OWASP

The Open Web Application Security Project

Application Security Verification Standard (2014)

Web Application Standard

Preface

Our biggest goal with this version of the standard was to increase adoption.

One of the major challenges of a standard such as this is that it needs to satisfy two distinct, and very different, targets: individuals who are involved in organizing or executing a software security program within their organization, and software security professionals who conduct verification of applications. While both targets seek an industry-accepted standard for verification of applications, they operate under different constraints. For example, one of the most widely voiced criticisms of ASVS 2009 standard was that it specified automated assessments as one of the levels (or sub-levels). Many large organizations see automated assessments as a point of entry into the verification hierarchy, and thus a fully automated level is a convenient concept for them. Information security professionals, however, know that the depth and breadth of such a review will depend on what technology is used to perform the scan, thus leaving too much room for interpretation of the standard. ASVS 2014 introduces a Cursory Level 0 to allow for the flexibility needed to overcome this challenge.

On a similar note, one of the main goals for this version of the standard was to focus on the "what" and not the "how". Whereas the previous version of the standard talked about dynamic scanning, static analysis, Threat Modeling, and design reviews, you will notice that such terms do not appear in this version of the standard. Instead, we essentially define security requirements that must be verified for an application to achieve a certain level. How those requirements are verified is left up to the verifier.

Another major challenge that we overcame is to clearly separate requirements from design from scope. The previous version of the standard did not clearly distinguish between these concepts, leaving room for confusion. In this version, Level 3 is where design considerations are introduced and clearly separated from detailed verification requirements. Furthermore, we have now separated out the concept of scope completely – the new (+) notation allows for a verifier to optionally include third party components and frameworks in their review.

We expect that there will most likely never be 100% agreement on this standard. Risk analysis is always subjective to some extent, which creates a challenge when attempting to generalize in a one size fits all standard. However, we hope that the latest updates made in this version are a step in the right direction, and respectfully enhance the concepts introduced in this important industry standard.

Acknowledgements

Version 2014

Project Leads: Sahba Kazerooni (Security Compass, http://www.securitycompass.com),

Daniel Cuthbert (SensePost, http://www.sensepost.com/)

Lead Authors: Andrew van der Stock, Sahba Kazerooni, Daniel Cuthbert, Krishna Raja

Reviewers and contributors: Jerome Athias, Boy Baukema, Archangel Cuison, Sebastien Deleersnyder, Antonio Fontes, Evan Gaustad, Safuat Hamdy, Ari Kesäniemi, Scott Luc, Jim Manico, Mait Peekma, Pekka Sillanpää, Jeff Sergeant, Etienne Stalmans, Colin Watson, Dr Emin Tatli.

Additionally, thanks are given to the application security verification community and others interested in trusted web computing for their enthusiastic advice and assistance throughout this effort.

Version 2009

As ASVS 2014 includes many of the original requirements, the following contributors are recognized for their efforts during the original Application Security Verification Standard effort: Mike Boberski, Jeff Williams, Dave Wichers, Pierre Parrend (OWASP Summer of Code), Andrew van der Stock, Nam Nguyen, John Martin, Gaurang Shah, Theodore Winograd, Stan Wisseman, Barry Boyd, Steve Coyle, Paul Douthit, Ken Huang, Dave Hausladen, Mandeep Khera Scott Matsumoto, John Steven, Stephen de Vries, Dan Cornell, Shouvik Bardhan, Dr. Sarbari Gupta, Eoin Keary, Richard Campbell, Matt Presson, Jeff LoSapio, Liz Fong, George Lawless, Dave van Stein, Terrie Diaz, Ketan Dilipkumar Vyas, Bedirhan Urgun, Dr. Thomas Braun, Colin Watson, Jeremiah Grossman.

Copyright and License

Table of Contents

Introduction

The primary aim of the OWASP Application Security Verification Standard (ASVS) is to normalize the range in the coverage and level of rigor available in the market when it comes to performing web application security verification.

The Open Web Application Security Project (OWASP) is an open community dedicated to enabling organizations to develop, purchase, and maintain applications that can be trusted. All of the OWASP tools, documents, forums, and chapters are free and open to anyone interested in improving application security. We advocate approaching application security as a people, process, and technology problem, because the most effective approaches to application security include improvements in all of these areas. We can be found at www.owasp.org.

OWASP is a new kind of organization. Our freedom from commercial pressures allows us to provide unbiased, practical, cost-effective information about application security. OWASP is not affiliated with any technology company, although we support the informed use of commercial security technology. Similar to many open-source software projects, OWASP produces many types of materials in a collaborative, open way. The OWASP Foundation is a not-for-profit entity that ensures the project's long-term success.

The ASVS standard provides a basis for verifying application technical security controls, as well as any technical security controls in the environment that are relied on to protect against vulnerabilities such as Cross-Site Scripting (XSS) and SQL injection.[1] This standard can be used to establish a level of confidence in the security of Web applications.

[1] For more information about common Web application vulnerabilities, see the OWASP Top Ten (OWASP, 2013).

How to Use This Standard

The ASVS standard can be used by both consumers and service or tool providers.

ASVS has two main goals, as depicted in the figure below: to help organization's develop and maintain secure applications; and to allow security service/tools providers and consumers to align their requirements and offerings.

