OWASP Security Evaluation

Introduction

The OWASP Top-10 covers the most critical Web application security risks. The OWASP Top 10 Web Application Security Risks, as of the 2017 list, are:

- **A1: Injection:**
    - Injection flaws, such as SQL, OS, and LDAP injection, occur when untrusted data is sent to an interpreter as part of a command or query. The attacker's hostile data can trick the interpreter into executing unintended commands or accessing unauthorized data.
- **A2: Broken Authentication**
    - Application functions related to authentication and session management are often implemented incorrectly, allowing attackers to compromise passwords, keys, or session tokens, or to exploit other implementation flaws to assume other users' identities temporarily or permanently.
- **A3: Sensitive Data Exposure**
    - Many web applications and APIs do not properly protect sensitive data, such as financial, healthcare, and PII. Attackers may steal or modify such weakly protected data to conduct credit card fraud, identity theft, or other crimes. Sensitive data may be compromised without extra protection, such as encryption at rest or in transit, and requires special precautions when exchanged with the browser.
- **A4: XML External Entities**
    - 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.

- **A5: Broken Access Control**
    - Restrictions on what authenticated users are allowed to do are often not properly enforced. Attackers can exploit these flaws to access unauthorized functionality and/or data, such as access other users' accounts, view sensitive files, modify other users' data, change access rights, etc.
- **A6: Security Misconfiguration**
    - Security misconfiguration is the most commonly seen issue. This is commonly a result of insecure default configurations, incomplete or ad hoc configurations, open cloud storage, misconfigured HTTP headers, and verbose error messages containing sensitive information. Not only must all operating systems, frameworks, libraries, and applications be securely configured, but they must be patched/upgraded in a timely fashion.
- **A7: Cross-Site Scripting (XSS)**
    - XSS flaws occur whenever an application takes untrusted data and sends it to a web browser without proper validation and escaping. XSS allows attackers to execute scripts in the victim’s browser which can hijack user sessions, deface web sites, or redirect the user to malicious sites.
- **A8: 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.
- **A9: Using components with known vulnerabilities**
    - Components, such as libraries, frameworks, and other software modules, run with the same privileges as the application. If a vulnerable component is exploited, such an attack can facilitate serious data loss or server takeover. Applications and APIs using components with known vulnerabilities may undermine application defenses and enable various attacks and impacts.
- **A10: Insufficient logging and 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.

