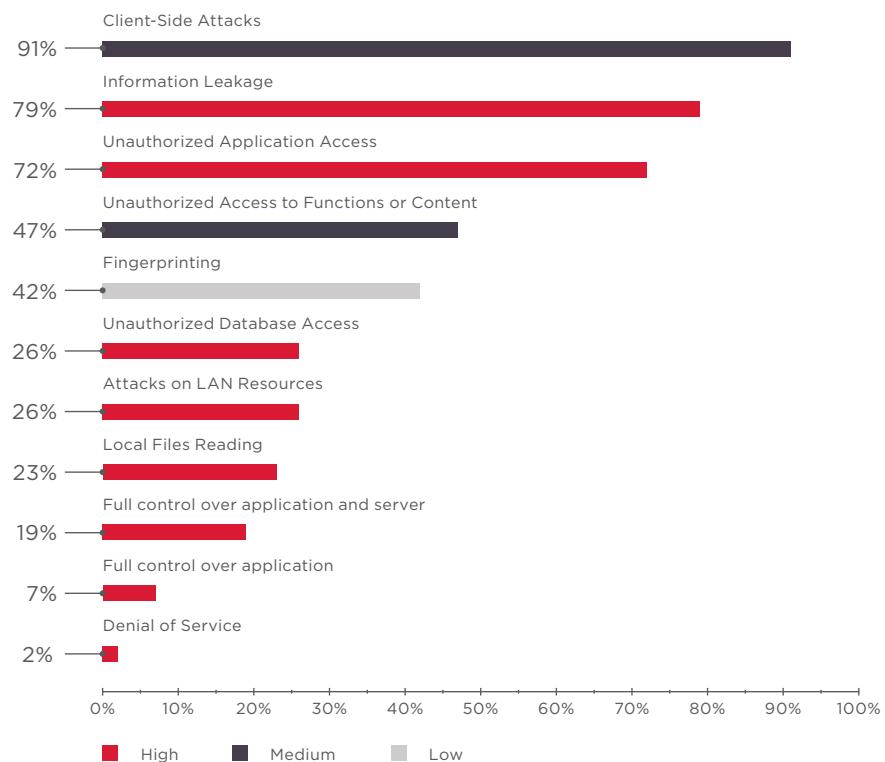


Figure 11. Most common threats (percentage of tested web applications)

Card data for 380,000 visitors of the [British Airways web application](#) was stolen due to injection of malicious JavaScript code. In the aftermath of the incident, [airline shares fell by 3.8 percent](#), while the company itself may be on the receiving end of a [fine of up to GBP 500 million](#)

As in past years, nearly every web application contains vulnerabilities that allow attacks against users. And also as in past years, Cross-Site Scripting (XSS) is at fault in most cases. But this year, the percentage climbed even higher still (from 77.9% in 2017 to 88.5% in 2018). Just one such vulnerability can cause serious consequences, as confirmed by headline-grabbing data breaches.



Filter user inputs, preferably using standard functions of the relevant programming language.

Follow [Content Security Policy](#) principles and use [Subresource Integrity \(SRI\)](#) for protection from malicious JavaScript code

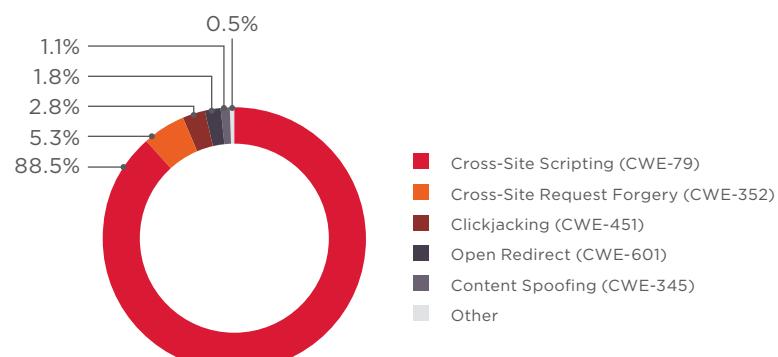


Figure 12. Vulnerabilities allowing attacks against users

## Vulnerabilities in test and production applications

We observed a nearly three-fold increase in the percentage of production applications containing high-severity vulnerabilities (25% in 2017 compared to 71% in 2018). In addition, the average number of vulnerabilities per web application grew for both test and production applications.