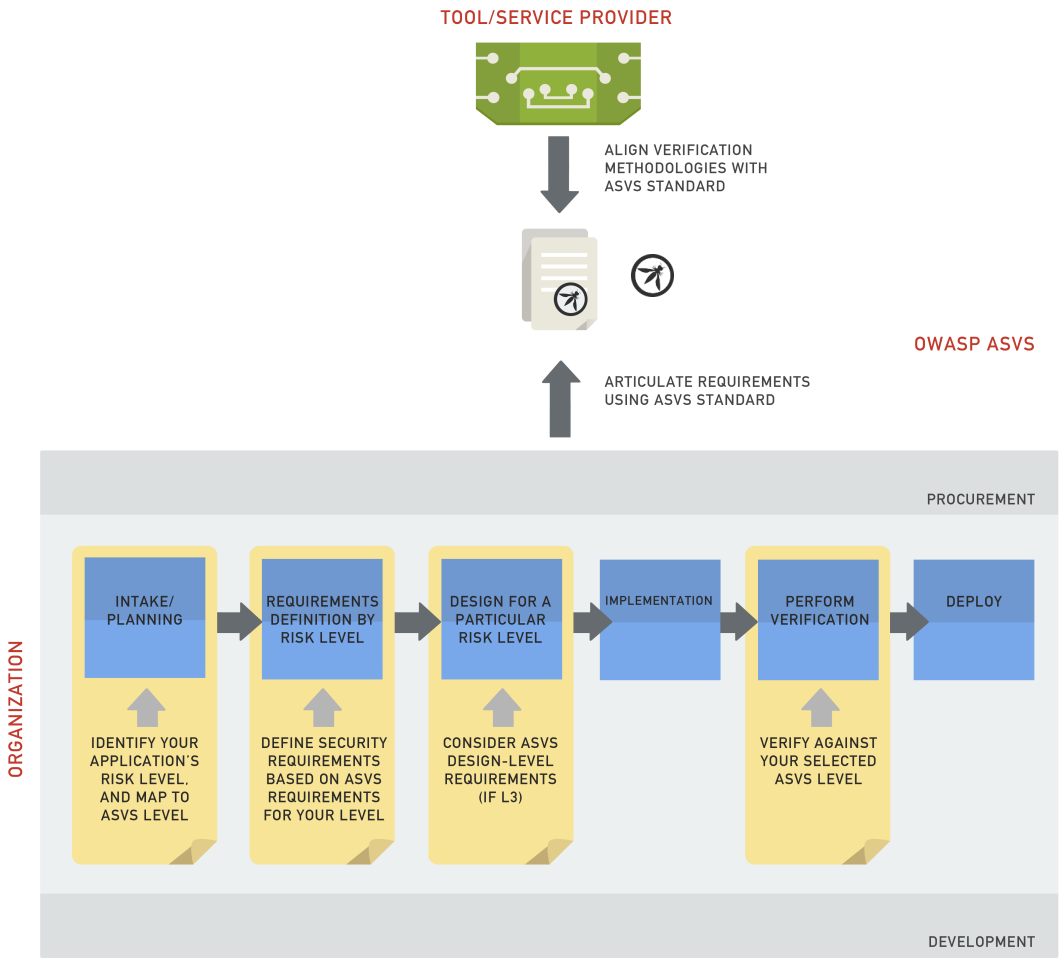

Figure 1 – Uses of ASVS for organizations and tool/service providers

The example scenarios below further demonstrate the common use cases of ASVS using a fictional organization (ACME Bank) and a fictional security services firm (Hack All the Things).

Use Case 1: Certification of Applications

ACME Bank has developed a new Internet Banking portal, which is due to be deployed into their production environment. The application has followed the bank's SDLC process and should be in a secure state. The Internal security team at ACME Bank has been tasked to ensure that once deployed into the production environment, it does not pose a risk to other applications, due to it being hosted on a shared platform and database. After an internal threat modeling exercise was performed, it was agreed that the application had a high-risk associated with it and the data stored within it.

The team makes use of a well-known web application scanning tool and start the process of mapping out the application in preparation for the automated scanning phase. Once complete, the automated scanning tool is started and left to complete. Once the report has been generated, the security analyst tests for false positives (such as SQL injection, or XSS) and amends the report as necessary. Any findings discovered are reported back to the system owners and development team, in order to be rectified. Once this has been completed, the re-test of the application is performed to ensure they have been resolved in a suitable manner.

In this example, using the ASVS could allow the internal team to test for common application flaws as well as verify that it had been developed in accordance to the bank's security standard.

Use Case 2: Alignment of testing methodology

Hack All the Things (HATT) is a penetration-testing consultancy, whose main area of expertise is performing application security assessments for clients at an infrastructure and application level. They have decided to align their internal testing methodology with that of the OWASP ASVS to offer their clients peace of mind when performing assessments.

In order to achieve this, all staff are required to manually test the application in question using the detailed verification requirements, as outlined by the ASVS document.

In this instance, adopting the ASVS allows HATT to offer a series of application assessments based on the four ASVS levels, and at the same time, allowing clients to understand what has been assessed.

Use Case 3: Selection of external supplier

ACME Bank has finally completed all development on their new Internet Banking portal and the banking regulators require them to have an external consultancy perform an assessment of the application to ensure it meets the regulatory requirements with regards to security.

ACME Bank has chosen a supplier from their list of preferred suppliers and asked HATT to perform an assessment. ACME Bank supplied the consultancy with all the source code and documentation and scheduled the assessment. The external test was conducted in a phased approach, with a fully-automated static analysis code review performed on the source code alongside a manual application security assessment. In addition, business logic was tested to ensure that the application performed as expected, as outlined in the functional specification documentation supplied. Once the assessment was complete, a report was created and delivered to ACME Bank staff.

By both parties adopting the ASVS during this process, the suitable level was chosen and tested for. As a result, both ACME bank and HATT were in sync with what had to be achieved and what the required outcome was.

Application Security Verification Levels

The ASVS defines four levels of verification, with each level increasing in depth as the verification moves up the levels.

The depth is defined in each level by a set of security verification requirements that must be addressed (these are included in the requirements tables towards the end of this document). It is a verifier's responsibility to determine if a target of verification (TOV) meets all of the requirements at the level targeted by a review. If the application meets all of the requirements for that level, then it can be considered an OWASP ASVS Level N application, where N is the verification level that application complied with. If the application does not meet all the requirements for a particular level, but does meet all the requirements for a lower level of this standard, then it can be considered to have passed the lower level of verification.

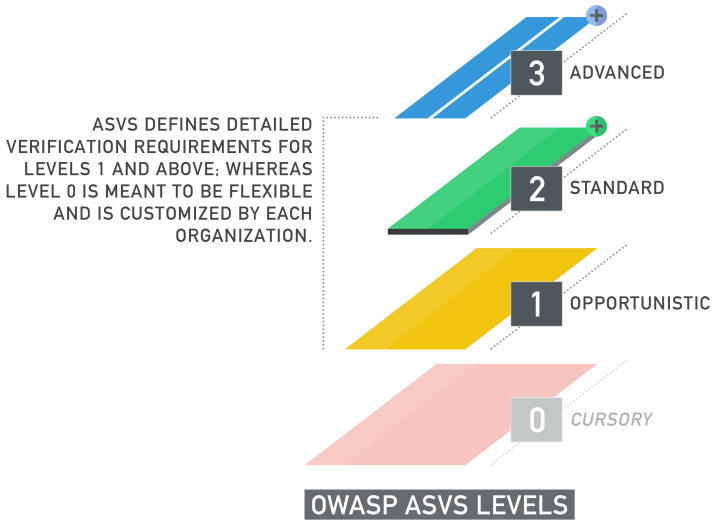

Figure 2 – OWASP ASVS Levels

The breadth of the verification is defined by what parts of the application are reviewed for each security requirement. For example, the scope of the review may go beyond the application's custom-built code and include external components. Achieving a verification level under such scrutiny can be represented by annotating a "+" symbol to the verification level.

Level 0: Cursory

Level 0 (or Cursory) is an optional certification, indicating that the application has passed some type of verification.

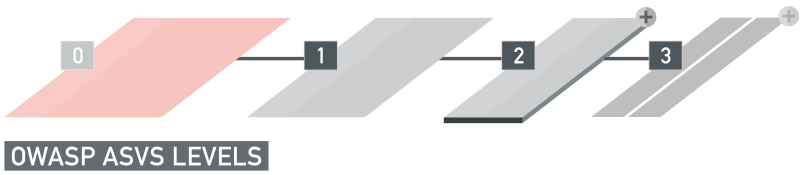

Figure3 – OWASP ASVS Level 0

Level 0 is designed to be a flexible point of entry into the verification hierarchy; it indicates that some type of review has been done on the application. The detailed verification requirements are not provided by ASVS. Instead, organizations can define their own minimum criteria (such as automated runtime scan, or strong authentication mechanism).

This level is most appropriate for organizations that have a large number of applications, and where a low cost point of entry may be required. One organization may use Level 0 to require a cursory automated scan of all of their external facing applications using the organization's commercial tool of choice; whereas another organization may define L0 requirements using data from a recent breach.

Unlike the other ASVS levels, Level 0 is not a prerequisite for other levels - an application can jump straight to Level 1 without achieving Level 0 certification (if L0 is not defined by the organization).

When defining Level 0 requirements, it is advised that each requirement be documented in a similar manner to the Detailed Verification Requirements in this document – clear, distinct, realistic, and verifiable.

Overview of Verification Requirements

L0 ASVS does not define the detailed verification requirements for this level. Application is assessed according to requirements as defined by the organization.

Level 1: Opportunistic

An application achieves Level 1 (or Opportunistic) certification if it adequately defends against application security vulnerabilities that are easy to discover.