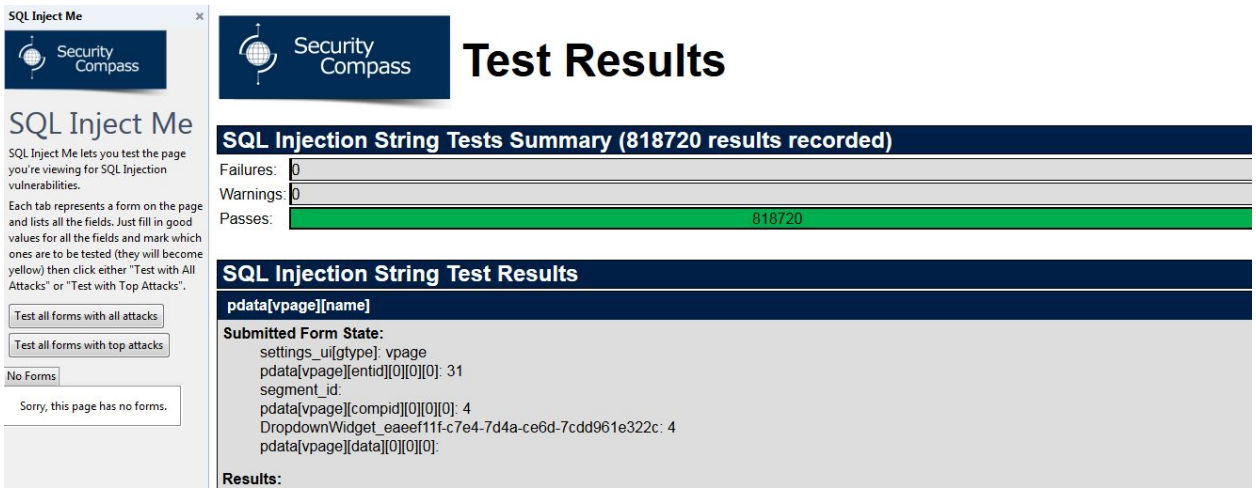
Tools Matrix

Find an open-source tool for each of the OWASP Top-10 security risks for Convert Experiences App and document what needs to be done/fixed. Following is a risk and tool matrix.

RISK	TOOL	FAILURES
A1: Injection	SQL Inject Me	0 failures detected
A2: Broken Authentication	HackBar	0 failures detected
A3: Sensitive Data Exposure	Burp Suite Professional	0 failures detected
A4: XML External Entities (XEE)	Burp Suite Professional	0 failures detected
A5: Broken Access Control	Burp Suite Professional	
A6: Security Misconfiguration	Burp Suite Professional	0 failures detected
A7: Cross-Site Scripting (XSS)	ZAP	0 failures detected
A8: Insecure Deserialization	Detectify	
A9: Using components with known vulnerabilities	HDIV	
A10: Insufficient logging & monitoring	HDIV/Detectify	

A1: Injection – SQL Inject Me (v0.4.71)

SQL injection is both prevalent and of severe impact security risk. The Firefox add-on **SQL Inject Me** is useful to test the Convert Experiences App where we can test all site's forms or selected parameters with all available attacks. This tool run as a sidebar.



Results are reported to a separate Firefox tab when the test run is complete. Totally, the tool found **0 failures out of 818720 tests** in the main page of Convert Experiences app (https://app.convertexperiments.com). You may see the detailed results in the following table:

Page	Results in brief	Detailed Documentation
https://app.convertexperiments.com	Failures: 0 Warnings: 0 Passes: 818720 tests	
https://app.convertexperiments.com/#/dashboard	Failures: 0 Warnings: 0 Passes: 190060 tests	

https://app.convertexperiments.com/#/projects/1002839/settings	Failures: 0 Warnings: 0 Passes: 540940 tests	
We can select other pages as well to test with this add-on		

A2: Broken Authentication and Session Management – HackBar (v1.6.1)

Session management vulnerabilities occur when developers fail to protect their users' sensitive information such as user names, passwords, and session tokens. *Broken authentication* vulnerabilities occur when developers fail to use authentication methods that have been adequately tested and rely on their own, often flawed, method for authenticating users. To prevent these types of vulnerabilities from occurring in your application, developers should first ensure that SSL is used for all authenticated parts of the application. In addition, verify that all credentials are stored in a hashed form.

