EC-Council

GUIDE TO CURRENT WEB APPLICATION PENETRATION TESTING PRACTICES

EC-Council would like to thank **Rakesh Sharma**, Vice President of Cloud and Container Security at Standard Chartered Bank, for authoring this whitepaper.

Abstract

Web application penetration testing (WAPT) practices vary with region, industry, organization size, and other factors. WAPT is an important aspect of cybersecurity, given that many data breaches are attributable to weak web security configurations. Potential web application security threats include unsecure coding practices, use of vulnerable open-source components, web server misconfigurations, and a lack of periodic vulnerability assessments. If a website goes live in production before it is thoroughly tested for security flaws, flaws may be exposed and exploited by threat actors, potentially leading to substantial data breaches. This white paper provides an overview of key WAPT practices and tools, recent data on web application security risks, common critical web application vulnerabilities, and guidance for organizations on how to address their web application security challenges.

Keywords: web application penetration testing, WAPT, web application security, penetration testing

Contents

Guide to Current Web Application Penetration Testing Practices

Media organizations, public services, government agencies, educational institutions, corporations, and nonprofits are all subject to cyberattacks. High-profile clients, partners, and businesses are especially vulnerable targets. Attack methodologies change as the technology and threat landscape evolves and cyber adversaries continually employ novel exploitation techniques. Enterprises have a difficult time dealing with security incidents, and law enforcement officials find it challenging to catch attackers in what has become a cat-and-mouse game, with cybercriminals remaining one step ahead most of the time.

Web applications and technologies have become a core component of digital infrastructure for companies that are creating industry-leading solutions and accelerating their digital transformation initiatives. However, this increase in web application adoption also means an increase in cybersecurity vulnerabilities. This has been especially true in recent years, as the COVID-19 pandemic was accompanied by a sharp spike in cyberattacks. During 2020, attacks on web applications increased by 800%, while distributed Denial of Service attacks increased by 147.63% over the previous year (Wilson, 2020). Throughout 2021, more than half of the websites tested by NTT application security researchers were vulnerable to at least one serious exploit (WhiteHat Security, 2022).

Application programming interfaces (APIs), microservices, serverless web applications, and corporate networks are all susceptible to the many threats associated with the advent of 5G networks, artificial intelligence and machine learning, the cloud, the Internet of Things, and other emerging technologies. Web application penetration testing (WAPT) is one of the most widely used techniques to assess the security of web applications, as it enables cybersecurity professionals to discover and exploit web vulnerabilities before malicious hackers can do the same. The need for professional penetration testing services cannot be understated. Penetration testers assist organizations in identifying vulnerable assets, uncovering hidden vulnerabilities, pinpointing cyberthreat origins, and reducing their attack surface. Web application vulnerability assessment and WAPT help to build a secure cyberinfrastructure guided by established industry frameworks (e.g., OWASP, NIST 800-115, PTES) to ensure that organizations have defenses against unforeseen hacks, data breaches, and other cybercrimes.

Web applications are the digital assets most affected by data breach incidents globally. As far back as 2003, cyberattacks occurred every 39 seconds on average (Talalaev, 2021). Since then, the size and scope of the internet have grown such that it is extremely difficult for researchers to calculate the frequency of web-based cyberattacks with any accuracy. However, other types of statistics shed light on the evolution of web application cyberattack trends. For example, more than 30,000 websites were identified as new targets daily, Sophos Labs reported in 2013 (Lyne, 2013). In the 5-year period leading up to the release of a 2019 report, the number of security threats rose by 67%, according to Accenture (Talalaev, 2021). More than 56% of compromised content management systems (CMSs) were outdated at the point of infection in 2019, according to Sucuri (Macleod, 2020). Lack of software updates, patches, and the use of weak or easily guessable passwords are also common problems that lead to web application attacks. Hackers often target WordPress-based websites and web apps because of the platform's large userbase and because WordPress security updates may not be applied to its latest themes, plugins, and extensions (Talalaev, 2021). In a 2019 study, researchers found that hackers could attack users in nine out of 10 web applications, malicious actors could gain access to applications on 39% of websites, and 68% of web applications were at risk of data breaches (Positive Technologies, 2020). In 2018, researchers discovered approximately 70 types of vulnerabilities in web applications (Positive Technologies, 2019). Eighty-seven percent of websites had medium security vulnerabilities in 2018 and 46% had high security vulnerabilities, one study revealed (Acunetix, 2019). That same year, the PT Security team found that four out of five web applications had configuration errors, including the use of default settings and standard passwords, lack of error reporting, full path disclosure, and other information leakage issues (Positive Technologies, 2019). Since the start of the COVID-19 pandemic, the use of web applications has increased as organizations shift to remote working arrangements and their employees become more distributed and mobile. Consequently, threats related to web application security have increased as well, making WAPT a critical part of many organizations' cybersecurity strategies.

