Threats and vulnerabilities in web applications 2020–2021
# Contents

Introduction .................................................. 3
Executive summary ........................................... 4
Web application vulnerabilities ............................ 5
  Most common vulnerabilities ............................. 7
  Broken Access Control vulnerabilities .............. 8
  Common vulnerabilities not in the OWASP Top 10  9
Web application security level ............................. 10
Web application threats ...................................... 12
  Attacks on users .......................................... 13
  Information leakage ..................................... 14
  Unauthorized access to application .................. 15
Comparing black-box and white-box ..................... 16
Participant snapshot ........................................ 18
Research methodology ..................................... 19
Introduction

Targeting web applications is one of the most common cyberattack methods. According to our research, 17 percent of all attacks involved exploitation of vulnerabilities and security flaws in web applications. Cybercriminals use compromised sites for a variety of purposes: to spread malware; to steal sensitive data; to implant unauthorized information; to commit fraud; to infiltrate a company’s internal infrastructure. All this directly threatens the operation and reputation of organizations, so web applications need to be protected and all weak points eliminated. This report seeks to highlight the main web application threats and vulnerabilities based on the results of comprehensive security analysis. The research methodology is discussed at the end of the report.
Executive summary

About threats:

- In the vast majority of web applications (98%), cybercriminals are able to attack users. Such attacks can result in the spread of malware, redirection to a malicious site, or data theft through social engineering.

- Breaches of sensitive data occurred in 91% of web applications. User IDs were most frequently disclosed (84% of cases). Two-thirds of applications suffered personal data breaches and about half leaked user credentials.

- Exposure to unauthorized access was detected in 84% of web applications. And full control over the target site was gained in 5% of cases.

About vulnerabilities:

- The most dangerous vulnerabilities in web applications were improper user authorization and authentication. These vulnerabilities allow unauthorized access to sensitive information and application features.

- In addition, authorization failures were linked to vulnerabilities in the oAuth protocol. oAuth implementation failures can be exploited by attackers to intercept a session and user credentials, leading to unauthorized access to the application.

- The average number of vulnerabilities per web application decreased by more than a third compared to 2019. On average, there are 15 vulnerabilities per site, two of which are of high severity.

- 15% is the share of high-severity vulnerabilities in the total number of vulnerabilities identified. Two-thirds of sites contain such vulnerabilities.

- 72% of vulnerabilities were related to flaws in web application code.

In 2020–2021, 48% of web applications had low or extremely low security level, a figure that showed no improvement against 2019.
The share of web applications containing high-severity vulnerabilities was 66% in 2020 and 62% in 2021, significantly more than in 2019.
The past two years have seen a slight decrease in the share of high-severity vulnerabilities with potentially very negative consequences in the total number of vulnerabilities. What’s more, the percentage of medium-severity vulnerabilities fell by 23 percentage points, while the proportion of low-severity vulnerabilities more than doubled. In addition, the average number of high- and medium-severity vulnerabilities per application decreased by 1.7 and 2.5 times, respectively, compared to 2019, which indicates a shift to secure development approaches.
Most common vulnerabilities

Vulnerabilities related to Broken Access Control have become the most live in the past two years, adding 63 percentage points, compared to 2019. See below for more about these vulnerabilities.

Security analysis of web applications showed that 83% of them had vulnerabilities associated with security misconfiguration. 48% of analyzed applications were found to have no Strict-Transport-Security header (CWE-523), which requires the user’s browser to use HTTPS for secure data transfer. Having this header configured ensures secure transmission of data, including sensitive information, which in turn protect against man-in-the-middle attacks. In 34% of applications, the X-Content-Type-Options header (CWE-16) was misconfigured, exposing them to content-spoofing attacks, such as Cross-Site Scripting.
Broken Access Control vulnerabilities were identified in all web applications studied in 2020–2021. This type of vulnerability in a web application can lead to unauthorized access to sensitive information, data modification or deletion, as well as unauthorized access to the user’s personal account or application features. Given the potential consequences of exploitation, Broken Access Control vulnerabilities require prompt remedial action.

In one security analysis project, an Improper Authorization (CWE-285) vulnerability made it possible to view other users’ data, such as name, phone number, and email address, by bruteforcing the company’s tax number in a request to a vulnerable script.