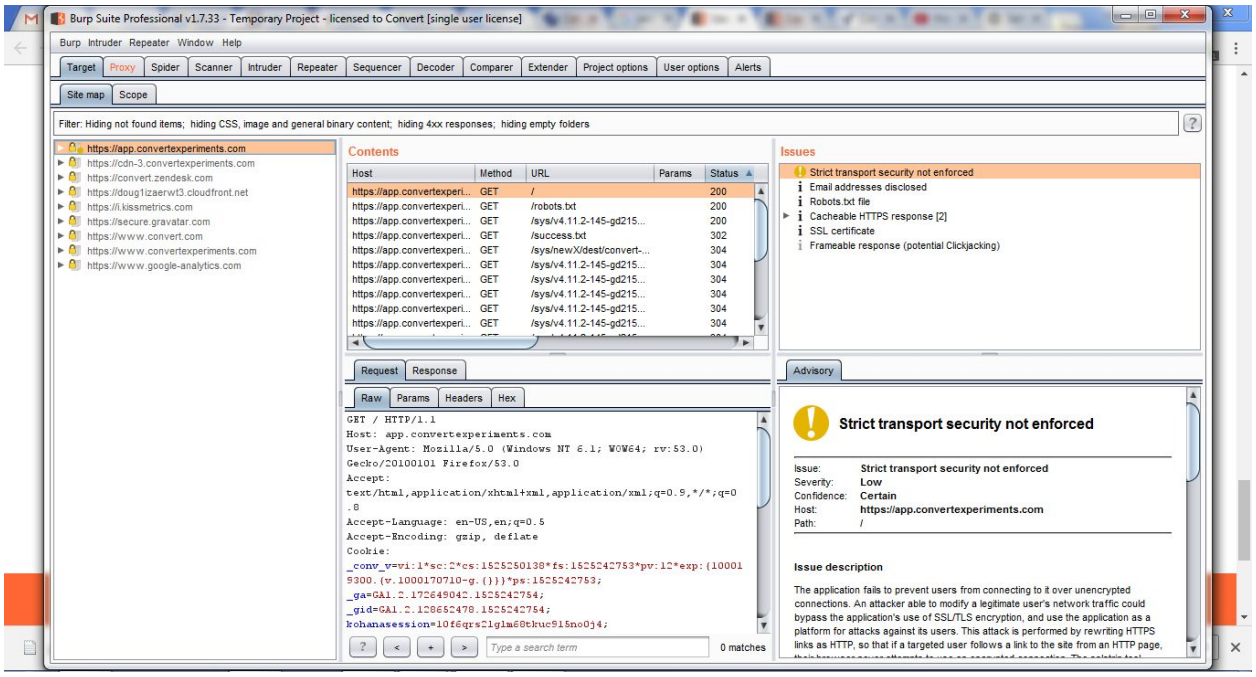
Here, we could use the Firefox add-on **HackBar**. Once installed, hit F9 to show HackBar, then select Encryption, followed by MD5 Menu or SHA1, then Send to, which will pull results, if available.

In this specific test case, **no risky results** were returned.

A3: Sensitive Data Exposure

Sensitive Data Exposure vulnerabilities can occur when a web application does not adequately protect sensitive information from being disclosed to attackers. This can include information such as credit card data, medical history, session tokens, or other authentication credentials.

For this security risk, we could use **Burp Suite Professional**. To launch Burp, read these instructions.



A4: XML External Entities (XEE)

XXE Injection can occur when XML parsers are overly permissive in their configurations and allow for the processing of external XML entities. These external entities can reference files on the local file system or even share drives. The successful exploitation of XXE can result in the ability to compromise sensitive configuration files, the mapping of internal networks, and even the sending of email.

There are two primary types of XML injection:

- XXE attacks which include output within the server's response.
- Blind XXE - Attacks which process an entity, but do not include the results within output. We must instead entice the application server to 'send us' the response.

Using **Burp Suite Professional** we did not detect any XEE injection vulnerabilities.

A5: Broken Access Control

https://support.portswigger.net/customer/portal/articles/1965720-using-burp-to-test-for-missing-function-level-access-control