Figure 4 – OWASP ASVS Level 1

The specific set of vulnerabilities against which Level 1 verification is measured is detailed in the Detailed Verification Requirements, but typically includes vulnerabilities that a verifier can identify with minimal-to-low effort. As such, this level cannot be considered a thorough inspection or verification of the application, but more of a quick inspection.

Level 1 is typically appropriate for applications where some confidence in the correct use of security controls is required, or to provide a quick sweep of a fleet of enterprise applications, to assist in developing a roadmap for more thorough inspections at a later date.

Threats to the application will most likely be from attackers who are using simple techniques to identify easy-to-find and easy-to-exploit vulnerabilities. This is in contrast to a determined attacker who will spend focused energy to specifically target the application.

Overview of Verification Requirements

L1 Application is assessed according to the Level 1 requirements in the "Detailed Verification Requirements" section.

Level 2: Standard

An application achieves Level 2 (or Standard) verification if it also adequately defends against prevalent application security vulnerabilities whose existence poses moderate-to-serious risk.

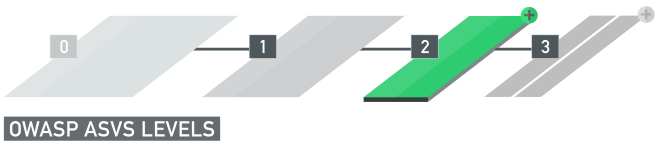

Figure 5 – OWASP ASVS Level 2

The specific set of vulnerabilities against which Level 2 verification is measured is detailed in the Detailed Verification Requirements, but would include OWASP Top 10 vulnerabilities and business logic vulnerabilities.

Level 2 ensures that evaluated security controls are in place, effective, and used as needed within the application to enforce application-specific policies.

Level 2 represents an industry standard for which the majority of an organization's sensitive applications would strive. Level 2 is typically appropriate for applications that handle significant business-to-business transactions, including those that process healthcare information, implement business-critical or sensitive functions, or process other sensitive assets.

Threats to security will typically be opportunists and possibly determined attackers (skilled and motivated attackers focusing on specific targets using purpose-built scanning tools as well as manual testing techniques).

Overview of Verification Requirements

L2 Application is assessed according to the Level 2 requirements in the "Detailed Verification Requirements" section.

Level 3: Advanced

An application achieves Level 3 (or Advanced) certification if it also adequately defends against all advanced application security vulnerabilities, and also demonstrates principles of good security design.

Figure 6 – OWASP ASVS Level 3

The specific set of vulnerabilities against which Level 3 verification is measured is detailed in the Detailed Verification Requirements, but would include more difficult to exploit vulnerabilities, which would most likely be exploited by determined attackers.

Level 3 is the only ASVS level which also requires an inspection of the application's design. In addition, the following requirements were added:

- Any major security controls which have a cross-cutting impact (such as input validation or authorization) should be implemented in a centralized manner.
- Security controls that perform validation should make decisions using a whitelist ("positive") approach.
- Input validation should not be relied on as the only defense against injection and scripting vulnerabilities. Rather, input validation should always be the second line of defense, with parameterization and output encoding being the primaries, respectively.

Level 3 verification is typically appropriate for critical applications that protect life and safety, critical infrastructure, or defense functions or have the potential of facilitating substantial damage to the organization. Level 3 may also be appropriate for applications that process sensitive assets.

Threats to security will be from determined attackers (skilled and motivated attackers focusing on specific

targets using tools including purpose-built scanning tools).

Overview of Verification Requirements

L3.1 Application is assessed according to the Level 3 requirements in the "Detailed Verification Requirements" section.

L3.2 Application is verified that implementation of all security controls adhere to the following best practices:

Security controls are centralized within the application.

Security controls that perform validation make decisions using a whitelist ("positive") approach.

Data validation controls are complemented by output encoding routines.

All untrusted data that is output to SQL interpreters use parameterized interfaces, prepared statements, or are escaped properly.

Scope of Verification

The scope of the verification is separate from the requirements for achieving a level.

Be default, ASVS assumes that the scope of the verification includes all code that was developed or modified in order to create the application or release. However, one may decide to include as part of verification the code for all third-party frameworks, libraries, and service security functionality that is invoked by or supports the security of the application. Achieving a verification level under such scrutiny can be represented by annotating a "+" symbol to the verification level. For example, an application may be labelled as ASVS L3+ certified.

Including third party components is optional and is not required to achieve to any ASVS level. Such level of scrutiny may be suitable for highly sensitive or mission critical applications. As such, (+) certification will in most cases be associated with Level 3.

When third party components are included in the verification, it is not required that all detailed verification requirements be applied to third party components. In fact, most detailed verification requirements will not be applicable to third party components and can thus be checked against the base code only. Detailed verification requirements must be verified against the application's base code, and they are verified against third party components if applicable. Only then can an application achieve the (+) certification for that level.

Detailed Verification Requirements

This section of the OWASP Application Security Verification Standard (ASVS) defines detailed verification requirements that were derived from the high-level requirements for each of the verification levels defined in this standard. Each section below defines a set of detailed verification requirements grouped into related areas.

The ASVS defines the following security requirements areas. The numbering scheme has been kept consistent with the previous version of ASVS to help with individuals wishing to transition from one to the other.

V2. Authentication

V3. Session Management

V4. Access Control

V5. Malicious Input Handling

V7. Cryptography at Rest

V8. Error Handling and Logging

V9. Data Protection

V10. Communications

V11. HTTP

V13. Malicious Controls

V15. Business Logic

V16. File and Resource

V17. Mobile

V2: Authentication Verification Requirements

The table below defines the corresponding verification requirements that apply for each of the verification levels. Verification requirements for Level 0 are not defined by this standard.

	AUTHENTICATION VERIFICATION REQUIREMENT	LEVELS		
		1	2	3
V2.1	Verify all pages and resources require authentication except those specifically intended to be public (Principle of complete mediation).	✔	✔	✔
V2.2	Verify all password fields do not echo the user's password when it is entered.	✔	✔	✔
V2.4	Verify all authentication controls are enforced on the server side.	✔	✔	✔
V2.5	Verify all authentication controls (including libraries that call external authentication services) have a centralized implementation.			✔
V2.6	Verify all authentication controls fail securely to ensure attackers cannot log in.	✔	✔	✔
V2.7	Verify password entry fields allow or encourage the use of passphrases, and do not prevent long passphrases or highly complex passwords being entered, and provide a sufficient minimum strength to protect against the use of commonly chosen passwords.		✔	✔
V2.8	Verify all account identity authentication functions (such as registration, update profile, forgot username, forgot password, disabled / lost token, help desk or IVR) that might regain access to the account are at least as resistant to attack as the primary authentication mechanism.		✔	✔
V2.9	Verify users can safely change their credentials using a mechanism that is at least as resistant to attack as the primary authentication mechanism.		✔	✔
V2.12	Verify that all authentication decisions are logged. This should include requests with missing required information, needed for security investigations.		✔	✔
V2.13	Verify that account passwords are salted using a salt that is unique to that account (e.g., internal user ID, account creation) and use bcrypt, scrypt or PBKDF2 before storing the password.		✔	✔

