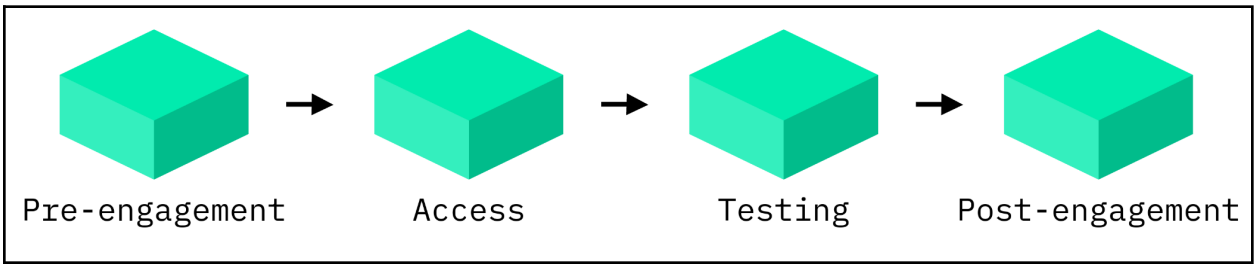
Forces Unseen

www.forcesunseen.com
contact@forcesunseen.com
+1 (312) 767-4029

Application Security Assessment Methodology



Pre-engagement

In this phase, Forces Unseen establishes the goals and scope of the engagement by gathering information about the target application and client objectives.

Goals

Understanding the purpose and business function of the application is essential to provide a comprehensive review. Functionality, type of data, and associated risks should be enumerated at this stage. This is also the best time to share any pre-existing security concerns regarding the application, recently released functionality, or particular classes of vulnerabilities.

Scoping

Understanding the scope will ensure that Forces Unseen is able to assess the application adequately. This includes reviewing items such as:

- The size of the codebase
- Software, frameworks, dependencies, and technologies in use
- API documentation/routes listings

Access

As a prerequisite to testing, Forces Unseen requires access to the application. This will take place prior to the target start date of the engagement.

Some items the team may need include:

- Application URLs
- Credentials for multiple accounts/organizations with varying privileges
- Application source code
- Documentation and diagrams
- Slack, email, or other communication channels

Testing

The majority of the apportioned time is spent in this phase. This is when tool-assisted manual security testing occurs. Forces Unseen will communicate with the client during this time in regards to access obstacles, critical vulnerabilities, and developer insight.

Tooling

Static Analysis

Forces Unseen uses a combination of open-source and in-house tools to scout for high-impact bugs in the codebase. If a unique vulnerability is discovered, a rule will be created to check for that class of vulnerability across the entire codebase.

Validation of Automated Scan Results

Forces Unseen performs automated application scanning using Burp Suite Professional and other third-party and in-house tools. Forces Unseen validates these results to ensure only bona fide vulnerabilities are included in the report.

Manual Activities

Manual assessment of the application is crucial for discovering impactful security vulnerabilities. This involves understanding the application logic and its business context. The adversarial “tire-kicking” often leads to the identification of complex and critical vulnerabilities that would otherwise go undiscovered. As all vulnerabilities are unique, custom proof-of-concept code is commonly created to exploit vulnerabilities. The following summary highlights key areas which are brought into focus during an engagement:

Authentication & Session Management

The authentication flow is one of the most important parts of an application. The application must be able to accurately determine who is logged in and ensure that the sessions are handled securely. This applies to plug and play authentication as well as custom implementations.

- SSO (OAuth, SAML, etc.)
- Token Handling (JWT, custom, etc.)
- User Enumeration
- Brute Forcing
- Account Takeover

Authorization & Business Logic

Asserting privileges, delegating access, and assuming roles are all common functionalities within applications. Functionalities of these types can give rise to authorization and logic bugs.

- Insecure Direct Object Reference
- Privilege Escalation
- Race Conditions
- Administrative Interfaces

Data Handling

Data encryption requirements and standards vary by category and context. Regulatory requirements often dictate minimum standards and practices. While data handling is often straightforward, nuances in implementation can result in unexpected or exploitable behavior.

- Persistent Data Encryption
- Transport Security
- Secrets Management
- Tenancy and Partitioning

Input Handling

All applications handle input data in one form or another. Should that data be confused as code by the application, the effects can be devastating. Validation of encoding and data-sanitization implementation ensures that applications are not vulnerable to injection-class attacks.

- SQL Injection
- Command Injection
- Server-Side Request Forgery
- XML External Entity Injection
- Server-Side Template Injection

Browser Security

Browsers are tremendously complex and practically an operating system in and of themselves. Browsers have also evolved significantly over time, gaining more and more opt-in security functionality. Due to the complex nature of browsers, web applications, and their interactions, applications must be tested for a variety of vulnerabilities and security feature implementations.

- Cross-Site Scripting
- Cross-Site Request Forgery
- Content Security Policy
- Strict Transport Security

Post-engagement

Once the assessment is complete, the team delivers a report that will provide the information necessary to reproduce and remediate the identified vulnerabilities. This will be provided as a PDF document along with any accompanying materials referenced in the report.

We then schedule a readout of the report to review and answer any questions. Remediation testing can be scheduled once the vulnerabilities have been remediated.