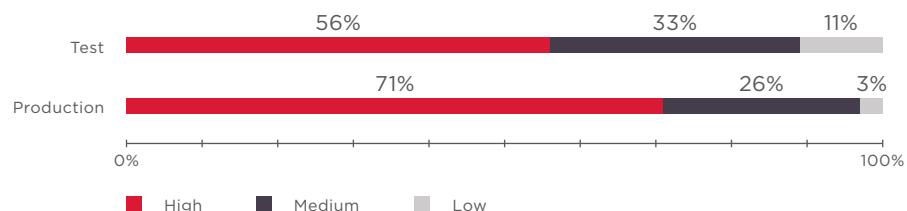


Figure 13. Applications compared by maximum vulnerability severity

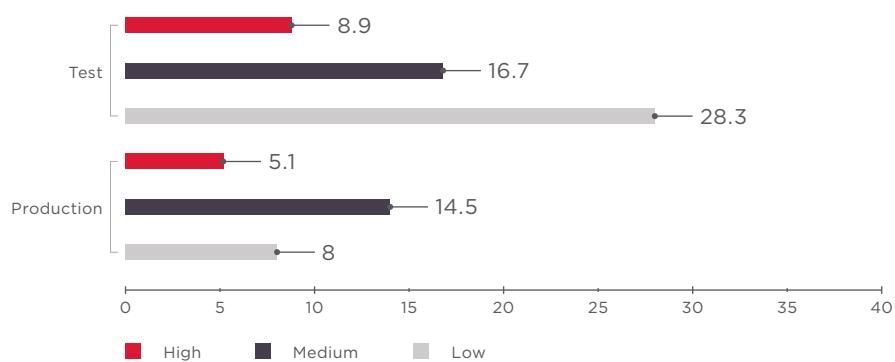


Figure 14. Average number of vulnerabilities per application

## Comparison of test techniques used

Every year, testing results reinforce our view that white-box analysis is the most effective method for finding web application issues. When testers have access to source code, the number of critical vulnerabilities found per application is much higher than with gray-box or black-box methods. In particular, the number of A1 – Injection vulnerabilities found is three times higher with white-box testing than without.

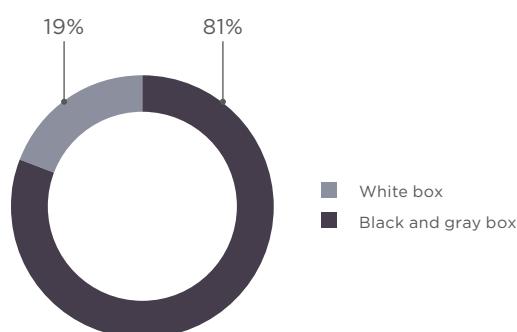


Figure 15. Testing methods (percentage of applications)