Speaking of improper authorization, we should mention the oAuth 2.0 authorization protocol, which was used in some of the projects under study. This protocol is widespread for its convenience over traditional login authorization methods. oAuth uses data from a single account and is completely secure. The user does not have to share their credentials to a third-party application directly, and web application developers need not worry about the security of credentials, since no username and password are used.

However, a misconfigured oAuth protocol can result in theft of personal data and interception of session data, for example, when sending unique access tokens associated with user sessions. In several projects, experts found that it was possible to attack users through Redirection to Untrusted Site (CWE-601). The attacker can send a specially crafted link, which, when clicked, redirects the user to an attacker-controlled resource and passes the user’s session token on the vulnerable site in the URL. In this way, the attacker can get the token and use it to access the user’s personal account.
Among high-severity vulnerabilities, the top two places are taken by improper authorization and authorization bypass with a user key; improper authentication rounds out the top three.

Common vulnerabilities not in the OWASP Top 10

Among the vulnerabilities not listed in the OWASP Top 10 2021, the most widespread is Uncontrolled Resource Consumption (CWE-400), which allows attackers to launch DoS attacks against applications. When analyzing one of the applications, experts managed to exploit this vulnerability by sending requests to generate numerous large reports in PDF format. This, in turn, caused an increase in resource consumption and delays in accessing the application.
Second place goes to Improper Check for Unusual or Exceptional Conditions (CWE-754), when a web application accepts non-standard parameters. In one of the studied applications, this vulnerability was used to extend the trial period. To do so, a request was made to the script for activating the trial period with a modified value, allowing the trial period to be extended by a month. This same vulnerability could be used for mass-sending emails with a malicious attachment seemingly from the application’s support team: in the HTTP request, the payload field contained a file with a .pdf extension and email addresses were specified.

Insertion of Sensitive Information Into Debugging Code (CWE-215) is one of the three most common vulnerabilities not in the OWASP Top 10 list. This vulnerability can provide an attacker with more information about the system, which can be used to select tools and methods for attacks.

Web application security level

Our sample consisted of 16 web applications used in organizations in the industrial sector. Of them, 13 were productive and running in normal mode; the rest were test applications in the development/debugging stage. High-severity vulnerabilities were identified in all test applications. Among the productive applications, 46% had a low or extremely low level of security. In 19% of web applications in the industrial sector, a cybercriminal can attack the LAN and access internal network resources through the web application, and in 31% execute OS commands on the server. For web applications in industry, vulnerabilities related to code injection and vulnerabilities caused by non-secure and outdated components were typical: such vulnerabilities were found in 81% and 44% of applications, respectively. On the whole, the state of web application security in industrial companies shows a positive trend: the share of applications with an extremely low security level decreased by more than three times compared to 2019.

Test systems rarely boast a high level of security. Most often, they are not developed down to the last detail, and system modifications during the tweaking process can result in numerous vulnerabilities.

The security level of 11 government websites were analyzed, six of which were productive. 67 percent of productive applications of government agencies were deemed to have low security level, which is not much different from previous years. The most common vulnerabilities—identified in all applications of government agencies—were related to broken access control. In 70 percent of applications, such vulnerabilities could lead to unauthorized access to the application and leakage of sensitive information, with personal data spillage cited most frequently.

A web application’s security level is determined by expert security analysis and depends directly on the danger level of the potential impact on the system, taking into account the data circulating in it.
The state of web application security in the IT sector showed a negative trend compared to 2019: about half of the productive applications under study had a low or extremely low security level. Moreover, the majority of applications were exposed to code injection vulnerabilities. In 29% of web applications, these vulnerabilities could have dangerous consequences, such as attacks on LAN resources. The IT industry was represented by 15 applications of IT companies—11 productive and four test ones.

We note the increased security level of online commerce sites: not a single application was found to have a low security level. This is due to a greater awareness of web application security on the part of developers, and the growing popularity of online commerce. The most typical threats to online commerce applications are attacks on clients caused by misconfigured security tools, including OAuth implementation failures and sensitive data leaks. It was possible to access user identifiers in all applications, and personal data in 44% of them. Online commerce was represented by nine web applications, eight of which were productive.