	AUTHENTICATION VERIFICATION REQUIREMENT	LEVELS		
		1	2	3
V2.16	Verify that credentials, and all other identity information handled by the application(s), do not traverse unencrypted or weakly encrypted links.	✔	✔	✔
V2.17	Verify that the forgotten password function and other recovery paths do not reveal the current password and that the new password is not sent in clear text to the user.	✔	✔	✔
V2.18	Verify that username enumeration is not possible via login, password reset, or forgot account functionality.	✔	✔	✔
V2.19	Verify there are no default passwords in use for the application framework or any components used by the application (such as “admin/password”).	✔	✔	✔
V2.20	Verify that a resource governor is in place to protect against vertical (a single account tested against all possible passwords) and horizontal brute forcing (all accounts tested with the same password e.g. “Password1”). A correct credential entry should incur no delay. Both these governor mechanisms should be active simultaneously to protect against diagonal and distributed attacks.		✔	✔
V2.21	Verify that all authentication credentials for accessing services external to the application are encrypted and stored in a protected location (not in source code).		✔	✔
V2.22	Verify that forgot password and other recovery paths send a link including a time-limited activation token rather than the password itself. Additional authentication based on soft-tokens (e.g. SMS token, native mobile applications, etc.) can be required as well before the link is sent over.		✔	✔
V2.23	Verify that forgot password functionality does not lock or otherwise disable the account until after the user has successfully changed their password. This is to prevent valid users from being locked out.		✔	✔
V2.24	Verify that there are no shared knowledge questions/answers (so called "secret" questions and answers).		✔	✔
V2.25	Verify that the system can be configured to disallow the use of a configurable number of previous passwords.		✔	✔
V2.26	Verify re-authentication, step up or adaptive authentication, SMS or other two factor authentication, or transaction signing is required before any application-specific sensitive operations are permitted as per the risk profile of the application.			✔

Table 1 - OWASP ASVS Authentication Requirements (V2)

V3: Session Management Verification Requirements

The table below defines the corresponding verification requirements that apply for each of the verification levels. Verification requirements for Level 0 are not defined by this standard.

	SESSION MANAGEMENT VERIFICATION REQUIREMENT	LEVELS		
		1	2	3
V3.1	Verify that the framework's default session management control implementation is used by the application.	✔	✔	✔
V3.2	Verify that sessions are invalidated when the user logs out.	✔	✔	✔
V3.3	Verify that sessions timeout after a specified period of inactivity.	✔	✔	✔
V3.4	Verify that sessions timeout after an administratively-configurable maximum time period regardless of activity (an absolute timeout).		✔	✔
V3.5	Verify that all pages that require authentication to access them have logout links.	✔	✔	✔
V3.6	Verify that the session id is never disclosed other than in cookie headers; particularly in URLs, error messages, or logs. This includes verifying that the application does not support URL rewriting of session cookies.	✔	✔	✔
V3.7	Verify that the session id is changed on login to prevent session fixation.		✔	✔
V3.8	Verify that the session id is changed upon re-authentication.		✔	✔
V3.10	Verify that only session ids generated by the application framework are recognized as valid by the application.		✔	✔
V3.11	Verify that authenticated session tokens are sufficiently long and random to withstand session guessing attacks.		✔	✔
V3.12	Verify that authenticated session tokens using cookies have their path set to an appropriately restrictive value for that site. The domain cookie attribute		✔	✔

	SESSION MANAGEMENT VERIFICATION REQUIREMENT	LEVELS		
		1	2	3
	restriction should not be set unless for a business requirement, such as single sign on.		✓	✓
V3.14	Verify that authenticated session tokens using cookies sent via HTTP, are protected by the use of "HttpOnly".	✓	✓	✓
V3.15	Verify that authenticated session tokens using cookies are protected with the "secure" attribute and a strict transport security header (such as Strict-Transport-Security: max-age=60000; includeSubDomains) are present.	✓	✓	✓
V3.16	Verify that the application does not permit duplicate concurrent user sessions, originating from different machines.		✓	✓

Table 2 - OWASP ASVS Session Management Requirements (V3)

V4: Access Control Verification Requirements

The table below defines the corresponding verification requirements that apply for each of the verification levels. Verification requirements for Level 0 are not defined by this standard.

	ACCESS CONTROL VERIFICATION REQUIREMENT	LEVELS		
		1	2	3
V4.1	Verify that users can only access secured functions or services for which they possess specific authorization.	✔	✔	✔
V4.2	Verify that users can only access secured URLs for which they possess specific authorization.	✔	✔	✔
V4.3	Verify that users can only access secured data files for which they possess specific authorization.	✔	✔	✔
V4.4	Verify that direct object references are protected, such that only authorized objects or data are accessible to each user (for example, protect against direct object reference tampering).	✔	✔	✔
V4.5	Verify that directory browsing is disabled unless deliberately desired.	✔	✔	✔
V4.8	Verify that access controls fail securely.	✔	✔	✔
V4.9	Verify that the same access control rules implied by the presentation layer are enforced on the server side for that user role, such that controls and parameters cannot be re-enabled or re-added from higher privilege users.		✔	✔
V4.10	Verify that all user and data attributes and policy information used by access controls cannot be manipulated by end users unless specifically authorized.		✔	✔
V4.11	Verify that all access controls are enforced on the server side.	✔	✔	✔
V4.12	Verify that there is a centralized mechanism (including libraries that call external authorization services) for protecting access to each type of protected resource.			✔

	ACCESS CONTROL VERIFICATION REQUIREMENT	LEVELS		
		1	2	3
V4.14	Verify that all access control decisions are be logged and all failed decisions are logged.		✓	✓
V4.16	Verify that the application or framework generates strong random anti-CSRF tokens unique to the user as part of all high value transactions or accessing sensitive data, and that the application verifies the presence of this token with the proper value for the current user when processing these requests.	✓	✓	✓
V4.17	Aggregate access control protection – verify the system can protect against aggregate or continuous access of secured functions, resources, or data. For example, possibly by the use of a resource governor to limit the number of edits per hour or to prevent the entire database from being scraped by an individual user.		✓	✓

Table 3 - OWASP ASVS Access Control Requirements (V4)

V5: Malicious Input Handling Verification Requirements

The table below defines the corresponding verification requirements that apply for each of the verification levels. Verification requirements for Level 0 are not defined by this standard.

	INPUT VALIDATION VERIFICATION REQUIREMENT	LEVELS		
		1	2	3
V5.1	Verify that the runtime environment is not susceptible to buffer overflows, or that security controls prevent buffer overflows.	✓	✓	✓
V5.3	Verify that all input validation failures result in input rejection.	✓	✓	✓
V5.4	Verify that a character set, such as UTF-8, is specified for all sources of input.		✓	✓
V5.5	Verify that all input validation or encoding routines are performed and enforced on the server side.	✓	✓	✓
V5.6	Verify that a single input validation control is used by the application for each type of data that is accepted.			✓
V5.7	Verify that all input validation failures are logged.			✓
V5.8	Verify that all input data is canonicalized for all downstream decoders or interpreters prior to validation.		✓	✓
V5.10	Verify that the runtime environment is not susceptible to SQL Injection, or that security controls prevent SQL Injection.	✓	✓	✓
V5.11	Verify that the runtime environment is not susceptible to LDAP Injection, or that security controls prevent LDAP Injection.	✓	✓	✓
V5.12	Verify that the runtime environment is not susceptible to OS Command Injection, or that security controls prevent OS Command Injection.	✓	✓	✓
V5.13	Verify that the runtime environment is not susceptible to XML External Entity attacks or that security controls prevents XML External Entity attacks.	✓	✓	✓

	INPUT VALIDATION VERIFICATION REQUIREMENT	LEVELS		
		1	2	3
V5.14	Verify that the runtime environment is not susceptible to XML Injections or that security controls prevents XML Injections.	✓	✓	✓
V5.16	Verify that all untrusted data that are output to HTML (including HTML elements, HTML attributes, JavaScript data values, CSS blocks, and URI attributes) are properly escaped for the applicable context.	✓	✓	✓
V5.17	If the application framework allows automatic mass parameter assignment (also called automatic variable binding) from the inbound request to a model, verify that security sensitive fields such as “accountBalance”, “role” or “password” are protected from malicious automatic binding.		✓	✓
V5.18	Verify that the application has defenses against HTTP parameter pollution attacks, particularly if the application framework makes no distinction about the source of request parameters (GET, POST, cookies, headers, environment, etc.)		✓	✓
V5.19	Verify that for each type of output encoding/escaping performed by the application, there is a single security control for that type of output for the intended destination.			✓

Table 4 - OWASP ASVS Malicious Input Handling Requirements (V5)

V7: Cryptography at Rest Verification Requirements

The table below defines the corresponding verification requirements that apply for each of the verification levels. Verification requirements for Level 0 are not defined by this standard.