Steps Involved in Web Application Penetration Testing

The WAPT process entails four key steps: information gathering; research and exploration; scanning, reporting, and recommendation; and threat remediation (Struk, n.d.).

Step 1: Information Gathering

The two phases of information gathering are active reconnaissance and passive reconnaissance. Active reconnaissance involves probing the target system and making it produce outputs as a result of penetration tests. DNS reverse lookup, forward transfers, Shodan port scanning, and web application fingerprinting are examples of active reconnaissance. Passive reconnaissance involves retrieving and analyzing information about the target system and organization that is available online and sifting through it for the purpose of mounting an attack without communicating directly with the target IT infrastructure (e.g., computers, networks, email system) or the organization's employees. Open-source intelligence (OSINT)—that is, data publicly available on the internet or through other means—is the primary source of information used in passive reconnaissance. Multiple OSINT sources can be used together to gather helpful information before conducting WAPT. These sources may include Whois lookup information, previous versions of websites retrieved from the Internet Archive's Wayback Machine, publicly available information about the organization and its employees posted on social media websites, government database records, leaked information, niche search engines such as Shodan, and customized Google searches.

Step 2: Research and Exploration

Using the intelligence gathered in Step 1, security professionals then begin to research possible exploits and uncover hidden attack surface vectors, such as an unpatched web server operating system or an outdated web application component. The goal of the research phase is to identify, prioritize, and categorize threats. In addition, as part of the exploration process, consideration is given to which tools to use for remediating threats. Once security professionals have defined the scope of the problem, they can determine what security measures to employ to address them.

Step 3: Scanning, Reporting, and Recommendation

The next step is scanning the web application for vulnerabilities and threats. Security professionals carefully examine all ports, domains, subdomains, web hosts, URLs, parameters, and parameter values in applications. After identifying all the vulnerabilities, they prepare a penetration test report to document their findings in clear, concise language. They outline how to remediate the threats and assign them priority levels. They provide written suggestions and recommendations for preventing future exploitation, which they present to clients. Most reports are business oriented and are forwarded to the organization's IT and management departments for review and to increase visibility of all the security risks the organization faces.

Step 4: Threat Remediation

During the final step, professionals remediate web application security issues that might allow unauthorized users to access sensitive information on networks. Security professionals work closely with IT teams to help resolve these vulnerabilities. Once the threats are dealt with, the teams patch or update systems and implement a data backup and recovery plan.

Common Web Application Security Vulnerabilities

Some of the most common critical vulnerabilities and security issues found in the web application domain are listed below.

1 Remote Code Execution

Remote code execution (RCE) occurs when an attacker gains access to a system with execute permissions and acquires the same capabilities to inject and run code on servers as a local user on the system. Typically, attackers are able to hide their presence well since they have local administrative access, making uncovering RCE cases nearly impossible in some instances. Risks associated with RCE threats include information theft, compromise of host devices and systems, propagation of malware, setup of backdoors, and attackers gaining the ability to move laterally, potentially gaining complete control over cloud services and private enterprise networks.

2 Weak Passwords and Brute Force Attacks

Brute force attacks increased dramatically in 2021 (Bisson, 2021), with hackers finding new ways to uncover vulnerabilities at security entry points. A stringent password policy helps, but many employees neglect personal cybersecurity hygiene and set weak passwords. Malicious emails may coax targets into making their login credentials available to online exchanges. Credential stuffing and password-spraying attacks are also consequences of weak password usage.

3 Source Code Disclosure

Source code disclosures occur when attackers gain access to server-side source code from the web browser without having authorized access to the code by design. Server-side source code exposure is not intentional. It can be disclosed to clients when errors occur in scripts or due to misconfigurations. Accessing source code can help attackers understand an organization's business logic and discover security flaws in code implementation for further exploitation.