A6: Security Misconfiguration

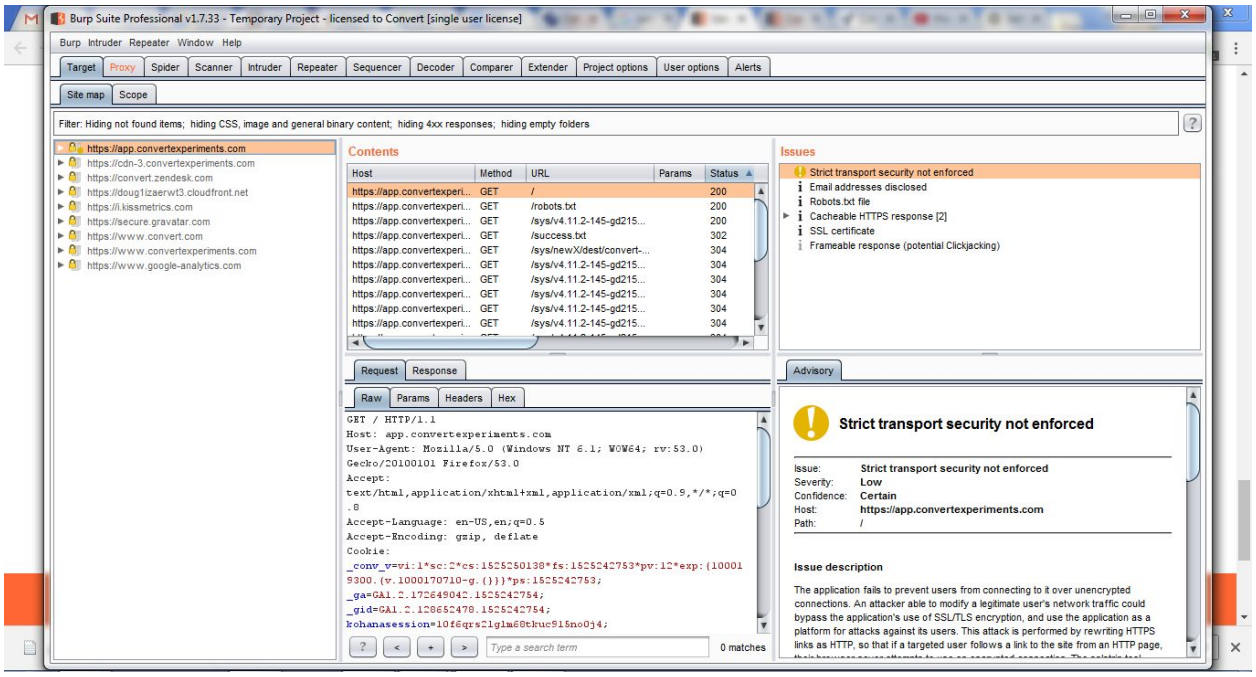
Application misconfiguration attacks exploit configuration weaknesses found in web applications.

Security misconfiguration can happen at any level of an application stack, including the platform, web server, application server, database, and framework.

Many applications come with unnecessary and unsafe features, such as debug and QA features, enabled by default. These features may provide a means for a hacker to bypass authentication methods and gain access to sensitive information, perhaps with elevated privileges.

Likewise, default installations may include well-known usernames and passwords, hard-coded backdoor accounts, special access mechanisms, and incorrect permissions set for files accessible through web servers.

In this example we will demonstrate how to use **Burp Suite Professional** (Spider and/or Site Map) to check for directory listings. Instructions are available here.



A7: Cross-Site Scripting (XSS) – ZAP (v2.7.0)

The **Zed Attack Proxy (ZAP)**, also an OWASP project, is “an easy to use integrated penetration testing tool for finding vulnerabilities in Web applications.” It is designed to be used by people with a wide range of security experience and as such is ideal for developers and functional testers who are new to penetration testing. ZAP provides automated scanners as well as a set of tools that allow you to find security vulnerabilities manually. While ZAP is capable of a variety of web application security checks, we’ll use it here to test for cross-site scripting (XSS).

ZAP was defined as one of our proxies, fired up after installation and the browser was set to run traffic through it. ZAP was pointed at Convert Experiences App. After viewing all the pages, the ZAP UI was populated with visited pages and results were taken.

ZAP tool did not find any XSS risks.

A8: Insecure Deserialization

https://blog.detectify.com/2018/03/21/owasp-top-10-insecure-deserialization/

A9: Using Components with known vulnerabilities

https://hdivsecurity.com/owasp-using-components-with-known-vulnerabilities

A10: Insufficient Logging & Monitoring

https://blog.detectify.com/2018/04/06/owasp-top-10-insufficient-logging-monitoring/
https://hdivsecurity.com/owasp-insufficient-logging-and-monitoring