	CRYPTOGRAPHY AT REST VERIFICATION REQUIREMENT	LEVELS		
		1	2	3
V7.1	Verify that all cryptographic functions used to protect secrets from the application user are implemented server side.		✔	✔
V7.2	Verify that all cryptographic modules fail securely.		✔	✔
V7.3	Verify that access to any master secret(s) is protected from unauthorized access (A master secret is an application credential stored as plaintext on disk that is used to protect access to security configuration information).		✔	✔
V7.6	Verify that all random numbers, random file names, random GUIDs, and random strings are generated using the cryptographic module's approved random number generator when these random values are intended to be unguessable by an attacker.		✔	✔
V7.7	Verify that cryptographic modules used by the application have been validated against FIPS 140-2 or an equivalent standard.			✔
V7.8	Verify that cryptographic modules operate in their approved mode according to their published security policies.			✔
V7.9	Verify that there is an explicit policy for how cryptographic keys are managed (e.g., generated, distributed, revoked, expired). Verify that this policy is properly enforced.		✔	✔

Table 5 - OWASP ASVS Cryptography at Rest Requirements (V7)

V8: Error Handling and Logging Verification Requirements

The table below defines the corresponding verification requirements that apply for each of the verification levels. Verification requirements for Level 0 are not defined by this standard.

	ERROR HANDLING AND LOGGING VERIFICATION REQUIREMENT	LEVELS		
		1	2	3
V8.1	Verify that the application does not output error messages or stack traces containing sensitive data that could assist an attacker, including session id and personal information.	✔	✔	✔
V8.2	Verify that all error handling is performed on trusted devices		✔	✔
V8.3	Verify that all logging controls are implemented on the server.		✔	✔
V8.4	Verify that error handling logic in security controls denies access by default.		✔	✔
V8.5	Verify security logging controls provide the ability to log both success and failure events that are identified as security-relevant.		✔	✔
V8.6	Verify that each log event includes: a timestamp from a reliable source, severity level of the event, an indication that this is a security relevant event (if mixed with other logs), the identity of the user that caused the event (if there is a user associated with the event), the source IP address of the request associated with the event, whether the event succeeded or failed, and a description of the event.		✔	✔
V8.7	Verify that all events that include untrusted data will not execute as code in the intended log viewing software.			✔
V8.8	Verify that security logs are protected from unauthorized access and modification.		✔	✔
V8.9	Verify that there is a single application-level logging implementation that is used by the software.			✔

	ERROR HANDLING AND LOGGING VERIFICATION REQUIREMENT	LEVELS		
		1	2	3
V8.10	Verify that the application does not log application-specific sensitive data that could assist an attacker, including user's session identifiers and personal or sensitive information. The length and existence of sensitive data can be logged.		✔	✔
V8.11	Verify that a log analysis tool is available which allows the analyst to search for log events based on combinations of search criteria across all fields in the log record format supported by this system.		✔	✔
V8.13	Verify that all non-printable symbols and field separators are properly encoded in log entries, to prevent log injection.			✔
V8.14	Verify that log fields from trusted and untrusted sources are distinguishable in log entries.			✔
V8.15	Verify that logging is performed before executing the transaction. If logging was unsuccessful (e.g. disk full, insufficient permissions) the application fails safe. This is for when integrity and non-repudiation are a must.			✔

Table 6 - OWASP ASVS Error Handling and Logging Requirements (V8)

V9: Data Protection Verification Requirements

The table below defines the corresponding verification requirements that apply for each of the verification levels. Verification requirements for Level 0 are not defined by this standard.

	DATA PROTECTION VERIFICATION REQUIREMENT	LEVELS		
		1	2	3
V9.1	Verify that all forms containing sensitive information have disabled client side caching, including autocomplete features.	✓	✓	✓
V9.2	Verify that the list of sensitive data processed by this application is identified, and that there is an explicit policy for how access to this data must be controlled, and when this data must be encrypted (both at rest and in transit). Verify that this policy is properly enforced.			✓
V9.3	Verify that all sensitive data is sent to the server in the HTTP message body (i.e., URL parameters are never used to send sensitive data).	✓	✓	✓
V9.4	Verify that all cached or temporary copies of sensitive data sent to the client are protected from unauthorized access or purged/invalidated after the authorized user accesses the sensitive data (e.g., the proper no-cache and no-store Cache-Control headers are set).		✓	✓
V9.5	Verify that all cached or temporary copies of sensitive data stored on the server are protected from unauthorized access or purged/invalidated after the authorized user accesses the sensitive data.		✓	✓
V9.6	Verify that there is a method to remove each type of sensitive data from the application at the end of its required retention period.			✓
V9.7	Verify the application minimizes the number of parameters sent to untrusted systems, such as hidden fields, Ajax variables, cookies and header values.			✓
V9.8	Verify the application has the ability to detect and alert on abnormal numbers of requests for information or processing high value transactions for that user role, such as screen scraping, automated use of web service extraction, or data loss prevention. For example, the average user should not be able to access more than 5 records per hour or 30 records per day, or add 10 friends to a social network per minute.			✓

Table 7 - OWASP ASVS Data Protection Requirements (V9)

V10: Communications Security Verification Requirements

The table below defines the corresponding verification requirements that apply for each of the verification levels. Verification requirements for Level 0 are not defined by this standard.

	COMMUNICATIONS SECURITY VERIFICATION REQUIREMENT	LEVELS		
		1	2	3
V10.1	Verify that a path can be built from a trusted CA to each Transport Layer Security (TLS) server certificate, and that each server certificate is valid.	✓	✓	✓
V10.2	Verify that failed TLS connections do not fall back to an insecure HTTP connection.			✓
V10.3	Verify that TLS is used for all connections (including both external and backend connections) that are authenticated or that involve sensitive data or functions.		✓	✓
V10.4	Verify that backend TLS connection failures are logged.		✓	✓
V10.5	Verify that certificate paths are built and verified for all client certificates using configured trust anchors and revocation information.			✓
V10.6	Verify that all connections to external systems that involve sensitive information or functions are authenticated.		✓	✓
V10.7	Verify that all connections to external systems that involve sensitive information or functions use an account that has been set up to have the minimum privileges necessary for the application to function properly.		✓	✓
V10.8	Verify that there is a single standard TLS implementation that is used by the application that is configured to operate in an approved mode of operation (See http://csrc.nist.gov/groups/STM/cmvp/documents/fips140-2/FIPS1402IG.pdf).			✓
V10.9	Verify that specific character encodings are defined for all connections (e.g., UTF-8).			✓

Table 8 - OWASP ASVS Communications Security Requirements (V10)

V11: HTTP Security Verification Requirements

The table below defines the corresponding verification requirements that apply for each of the verification levels. Verification requirements for Level 0 are not defined by this standard.