4 Overflow Vulnerabilities

Overflow vulnerabilities, or buffer overflow vulnerabilities, may be found within the code of web application systems. Sometimes developers make mistakes while designing web applications, and the errors become obvious in the code. In extreme cases, a hacker can exploit the code to gain administrative access to systems and halt enterprise operations completely.

5 TLS/SSL Vulnerabilities

Hackers can launch attacks against SSL versions with weak cryptographic implementations and exposure to OnPath attacks. Sometimes hackers trick users into downgrading to vulnerable SSL versions. SSL 3.0 is a vulnerability associated with cipher block chaining mode. It occurs when hackers modify padded content found in block ciphers and intercept credentials. In these cases, the hacker impersonates the server and intercepts the handshake signals sent by clients. Other common SSL/TLS vulnerabilities include Heartbleed, CRIME, BEAST, and POODLE (Prodromou, 2019).

6 Content Management System Vulnerabilities

Vulnerabilities in WordPress and other CMSs are complex because of the wide range of themes, plugins, and features these platforms use. WordPress does provide automatic updates, but the add-ons and plugins used within its webpages constitute the biggest security risk. Developers have different security policies for these add-ons, themes, and extensions. The attack surface expands depending on what plugins or features developers decide to incorporate into their WordPress websites. RCE and cross-site scripting (XSS) are the most common attacks associated with WordPress vulnerabilities.

7 Cross-Site Scripting

Every web browser executes scripts supplied as user inputs on websites, and XSS occurs when attackers host malicious JavaScript code on platforms that users then download and execute. The three most common types of XSS vulnerabilities are stored XSS, DOM-based XSS, and reflected (nonpersistent) XSS. Stored XSS stores malicious scripts on the website that are executed when users click on links or take other actions. Reflected XSS uses social engineering techniques to redirect users to vulnerable websites and trick them into running malicious code. DOM-based XSS takes advantage of the Document Object Model and executes injected scripts into single-page web applications or on-site engines that are JavaScript-heavy, with scripts not being stored on the server side and request input not being sent to the web server.

8 SQL Injections

Large companies—including Yahoo, Google, IBM, and Sony Pictures—have been hacked via SQL injections in the past. SQL injection is a type of attack in which hackers exploit vulnerabilities and security weaknesses in web applications to inject malicious SQL code into underlying databases. Attackers may use OSINT to find vulnerable websites and payloads to target.

9 Host-Header Injections

Servers often host multiple websites or web applications at the same IP address. The host header directs incoming HTTP requests to the specified site or application. Vulnerabilities can cause a host header to process requests in unsafe ways. If a header is not properly validated, an attacker may be able to provide invalid inputs and make the web server crash, leading to web cache poisoning and redirecting users to malicious domains.

10 ASP.NET and PHP Vulnerabilities

ASP.NET applications run on Internet Information Services web servers. WordPress websites are written in PHP, an open-source programming language. Both are vulnerable to security flaws. WordPress sites tend to be unsafe because of the number of plugins, extensions, themes, and integrations used. Vulnerabilities are typically introduced in active development updates or open-source libraries and other components.

How to Address Common Web Application Penetration Testing Challenges

One of the biggest challenges for web application developers is detecting vulnerabilities and flaws during the design and development stages of the software development life cycle (SDLC). Failing to address web application vulnerabilities in a timely manner can lead to huge financial losses. Many web application security threats are preventable, particularly if automated scanning is used during vulnerability assessments. Below are some of the most common ways to address major web application security vulnerabilities (Software Testing Help, 2022).

Enterprises can prevent SQL injection attacks by defining preset or parameterized queries to sanitize user inputs in web applications. Prepared statements are an effective way of separating SQL code and SQL data, thus eliminating any vulnerability associated with SQL injections. LIMIT and various SQL controls can help as well.

XSS vulnerabilities can be mitigated by implementing a content security policy and applying context-based encoding to application page rendering. Escaping special characters and input validation are key to fixing XSS issues. Adding allow list filters is another effective strategy, since attackers are capable of bypassing deny lists. Whenever possible, web applications should be designed using the latest technology frameworks—such as Ruby on Rails, Angular, and React—since these technology stacks are designed to address most modern web application security vulnerabilities.

