



AppCheck vs OWASP Top Ten

Based on a broad consensus, the OWASP Top Ten defines the current most critical web application security flaws.



The following table outlines how AppCheck identifies these vulnerabilities and helps to mitigate risk.

Category	Description	AppCheck Coverage
<p>A1:2017-Injection</p> 	<p>Injection attacks are the most common type of fault found in web applications, they are usually the result of unfiltered user input being directly included into command executions or database queries.</p>	<p>AppCheck performs comprehensive checks for a wide range of injection vulnerabilities including:</p> <ul style="list-style-type: none"> • SQL Injection • NoSQL Injection • XPath Injection • Code Injection • Command Injection • LDAP Injection • Expression Language Injection <p>The AppCheck Vulnerability Analysis Engine provides detailed rationale behind each finding including a custom narrative to explain the detection methodology, verbose technical detail and proof of concept evidence through safe exploitation.</p>
<p>A2:2017-Broken Authentication</p> 	<p>Sometimes authentication can be implemented incorrectly or an application can contain routes to sensitive data that haven't been correctly protected by an authentication barrier. In other cases, it can be the session token that is vulnerable either to enumeration or not expiring, this can allow an attacker to guess the session token of another user (e.g. an administrator) and take control of their session to steal data.</p>	<p>While crawling an application, AppCheck analyses session tokens to identify security flaws such as insufficient entropy and other common session management flaws that could permit session token prediction.</p> <p>AppCheck also includes configurable password guessing modules to identify weak account credentials within systems such as:</p> <ul style="list-style-type: none"> • HTML Form Authentication • Outlook Web Access (OWA) • Content Management Systems (e.g. WordPress) • NTLM/Basic Authentication • Management systems and SSL VPN gateways

AppCheck vs OWASP Top Ten



Category

Description

AppCheck Coverage

A3:2017-Sensitive Data Exposure



This is usually the accidental exposure of files or folders that should not be publicly accessible, for instance a hidden folder called invoices provided for the convenience of remote workers or a hidden “.git” directory accidentally served up from the root directory of the web server which contains all the source code for the application.

AppCheck includes multiple plug-ins to identify sensitive data disclosure vulnerabilities including:

- Insufficiently protected administrative interfaces
- Publicly accessible source code repositories such as GIT and SVN
- Publicly accessible archives and files containing sensitive information
- Verbose error and Stack Trace output containing sensitive information
- Source code disclosure
- Hidden folders and files (forced browsing)
- Backup and temporary files detection

A4:2017-XML External Entities (XXE)



Many older or poorly configured XML processors evaluate external entity references within XML documents. External entities can be used to disclose internal files using the file URI handler, internal file shares, internal port scanning, remote code execution, and denial of service attacks.

AppCheck includes a comprehensive XXE module that identifies all forms of XXE and related XML injection vulnerabilities. A variety of techniques are employed such as Out-of-Band detection using DNS side channels and Signature based detection.

A5:2017-Broken Access Control



Similar to “Broken Authentication and Session Management” this is where routes / views within the application are not properly protected. For example it’s not uncommon to see that admin controls are just hidden from the application menu and that the function is not actually restricted from an average user, the application is just relying on it not being visible.

Access control mechanisms are validated by attempting to access components that should be restricted or should require prior authentication but fail to protect the resource.

Insecure or superficial access control systems that simply hide components but do not properly secure them are also identified.

AppCheck vs OWASP Top Ten



Category

Description

AppCheck Coverage

A6:2017-Security Misconfiguration



This is often out of date or un-patched frameworks or the stack on which the framework sits, often it can be a case of changing the settings within the stack to harden the security of the setup. For instance many default web server SSL setups make ciphers available with known vulnerabilities.

AppCheck maintains a database of common configuration faults and out of date or un-patched frameworks and will flag these if detected.

If configured to do so, AppCheck will perform a comprehensive infrastructure assessment against all IP addresses and web applications defined within the scope.

A7:2017-Cross-Site Scripting (XSS)



Cross site scripting is a type of injection attack where by an attacker is able to inject JavaScript content into an application that runs in a user's browser. Often thought of as an attack against the users of an application rather than the application itself, some more complicated XSS attacks target the administration and backend systems of an application (2nd order attacks).

AppCheck includes unparalleled XSS detection capabilities that have been credited by Google, Microsoft and eBay (among others) via their bug bounty programs. Our detection methodology emulates the actions of a skilled consultant by building from suspected case to a fully exploitable vulnerability whilst avoiding common input filters that cause other scanners to fail.

All XSS variants are covered including;

- Reflected & Stored XSS
- DOM Based XSS
- HTML5 postMessage XSS
- Adobe Flash XSS
- Second order / Delayed Execution XSS

A8:2017-Insecure Deserialization



Insecure deserialization often leads to remote code execution. Even if deserialization flaws do not result in remote code execution, they can be used to perform attacks, including replay attacks, injection attacks, and privilege escalation attacks.

AppCheck will identify and safely exploit both generic and specific deserialization vulnerabilities across a wide variety of frameworks and libraries.

AppCheck vs OWASP Top Ten



Category

Description

AppCheck Coverage

A9:2017-Using Components with Known Vulnerabilities



With the rise of the huge number of 3rd party components freely available on the internet for inclusion in applications, it's not uncommon for a developer to find a component or library and include it in an application to solve a problem or provide a widget. However vulnerabilities are often discovered in these components and either newer versions are released or they have been abandoned.

AppCheck includes a regularly updated database containing thousands of known vulnerabilities within content management systems, application frameworks, server and client-side components.

The following dedicated assessment components are also provided by AppCheck:

- CMS Build review for; Umbraco, WordPress, Drupal, Magento, Joomla and DNN
- Web Server & Proxy vulnerability checks for nginx, Apache, IIS, Tomcat, Struts, F5 Load balancers plus many more
- Scanning for known server-side script vulnerabilities
- Client-Side JavaScript library checks to identify vulnerable and unsupported components

A10:2017-Insufficient Logging & Monitoring



Insufficient logging and monitoring, coupled with missing or ineffective integration with incident response, allows attackers to further attack systems, maintain persistence, pivot to more systems, and tamper, extract, or destroy data. Most breach studies show time to detect a breach is over 200 days, typically detected by external parties rather than internal processes or monitoring.

Through creating a realistic attack scenario, AppCheck helps to flex monitoring and logging solutions and so can highlight weaknesses and omissions in current processes, for which our security team are always on hand to offer advice on best practice.