	HTTP SECURITY VERIFICATION REQUIREMENT	LEVELS		
		1	2	3
V11.2	Verify that the application accepts only a defined set of HTTP request methods, such as GET and POST and unused methods are explicitly blocked.	✓	✓	✓
V11.3	Verify that every HTTP response contains a content type header specifying a safe character set (e.g., UTF-8).	✓	✓	✓
V11.6	Verify that HTTP headers in both requests and responses contain only printable ASCII characters.		✓	✓
V11.8	Verify that HTTP headers and / or other mechanisms for older browsers have been included to protect against clickjacking attacks.	✓	✓	✓
V11.9	Verify that HTTP headers added by a frontend (such as X-Real-IP), and used by the application, cannot be spoofed by the end user.		✓	✓
V11.10	Verify that the HTTP header, X-Frame-Options is in use for sites where content should not be viewed in a 3rd-party X-Frame. A common middle ground is to send SAMEORIGIN, meaning only websites of the same origin may frame it.		✓	✓
V11.12	Verify that the HTTP headers do not expose detailed version information of system components.		✓	✓

Table 9 - OWASP ASVS HTTP Security Requirements (V11)

V13: Malicious Controls Verification Requirements

The table below defines the corresponding verification requirements that apply for each of the verification levels. Verification requirements for Level 0 are not defined by this standard.

	MALICIOUS CONTROLS VERIFICATION REQUIREMENT	LEVELS		
		1	2	3
V13.1	Verify that no malicious code is in any code that was either developed or modified in order to create the application.			✔
V13.2	Verify that the integrity of interpreted code, libraries, executables, and configuration files is verified using checksums or hashes.			✔
V13.3	Verify that all code implementing or using authentication controls is not affected by any malicious code.			✔
V13.4	Verify that all code implementing or using session management controls is not affected by any malicious code.			✔
V13.5	Verify that all code implementing or using access controls is not affected by any malicious code.			✔
V13.6	Verify that all input validation controls are not affected by any malicious code.			✔
V13.7	Verify that all code implementing or using output validation controls is not affected by any malicious code.			✔
V13.8	Verify that all code supporting or using a cryptographic module is not affected by any malicious code.			✔
V13.9	Verify that all code implementing or using error handling and logging controls is not affected by any malicious code.			✔
V13.10	Verify all malicious activity is adequately sandboxed.			✔
V13.11	Verify that sensitive data is rapidly sanitized from memory as soon as it is no longer needed and handled in accordance to functions and techniques supported by the framework/library/operating system.			✔

Table 10 - OWASP ASVS Malicious Controls Requirements (V13)

V15: Business Logic Verification Requirements

The table below defines the corresponding verification requirements that apply for each of the verification levels. Verification requirements for Level 0 are not defined by this standard.

	BUSINESS LOGIC VERIFICATION REQUIREMENT	LEVELS		
		1	2	3
V15.1	Verify the application processes or verifies all high value business logic flows in a trusted environment, such as on a protected and monitored server.		✔	✔
V15.2	Verify the application does not allow spoofed high value transactions, such as allowing Attacker User A to process a transaction as Victim User B by tampering with or replaying session, transaction state, transaction or user IDs.		✔	✔
V15.3	Verify the application does not allow high value business logic parameters to be tampered with, such as (but not limited to): price, interest, discounts, PII, balances, stock IDs, etc.		✔	✔
V15.4	Verify the application has defensive measures to protect against repudiation attacks, such as verifiable and protected transaction logs, audit trails or system logs, and in highest value systems real time monitoring of user activities and transactions for anomalies.		✔	✔
V15.5	Verify the application protects against information disclosure attacks, such as direct object reference, tampering, session brute force or other attacks.		✔	✔
V15.6	Verify the application has sufficient detection and governor controls to protect against brute force (such as continuously using a particular function) or denial of service attacks.		✔	✔
V15.7	Verify the application has sufficient access controls to prevent elevation of privilege attacks, such as allowing anonymous users from accessing secured data or secured functions, or allowing users to access each other's details or using privileged functions.		✔	✔
V15.8	Verify the application will only process business logic flows in sequential step order, with all steps being processed in realistic human time, and not process out of order, skipped steps, process steps from another user, or too quickly submitted transactions.		✔	✔
V15.9	Verify the application has additional authorization (such as step up or adaptive authentication) for lower value systems, and / or segregation of		✔	✔

	BUSINESS LOGIC VERIFICATION REQUIREMENT	LEVELS		
		1	2	3
	duties for high value applications to enforce anti-fraud controls as per the risk of application and past fraud.			
V15.10	Verify the application has business limits and enforces them in a trusted location (as on a protected server) on a per user, per day or daily basis, with configurable alerting and automated reactions to automated or unusual attack. Examples include (but not limited to): ensuring new SIM users don't exceed $10 per day for a new phone account, a forum allowing more than 100 new users per day or preventing posts or private messages until the account has been verified, a health system should not allow a single doctor to access more patient records than they can reasonably treat in a day, or a small business finance system allowing more than 20 invoice payments or $1000 per day across all users. In all cases, the business limits and totals should be reasonable for the business concerned. The only unreasonable outcome is if there are no business limits, alerting or enforcement.		✔	✔

Table 11 - OWASP ASVS Business Logic Requirements (V15)

V16: Files and Resources Verification Requirements

The table below defines the corresponding verification requirements that apply for each of the verification levels. Verification requirements for Level 0 are not defined by this standard.

	FILES AND RESOURCES VERIFICATION REQUIREMENT	LEVELS		
		1	2	3
V16.1	Verify that URL redirects and forwards do not include unvalidated data.	✔	✔	✔
V16.2	Verify that file names and path data obtained from untrusted sources is canonicalized to eliminate path traversal attacks.	✔	✔	✔
V16.3	Verify that files obtained from untrusted sources are scanned by antivirus scanners to prevent upload of known malicious content.	✔	✔	✔
V16.4	Verify that parameters obtained from untrusted sources are not used in manipulating filenames, pathnames or any file system object without first being canonicalized and input validated to prevent local file inclusion attacks.	✔	✔	✔
V16.5	Verify that parameters obtained from untrusted sources are canonicalized, input validated, and output encoded to prevent remote file inclusion attacks, particularly where input could be executed, such as header, source, or template inclusion	✔	✔	✔
V16.6	Verify remote IFRAMEs and HTML5 cross-domain resource sharing does not allow inclusion of arbitrary remote content.	✔	✔	✔
V16.7	Verify that files obtained from untrusted sources are stored outside the webroot.		✔	✔
V16.8	Verify that web or application server is configured by default to deny access to remote resources or systems outside the web or application server.		✔	✔
V16.9	Verify the application code does not execute uploaded data obtained from untrusted sources.		✔	✔
V16.10	Verify if Flash, Silverlight or other rich internet application (RIA) cross domain resource sharing configuration is configured to prevent unauthenticated or unauthorized remote access.		✔	✔

Table 12 - OWASP ASVS File and Resource Requirements (V16)

V17: Mobile Verification Requirements

The table below defines the corresponding verification requirements that apply for each of the verification levels. Verification requirements for Level 0 are not defined by this standard.