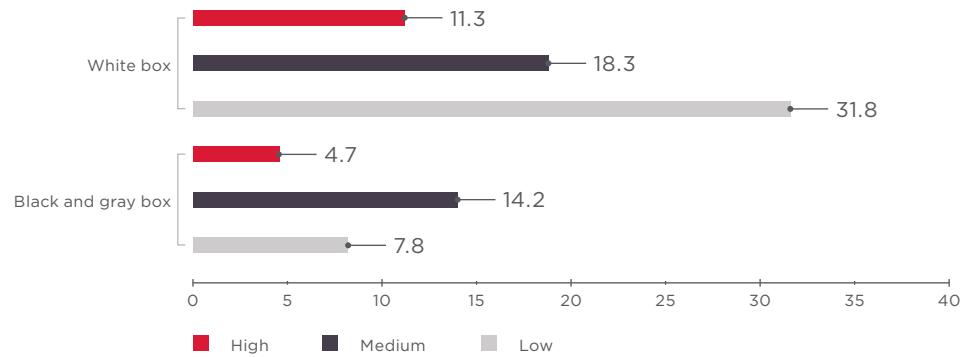


Figure 16. Average number of detected vulnerabilities per web application

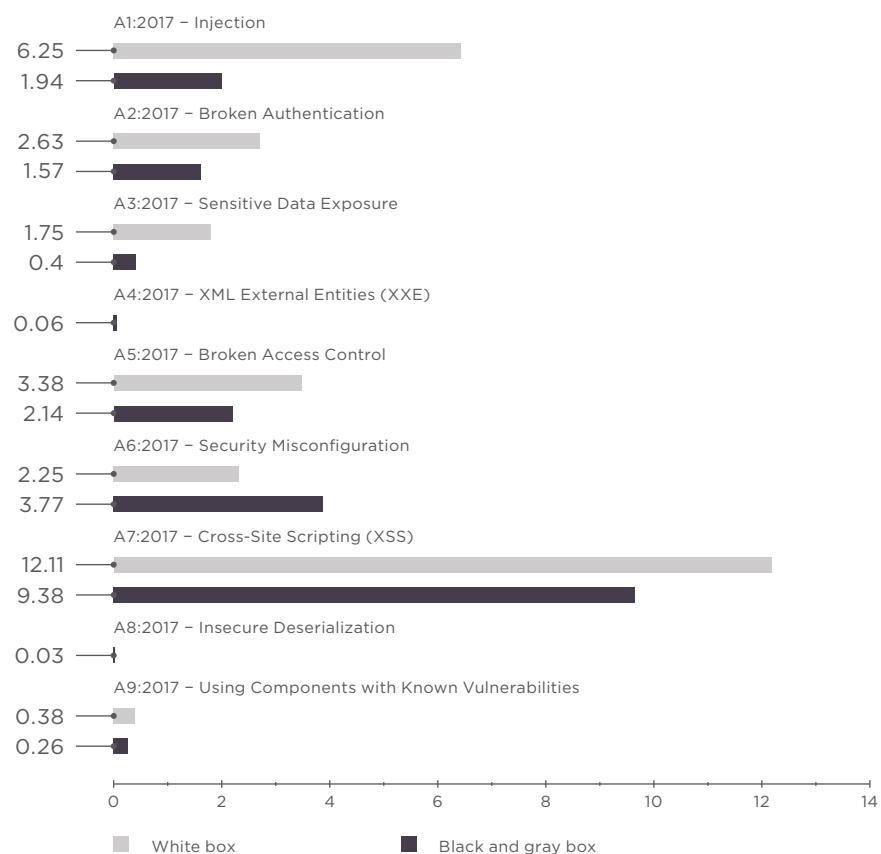


Figure 17. Number of OWASP Top 10-2017 vulnerabilities per web application

## Conclusions

Based on testing results, we conclude that most web applications are poorly protected. The percentage of web applications whose security was "extremely poor" nearly doubled compared to the year prior. The average number of vulnerabilities per application in certain vulnerability categories jumped dramatically.

By signing up on a site, users are forced to trust their data to the website owner. But 18 percent of web applications that handle personal data are at risk of losing control of it.

For effective protection of web applications, we recommend analyzing their state of security. White-box testing (with access to source code) makes this analysis much more effective, enabling clients to identify and fix vulnerabilities before cyberattackers strike. It should also be emphasized that such analysis must be recurring. Only a systematic approach can minimize the number of vulnerabilities and enable proper allocation of resources.

Our results indicate that high-severity vulnerabilities can be found in both test and production applications. Starting analysis at the earliest stages of web application development reduces fix-related costs and produces better results.

Fixes for 83 percent of vulnerabilities, including the majority of critical vulnerabilities, require that the web application developer make changes to code. Even modest rewriting of a web application can be incredibly expensive. To reduce the risk of disruption to business processes during remediation as new code is being prepared, we advise companies to consider solutions such as a web application firewall (WAF). Only by employing a comprehensive approach to web application security can companies minimize the risk of crippling cyberattacks, save money, and win client trust.