Financial institutions were represented by five web applications: four productive and one test application. A characteristic feature of all the applications under study was the presence of vulnerabilities caused by insecure design, in particular, the business logic errors. This made it possible to demonstrate unauthorized access to application features and content in all cases. Based on this number of web applications, an objective assessment of the security level of the entire industry cannot be given. But studies of existing web applications have shown that application security in the financial sector remains fairly high overall, and corresponds to the levels of previous years.

Figure 9. Share of applications by security level (share of all web applications studied)
Web application threats

In 84% of applications under study, threats related to unauthorized access to users’ personal accounts, including those of administrators, were identified. In 72% of web applications, an attacker can gain access to features or content that should not be available to them, such as viewing other users’ personal accounts or changing the length of a subscription trial period.

Most dangerous of all are attacks on local corporate networks and server-side execution of OS commands. Such cybercriminal actions can lead to data disclosure, access to application source code, and, most significantly, access to local network resources and attacks on other infrastructure nodes.

Most dangerous threats
In 17% of web applications, attacks on LAN resources were found to be feasible. Just as many applications contained vulnerabilities that could lead to server-side execution of OS commands.
Attacks on users

Since 2017, number one in the list of live threats to web applications has been attacks on users. One in two attacks on web application users may have been carried out using cross-site scripting (XSS; CWE-79) — a type of attack in which arbitrary code created by the attacker is executed in the user’s browser. XSS allows attackers to redirect web application clients to phishing sites, download malware to user devices, and impersonate other users.

Figure 11. Vulnerabilities allowing attacks against users

In addition, a misconfigured or missing X-Frame-Options header (CWE-16) has become a cause of attacks on users. This header is responsible for allowing or blocking the display of the page if it is in a frame. This header misconfigured or missing can lead to a clickjacking attack, whereby a vulnerable application is loaded in a frame and disguised as a button or other element, or not displayed to the user at all (a transparent iframe). By clicking this element, a user performs the attacker-chosen action in the context of the vulnerable website.

To protect against such attacks, we recommend preprocessing all data arriving from the user and then displayed in the browser. Characters that could potentially be used for HTML page formatting should be replaced with non-formatting characters. We also recommend taking more care with HTTP headers and using application-level firewalls to protect against XSS.

XSS comes in three flavors:

1. Stored XSS, in which malicious code is pre-injected into the application page.
2. Reflected XSS, in which a successful attack requires the user to click a specially crafted link.
3. DOM-based XSS, in which the code of a web application page processes transmitted data and attempts to execute the received data set as programming language commands.
Information leakage is the second most acute security threat to the web applications under study. The results of the security analysis showed that more than three-quarters of web applications were vulnerable to disclosure of user IDs. Personal data was disclosed in 60% of applications, and user credentials in 47%, up 13 and 16 percentage points on 2019, respectively. Personal data and credentials are desirable targets for attackers, as confirmed by the Cyberthreats Analysis 2021 report.

The threat of disclosure of sensitive information is often associated with improper authorization and authentication vulnerabilities.

When investigating one of the sites, experts gained access to sensitive information about web application users (CWE-359). This was made possible due to a vulnerable form for outputting a list of application users. Through a specific HTTP request, the names, email addresses, identifiers, and password hashes of all users were retrieved. From the information received, the passwords of almost 400 users were recovered. Such information could be used by cybercriminals for authorization in the application with appropriate privileges.
Unauthorized access to application

Unauthorized access to application was identified in 84 percent of web applications. Unauthorized access to the application is attainable due to authentication vulnerabilities, unlimited session lifetime, leaks, or lack of protection against brute-force attacks—in one of the applications, experts found that the application allowed users to set weak passwords that are easily crackable. By bruteforcing credentials, experts managed to gain access to the application with the privileges of the corresponding user, as well as to bruteforce and obtain other users’ promo codes.