	MOBILE VERIFICATION REQUIREMENT	LEVELS		
		1	2	3
V17.1	Verify that the client validates SSL certificates	✓	✓	✓
V17.2	Verify that unique device ID (UDID) values are not used as security controls.	✓	✓	✓
V17.3	Verify that the mobile app does not store sensitive data onto shared resources on the device (e.g. SD card or shared folders)	✓	✓	✓
V17.4	Verify that sensitive data is not stored in SQLite database on the device.	✓	✓	✓
V17.5	Verify that secret keys or passwords are not hard-coded in the executable.		✓	✓
V17.6	Verify that the mobile app prevents leaking of sensitive data via auto-snapshot feature of iOS.		✓	✓
V17.7	Verify that the app cannot be run on a jailbroken or rooted device.		✓	✓
V17.8	Verify that the session timeout is of a reasonable value.		✓	✓
V17.9	Verify the permissions being requested as well as the resources that it is authorized to access (i.e. AndroidManifest.xml, iOS Entitlements) .		✓	✓
V17.10	Verify that crash logs do not contain sensitive data.		✓	✓
V17.11	Verify that the application binary has been obfuscated.			✓

	MOBILE VERIFICATION REQUIREMENT	LEVELS		
		1	2	3
V17.12	Verify that all test data has been removed from the app container (.ipa, .apk, .bar).		✓	✓
V17.13	Verify that the application does not log sensitive data to the system log or filesystem.		✓	✓
V17.14	Verify that the application does not enable autocomplete for sensitive text input fields, such as passwords, personal information or credit cards.		✓	✓
V17.15	Verify that the mobile app implements certificate pinning to prevent the proxying of app traffic.			✓
V17.16	Verify no misconfigurations are present in the configuration files (Debugging flags set, world readable/writable permissions) and that, by default, configuration settings are set to their safest/most secure value.			✓
V17.17	Verify any 3rd-party libraries in use are up to date, contain no known vulnerabilities.			✓
V17.18	Verify that web data, such as HTTPS traffic, is not cached.			✓
V17.19	Verify that the query string is not used for sensitive data. Instead, a POST request via SSL should be used with a CSRF token.			✓
V17.20	Verify that, if applicable, any personal account numbers are truncated prior to storing on the device.			✓
V17.21	Verify that the application makes use of Address Space Layout Randomization (ASLR).			✓
V17.22	Verify that data logged via the keyboard (iOS) does not contain credentials, financial information or other sensitive data.			✓
V17.23	If an Android app, verify that the app does not create files with permissions of MODE_WORLD_READABLE or MODE_WORLD_WRITABLE			✓
V17.24	Verify that sensitive data is stored in a cryptographically secure manner (even when stored in the iOS keychain).			✓
V17.25	Verify that anti-debugging and reverse engineering mechanisms are implemented in the app.			✓
V17.26	Verify that the app does not export sensitive activities, intents, content providers etc. on Android.			✓

	MOBILE VERIFICATION REQUIREMENT	LEVELS		
		1	2	3
V17.27	Verify that mutable structures have been used for sensitive strings such as account numbers and are overwritten when not used. (Mitigate damage from memory analysis attacks).			✔
V17.28	Verify that any exposed intents, content providers and broadcast receivers perform full data validation on input (Android).			✔

Table 13 - OWASP ASVS Mobile Requirements (V17)

Appendix A: Applying ASVS in Practice

Different threats have different motivations, and some industries have unique information and technology assets as well as regulatory compliance requirements.

Below we provide industry-specific guidance regarding recommended ASVS levels. Although some unique criteria and some differences in threats exist for each industry, a common theme throughout all industry segments is that opportunistic attackers will look for any vulnerable applications reachable through the Internet, which is why ASVS Level 1 is recommended for all Internet-accessible applications regardless of industry. This is a suggested starting point, considering a small number of risk factors. Organizations are strongly encouraged to look more deeply at their unique risk characteristics based on the nature of their business. At the other end of the spectrum is ASVS Level 3, which is reserved for those cases that might endanger human safety or when a full application breach could severely impact the organization.

INDUSTRY SEGMENT	THREAT PROFILE	SUGGESTED ASVS LEVEL
Finance and Insurance	Although this segment will experience attempts from opportunistic attackers, it is often viewed as a high value target by motivated attackers and attacks are often financially motivated. Commonly, attackers are looking for sensitive data or account credentials that can be used to commit fraud or to benefit directly by leveraging money movement functionality built into applications. Techniques often include stolen credentials, application-level attacks, and social engineering. Some major compliance considerations include Payment Card Industry Data Security Standard (PCI DSS), Gramm-Leech	*Level 1: all Internet-accessible applications.*

INDUSTRY SEGMENT	THREAT PROFILE	SUGGESTED ASVS LEVEL
	Bliley act, Sarbanes Oxley (SOX).	
		Level 2: applications that contain sensitive information like credit card numbers, personal information, can move limited amounts of money in limited ways. Examples include: (i) transfer money between accounts at the same institution or (ii) a slower form of money movement (e.g. ACH) with transaction limits or (iii) wire transfers with hard transfer limits within a period of time.
		Level 3: applications that contain large amounts of sensitive information or that allow either rapid transfer of large sums of money (e.g. wire transfers) or transfer of large sums of money in the form of individual transactions or as a batch of smaller transfers.
Manufacturing, Professional, Transportation, Technology, Utilities, Infrastructure, Defense	These industries may not appear to have very much in common, but the threat actors who are likely to attack organizations in this segment are more likely to perform focused attacks with more time, skill, and resources. Often the sensitive information or systems are not easy to locate and require leveraging insiders and social engineering techniques. Attacks may involve insiders, outsiders, or be collusion between the two. Their goals may include gaining access to intellectual property for strategic or technological advantage. We also do not want to overlook attackers looking to abuse application functionality influence the behaviour of or disrupt	*Level 1: all Internet-accessible applications.*

INDUSTRY SEGMENT	THREAT PROFILE	SUGGESTED ASVS LEVEL
	sensitive systems.	
		Level 2: applications containing internal information or information about employees that may be leveraged in social engineering. Applications containing non-essential, but important intellectual property or trade secrets.
		Level 3: applications containing valuable intellectual property, trade secrets, or government secrets (e.g. in the United States this may be anything classified at Secret or above) that is critical to the survival or success of the organization. Applications controlling sensitive functionality (e.g. transit, manufacturing equipment, control systems) or that have the possibility of threatening safety of life.
Healthcare	Most attackers are looking for sensitive data that can be used to directly or indirectly profit from to include personally identifiable information and payment data. Often the data can be used for identity theft, fraudulent payments, or a variety of fraud schemes. HIPAA and HITECH Act are major compliance drivers in the United States and include data breach notification requirements.	*Level 1: all Internet-accessible applications.*
		Level 2: applications with small or moderate amounts of sensitive medical information (Protected Health Information), Personally Identifiable Information, or payment data.

INDUSTRY SEGMENT	THREAT PROFILE	SUGGESTED ASVS LEVEL
		Level 3: Applications used to control medical equipment, devices, or records that may endanger human life. Payment and Point of Sale systems (POS) that contain large amounts of transaction data that could be used to commit fraud. This includes any administrative interfaces for these applications.
Retail, Food, Hospitality	Many of the attackers in this segment utilize opportunistic "smash and grab" tactics. However, there is also a regular threat of specific attacks on applications known to contain payment information, perform financial transactions, or store personally identifiable information. Although less likely than the threats mentioned above, there is also the possibility of more advanced threats attacking this industry segment to steal intellectual property, gain competitive intelligence, or gain an advantage with the target organization or a business partner in negotiations.	*Level 1: all Internet-accessible applications.*
		Level 2: Suitable for business applications, product catalogue information, internal corporate information, and applications with limited user information (e.g. contact information). Applications with small or moderate amounts of payment data or checkout functionality.
		Level 3: Payment and Point of Sale systems (POS) that contain large amounts of transaction data that could be used to commit fraud. This includes any administrative interfaces for these applications. Applications with a large volume of sensitive information like full credit

INDUSTRY SEGMENT	THREAT PROFILE	SUGGESTED ASVS LEVEL
		card numbers, mother's maiden name, social security numbers etc.