## Client snapshot

In 2018, testing was performed of

**43 web applications**



Percentage of production  
applications

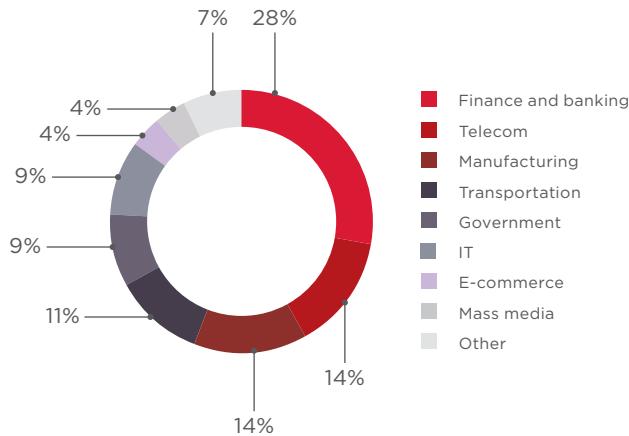


Figure 18. Participant portrait

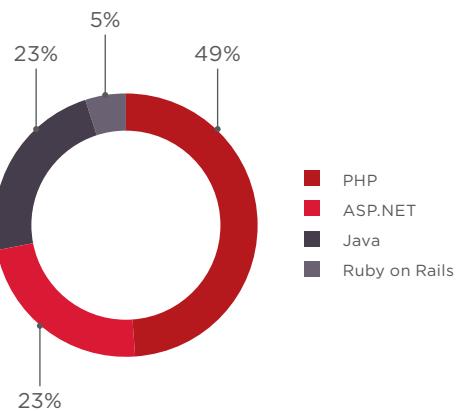


Figure 19. Development tools (percentage of web applications)

## Materials and methods

This report includes data from comprehensive security assessments of 43 fully functional web applications tested in 2018. Results of penetration testing, automated scanning, and testing of e-banking systems are not included here; this information is reviewed in the relevant reports. The dataset does not include applications whose owners did not consent to use of results for research purposes.

The security level of each application was measured using black-, gray-, or white-box methods with the assistance of automated tools. Black-box testing means looking at an application from the perspective of an external attacker who has no prior or inside knowledge of the application. Gray-box testing is similar to black-box testing, except that the attacker is defined as a user who has some privileges in the web application. White-box scanning refers to testing that makes use of all relevant information about the application, including its source code.

Vulnerabilities were classified according to the industry-standard Common Weakness Enumeration (CWE) system. Due to the many types and kinds of vulnerabilities that exist, for reader convenience we have singled out [OWASP Top 10–2017](#) vulnerabilities and analyzed how frequently they occur in the web applications we tested.

The statistics in this report include only code and configuration vulnerabilities. Other widespread security weaknesses, such as failure to install software updates, are not considered here. Our statistics do not include vulnerabilities related to OWASP Top 10–2017 category A10 (Insufficient Logging & Monitoring), since logging and monitoring practices were out of testing scope. Severity was evaluated based on the Common Vulnerability Scoring System ([CVSS v3.0](#)), assigning each vulnerability a rating of Low, Medium, or High (including critical).

### About Positive Technologies

[ptsecurity.com](#)  
info@ptsecurity.com

Positive Technologies is a leading global provider of enterprise security solutions for vulnerability and compliance management, incident and threat analysis, and application protection. Commitment to clients and research has earned Positive Technologies a reputation as one of the foremost authorities on Industrial Control System, Banking, Telecom, Web Application, and ERP security, supported by recognition from the analyst community. Learn more about Positive Technologies at [ptsecurity.com](#).

© 2019 Positive Technologies. Positive Technologies and the Positive Technologies logo are trademarks or registered trademarks of Positive Technologies. All other trademarks mentioned herein are the property of their respective owners.