Nowadays, a single password rarely provides the required level of security. We recommend that organizations use multi-factor or two-factor (2FA) authentication to protect web services against cyberattacks. However, these processes too are prone to vulnerabilities, so care needs to be taken in implementing them. In one of the web applications, experts identified an improper 2FA implementation (CWE-287) allowing them to intercept an HTTP request to send a confirmation code to the server. Using this intercepted request, they created a template for generating HTTP requests with different code values. These requests were then sent to the application server in a matter of seconds. All invalid requests were rejected by the server, but when the correct confirmation code was entered, all other invalid codes were accepted by the application. After refreshing the page, the 2FA code request form disappeared and access to the user’s personal account was granted.

Unauthorized access to application can lead to sensitive information leakage, data modification or deletion, attacks on users, and penetration of corporate infrastructure.
Comparing black-box and white-box

In 2020–2021, black-box or gray-box testing was most popular among clients; 79% of web applications were analyzed using these methods. Black-box testing involves analyzing and evaluating the security of a web application with no source data other than its address. Gray-box testing is essentially the same as black-box, except that investigators have information about points of entry into the system, plus an authorized user account with standard privileges.

![Figure 13. Web application testing methodology (share of web applications)](image)

Expert appraisal shows that many website vulnerabilities are due to coding flaws: in the past two years, 72% of vulnerabilities discovered were related to vulnerable code in web applications, such as SQL injection, XSS, and incorrect condition checking or exception handling. The other vulnerabilities were caused by improper administration and are fixable in the application settings. To rule out code-related vulnerabilities, we strongly recommend that organizations incorporate a secure development process in the web application lifecycle and use a code analyzer. For a more thorough survey of application vulnerabilities, white-box testing is used.

In white-box testing, the customer provides all available data about the application, including source code, for subsequent analysis.
Web applications subjected to white-box testing are analyzed by several researchers at once; alongside automated security analysis, manual code verification is carried out, which, although more time-consuming, eliminates false positives and guarantees that identified vulnerabilities are live. All this allows experts to thoroughly study the application and not miss a single line of code or hidden flaw. During the research, it was found that white-box testing detected twice as many high-severity vulnerabilities as black-box or gray-box.

**Conclusion**

To sum up the past two years, it must be said that the level of web application security still leaves much to be desired. However, we have observed a positive trend in the decrease in the average number of high- and medium-risk vulnerabilities per application, which indicates the adoption of secure development approaches and a more serious attitude toward web application protection. We hope in the near term to see the level of website security increase, and the number of identified vulnerabilities and successful attacks fall.

Achieving a high security level is hard, and maintaining this level is an equally labor-intensive process. Careful and secure web application design, development with source code analysis, quick remediation of vulnerabilities, and maximum automation of processes are our guiding principles to ensure a high level of web application security.

For security analysis, we recommend using productive copies of the web application. Test systems are usually not fully developed and lack full functionality, and various system modifications can cause new vulnerabilities to appear.

On top of security analysis, organizations need to take a proactive approach to protection and use a [web application firewall (WAF)](https://www.positivetechnologies.com) to mitigate the risks related to the ongoing introduction of new application features that may contain potential vulnerabilities. For its part, besides detection and prevention of known attacks, a WAF should detect exploitation of zero-day vulnerabilities, prevent attacks on users, analyze and correlate multiple events to identify attack chains, and block malware.
Participant snapshot

The study covered 58 web applications analyzed in the past two years.

74% of web applications under study were productive; 26% were in testing.

Figure 15. Study participants by sector

Figure 16. Web application development tools (percentage of applications)
Research methodology

The results of the 2020–2021 study of 58 web applications for which in-depth security analysis with a full range of checks was performed. The study does not include the results of security analysis of mobile applications, online banking, penetration testing, and automated scanning (see our website for information about such research). The owners of web applications included in the study have consented to the use of their data for research purposes.

The security level of each application was assessed manually, using black-, gray-, or white-box testing with the assistance of automated tools.

Vulnerabilities were classified according to the industry-standard Common Weakness Enumeration (CWE) system. For the sake of convenience, vulnerabilities from the OWASP Top 10 2021 are highlighted, since the classification of vulnerabilities is quite detailed, plus it is shown how often these vulnerabilities occur in the web applications we studied.

The report covers only vulnerabilities related to web application code and configuration. Other common security weaknesses, such as failure to manage software updates, are not considered. Severity was evaluated based on the Common Vulnerability Scoring System (CVSS v3.1), assigning each vulnerability a rating of Low, Medium, or High.