Table 14 – Applying OWASP ASVS in Practice

Appendix B: Glossary

- *Access Control* – A means of restricting access to files, referenced functions, URLs, and data based on the identity of users and/or groups to which they belong.
- *Application Component* – An individual or group of source files, libraries, and/or executables, as defined by the verifier for a particular application.
- *Application Security* – Application-level security focuses on the analysis of components that comprise the application layer of the Open Systems Interconnection Reference Model (OSI Model), rather than focusing on for example the underlying operating system or connected networks.
- *Application Security Verification* – The technical assessment of an application against the OWASP ASVS.
- *Application Security Verification Report* – A report that documents the overall results and supporting analysis produced by the verifier for a particular application.
- *Application Security Verification Standard (ASVS)* – An OWASP standard that defines four levels of application security verification for applications.
- *Authentication* – The verification of the claimed identity of an application user.
- *Automated Verification* – The use of automated tools (either dynamic analysis tools, static analysis tools, or both) that use vulnerability signatures to find problems.
- *Back Doors* – A type of malicious code that allows unauthorized access to an application.
- *Blacklist* – A list of data or operations that are not permitted, for example a list of characters that are not allowed as input.
- *Certificate Authority (CA)* – An entity that issues digital certificates.
- *Common Criteria (CC)* – A multipart standard that can be used as the basis for the verification of the design and implementation of security controls in IT products.
- *Communication Security* – The protection of application data when it is transmitted between application components, between clients and servers, and between external systems and the application.
- *Cross-Site Scripting (XSS)* – A security vulnerability typically found in web applications allowing the injection of client-side scripts into content.
- *Cascading Style Sheets (CSS)* - A style sheet language used for describing the presentation semantics of document written in a markup language, such as HTML.
- *Design Verification* – The technical assessment of the security architecture of an application.
- Internal Verification – The technical assessment of specific aspects of the security architecture of an application as defined in the OWASP ASVS.
- *Cryptographic module* – Hardware, software, and/or firmware that implements cryptographic algorithms and/or generates cryptographic keys.
- *Denial of Service (DOS) Attacks* – The flooding of an application with more requests than it can handle.

- *Dynamic Verification* – The use of automated tools that use vulnerability signatures to find problems during the execution of an application.
- *Easter Eggs* – A type of malicious code that does not run until a specific user input event occurs.
- *External Systems* – A server-side application or service that is not part of the application.
- *FIPS 140-2* – A standard that can be used as the basis for the verification of the design and implementation of cryptographic modules
- Globally Unique Identifier *(GUID)* – a unique reference number used as an identifier in software.
- Hyper Text Transfer Protocol *(HTTP)* – An application protocol for distributed, collaborative, hypermedia information systems. It is the foundation of data communication for the World Wide Web.
- *HTML* – The main markup language for the creation of web pages and other information displayed in a web browser.
- *Input Validation* – The canonicalization and validation of untrusted user input.
- *LDAP* – An application protocol for accessing and maintaining distributed directory information services over a network.
- *Malicious Code* – Code introduced into an application during its development unbeknownst to the application owner, which circumvents the application's intended security policy. Not the same as malware such as a virus or worm!
- *Malware* – Executable code that is introduced into an application during runtime without the knowledge of the application user or administrator.
- *Open Web Application Security Project (OWASP)* – The Open Web Application Security Project (OWASP) is a worldwide free and open community focused on improving the security of application software. Our mission is to make application security "visible," so that people and organizations can make informed decisions about application security risks. See: http://www.owasp.org/
- *Output Validation* – The canonicalization and validation of application output to Web browsers and to external systems.
- *OWASP Enterprise Security API (ESAPI)* – A free and open collection of all the security methods that developers need to build secure Web applications. See: http://www.owasp.org/index.php/ESAPI
- *OWASP Risk Rating Methodology* – A risk rating methodology that has been customized for application security. See: http://www.owasp.org/index.php/How_to_value_the_real_risk
- *OWASP Testing Guide* – A document designed to help organizations understand what comprises a testing program, and to help them identify the steps needed to build and operate that testing program. See: http://www.owasp.org/index.php/Category:OWASP_Testing_Project
- *OWASP Top Ten* – A document that represents a broad consensus about what the most critical Web application security flaws are. See: http://www.owasp.org/index.php/Top10
- *Positive* – See whitelist.
- *Salami Attack* – A type of malicious code that is used to redirect small amounts of money without detection in financial transactions.
- *Security Architecture* – An abstraction of an application's design that identifies and describes where and how security controls are used, and also identifies and describes the location and sensitivity of both user and application data.

- *Security Control* – A function or component that performs a security check (e.g. an access control check) or when called results in a security effect (e.g. generating an audit record).
- *Security Configuration* – The runtime configuration of an application that affects how security controls are used.
- *Static Verification* – The use of automated tools that use vulnerability signatures to find problems in application source code.
- *SQL Injection (SQLi)* – A code injection technique used to attack data driven applications, in which malicious SQL statements are inserted into an entry point.
- *Target of Verification (TOV)* – If you are performing application security verification according to the OWASP ASVS requirements, the verification will be of a particular application. This application is called the "Target of Verification" or simply the TOV.
- *Threat Modeling* - A technique consisting of developing increasingly refined security architectures to identify threat agents, security zones, security controls, and important technical and business assets.
- *Time Bomb* – A type of malicious code that does not run until a preconfigured time or date elapses.
- *Transport Layer Security* – Cryptographic protocols that provide communication security over the Internet
- *UAT* – Traditionally a test environment that behaves like the production environment where all software testing is performed before going live.
- *URI/URL* – A Uniform Resource Identifier is a string of characters used to identify a name or a web resource. A Uniform Resource Locator is often used as a reference to a resource.
- *Verifier* - The person or team that is reviewing an application against the OWASP ASVS requirements.
- *Whitelist* – A list of permitted data or operations, for example a list of characters that are allowed to perform input validation.
- *XML* – A markup language that defines a set of rules for encoding documents.

Appendix C: Where To Go From Here

The OWASP ASVS is a living document. If you are performing an application security verification according to this standard, then you should always review the articles that can be found on the OWASP ASVS project page.

OWASP is the premier site for Web application security. The OWASP site hosts many projects, forums, blogs, presentations, tools, and papers. Additionally, OWASP hosts two major Web application security conferences per year, and has over 80 local chapters. The OWASP ASVS project page can be found here http://www.owasp.org/index.php/ASVS

The following OWASP projects are most likely to be useful to users/adopters of this standard:

- *OWASP Top Ten Project* - http://www.owasp.org/index.php/Top_10
- *OWASP Code Review Guide* - http://www.owasp.org/index.php/Category:OWASP_Code_Review_Project
- *OWASP Testing Guide* - http://www.owasp.org/index.php/Testing_Guide
- *OWASP Enterprise Security API (ESAPI) Project* - http://www.owasp.org/index.php/ESAPI
- *OWASP Legal Project* - http://www.owasp.org/index.php/Category:OWASP_Legal_Project

Similarly, the following Web sites are most likely to be useful to users/adopters of this standard:

- *OWASP* - http://www.owasp.org
- *MITRE* - Common Weakness Enumeration – Vulnerability Trends, http://cwe.mitre.org/documents/vuln-trends.html
- *PCI Security Standards Council* - publishers of the PCI standards, relevant to all organizations processing or holding credit card data, https://www.pcisecuritystandards.org
- *PCI Data Security Standard (DSS) v2.0* - https://www.pcisecuritystandards.org/security_standards/documents.php?document=pci_dss_v2-0#pci_dss_v2-0