Adaptive hashing algorithms like bcrypt and argon2 can prevent authentication mechanism vulnerabilities. Putting a limit on the number of login attempts for web applications and setting strong passwords are important tactics for preventing authentication hacks. It is a good practice to change passwords often and not use the same password across all systems and user accounts on the network.

XML external entity attacks are best prevented by ensuring that configuration settings are solid for XML parsers and web applications. Disabling document type definitions is another way of eliminating these threats. Whenever possible, it is ideal to use less JSON code for web applications and avoid serializing data for less complexity.

It is important to perform regular file integrity checks and secure HTTPS connections to transmit data across web pages, forms, and other online channels. Using secondary encryption protocols like TLS and IPSEC is a good practice, as is removing any unnecessary additional elements or components in web applications. Web applications can get bogged down with features that are not required and serve only to increase the attack surface. Often, the simpler a design is, the better its functionality and the less complex its threat model. Automated configuration monitoring tools may be helpful in reducing bloat. Insider security threats and accidental exposure of sensitive data can be prevented by incorporating adequate employee security awareness training programs and instituting prudent hiring practices, such as background checks. Security auditors, red teams, and ethical hackers should collaborate with IT security staff to address enterprise security flaws and challenges, along with potential solutions, and structure security education programs based on team reports and findings.

Web application systems and services should be patched often and kept up to date. To avoid misconfiguration issues, regularly conducting vulnerability scans is necessary and should not be overlooked.

Access rights to user accounts should be restricted based on the principle of least privilege, and serialized data on networks should be deserialized. IT and security professionals should collect network and application logs often and enable continuous monitoring to spot malicious activities and avoid letting threats go undetected.

Conducting a Web Application Penetration Test

1 The Cost of a Web Application Penetration Test

The starting cost of a penetration test for an enterprise is anywhere between USD 4,000 and 10,000. This cost may go up depending on the initial assessment results, number of webpages in the application, and user matrix. Companies should also consider the amount of time taken to run tests and to design threat remediation plans afterward. For a web application with a single page or form, the cost of performing a penetration test will be far less than it would be for an application with multiple features, roles, and pages. There may be other security variables involved with WAPT, which may increase the total cost of the penetration test.

2 Common Web Application Penetration Testing Tools

It is essential to mandate use of the latest and most widely recognized WAPT tools and to ensure they meet industry standards for addressing security flaws or gaps in web applications. Some of the most popular tools used for WAPT purposes by security professionals are listed below (Jevtic, 2019).

Burp

Burp Suite is a hands-on WAPT package designed for testing workflows and launching new projects. It features an embedded browser that does not require any manual configuration. Users must install a CA certificate to run it on the external browser. Once users have Burp configured, they can view sent HTTPS requests on the Intercept tab. The embedded browser mode enables security professionals to perform reconnaissance and analysis, as well.

AppScan

AppScan is a fast, accurate, and agile WAPT solution for industry professionals. It integrates easily into DevOps workflows, offers built-in application security testing tools, and supports multiple programming languages. The software can crawl target web applications, automatically identify vulnerabilities, and provide actionable insights in the process. It supports compliance with standards such as PCI DSS, SANS 25, OWASP Top 10, and others.

Qualys

Qualys, based in the United States, offers a platform that is built on the cloud. It deals with vulnerability management use cases and provides advanced security configurations. The benefits of using Qualys include configuring scans and reports, running investigations on zero-day vulnerabilities, reconfiguring scanners, and generating executive reports. The Qualys Cloud Community Edition is free for all users, but the Qualys Cloud Enterprise Edition includes extra features for a price.

Nmap

Nmap is used to test open ports and running services at entry points and conduct vulnerability assessments. It is a network scanner and discovery tool used by professionals to perform security audits. Experts can use Nmap to collect detailed information about target hosts and complete a thorough security analysis.

Metasploit

Metasploit is a customizable open-source framework used by ethical hacking professionals to test target systems and discover vulnerabilities. It is compatible with most operating systems and features add-on support for multiple languages. Penetration testers can benefit from its ready-made code and discover blind spots in systems during the threat-hunting process.

Acunetix

Acunetix offers a 360-degree view of an organization's security posture and is designed to eliminate web application security risks. The U.S. Air Force, AWS, and Cognizant are some of its top clients, and it is used by more than 2,300 companies worldwide for web application vulnerability assessment purposes. Users can schedule recurring scans and scan multiple environments in one go. The user interface is intuitive, making the platform very user friendly.

Zed Attack Proxy

Zed Attack Proxy (ZAP), developed by OWASP, is one of the world's leading free web application security testing tools. It is managed by a team of community volunteers and regularly updated. Its extensive features include providing support for custom payloads, scanning APIs, exploring with Ajax Spider, active scanning, and automated authentications.

Iron Wasp

Iron Wasp is a tool used for finding bugs and debugging websites. It has a user base of more than thousands worldwide. It features dynamic JavaScript analysis support, web socket message analysis, scripting APIs for Python and Ruby-based web applications, and online cross-site web socket hijacking testing. It is entirely open source and free to use. Its features can be extended by using additional plugins and modules. Iron Wasp can detect false positives, CSFR tokens, broken authentication, privilege escalations, and a host of other security issues. A report is automatically generated for users in RTF and HTML formats for later reviewing, and the platform works with several operating systems.

Sonar Qube

Sonar Qube has an interactive graphical user interface and can analyze web applications in over 20 different languages. It can test the strength of source code in web applications and identify malicious code and hardcoded secrets using rule patterns. Following initial assessments, it makes immediate recommendations on how to remediate threats.

Conclusion

Security is a top organizational priority, and enterprises must incorporate it into their web application design. Many companies are coming up with innovative security solutions for web applications to stay competitive and combat hackers breaching their systems. Awareness of the key challenges associated with web application security vulnerabilities is critical, and developers are working on analyzing and studying flaws to ensure that they address them in upcoming releases, starting in the initial stages of the SDLC. A secure web application can help a company operate safely, ensure business continuity, boost reputation, and garner clients' trust, thus helping to build a better brand image.

References

Bisson, D. (2021, December 7). Credential phishing, brute force attacks both increased in H1 2021. ***Security Intelligence***. **https://securityintelligence.com/news/credential-phishing-brute-force-attacks-increased-2021/**

Jevtic, G. (2019, May 9). 17 best security penetration testing tools the pros use. ***NAP Global IT Services.*** **https://phoenixnap.com/blog/best-penetration-testing-tools**

Lyne, J. (2013, September 6). 30,000 websites hacked a day: How do you host yours? ***Forbes***. **https://www.forbes.com/sites/jameslyne/2013/09/06/30000-web-sites-hacked-a-day-how-do-you-host-yours/**

Macleod, R. (2020, January 28). Hacked website threat report – 2019. ***Sucuri Blog***. **https://blog.sucuri.net/2020/01/hacked-website-threat-report-2019.html**

Positive Technologies. (2019, March 5). ***Web application vulnerabilities: Statistics for 2018***. **https://www.ptsecurity.com/ww-en/analytics/web-application-vulnerabilities-statistics-2019/**

Positive Technologies. (2020, February 13). ***Web applications vulnerabilities and threats: Statistics for 2019***. **https://www.ptsecurity.com/ww-en/analytics/web-vulnerabilities-2020/**

Prodromou, A. (2019, March 31). TLS security 6: Examples of TLS vulnerabilities and attacks. ***The Acunetix Blog***. **https://www.acunetix.com/blog/articles/tls-vulnerabilities-attacks-final-part/**

Software Testing Help. (2022, June 13). ***Beginners guide to web application penetration testing***. **https://www.softwaretestinghelp.com/getting-started-with-web-application-penetration-testing/**

Struk, V. (n.d.). Your 2022 guide to web application penetration testing. ***Relevant***. **https://relevant.software/blog/penetration-testing-for-web-applications/**

Talalaev, A. (2022, February 21). Website hacking statistics you should know in 2022. ***Patchstack***. **https://patchstack.com/articles/website-hacking-statistics/**

WhiteHat Security. (2022, February 18). ***New report shows half of websites were vulnerable to exploitation throughout 2021*** [Press release]. **https://www.whitehatsec.com/news/new-report-shows-half-of-websites-were-vulnerable-to-exploitation-throughout-2021/**

Wilson, J. (2020, November 25). Cyber-attacks on web applications up by 800 per cent in H1 2020: Report. ***Canadian Occupational Safety***. **https://www.thesafetymag.com/ca/topics/technology/cyber-attacks-on-web-applications-up-800-per-cent-in-h1-2020-report/240124**

EC-Council

www.eccouncil.org