Application Security Verification Standard 4.0

Finale Version

März 2019

Frontispiece

Der ASVS Standard

Der Application Security Verification Standard ist eine Liste von Anwendungssicherheits-Anforderungen und -Tests, die von Architekten, Entwicklern, Testern, Sicherheitsexperten, Werkzeugherstellern und Verbrauchern verwendet werden können, um sichere Anwendungen zu konzipieren, zu implementieren, zu testen und zu verifizieren.

Copyright und Lizenz

Version 4.0.1, März 2019, deutsche Übersetzung, März 2020

Projektleiter

- Andrew van der Stock
- Daniel Cuthbert
- Jim Manico
- Josh C Grossman
- Mark Burnett

Beitragende und Rezensenten

- Osama Elnaggar
- Erlend Oftedal
- Serg Belkommen
- David Johansson
- Tonimir Kisasondi
- Ron Perris
- Jason Axley

- Abhay Bhargav
- Benedikt Bauer
- Elar Lang
- ScriptingXSS
- Philippe De Ryck
- Grog's Axle
- Marco Schnüriger
- Jacob Salassi
- Glenn ten Cate
- Anthony Weems
- bschach
- javixeneize
- Dan Cornell
- hello7s
- Lewis Ardern
- Jim Newman
- Stuart Gunter
- Geoff Baskwill
- Talargoni
- Ståle Pettersen
- Kelby Ludwig
- Jason Morrow
- Rogan Dawes
- Daniël Geerts

Der Application Security Verification Standard baut auf den Vorarbeiten der Beteiligten von ASVS 1.0 im Jahr 2008 bis 3.0 im Jahr 2016 auf. Viele der Struktur- und Verifizierungselemente, die heute noch in der ASVS enthalten sind, wurden ursprünglich von Mike Boberski, Jeff Williams und Dave Wichers geschrieben, aber es gibt darüber hinaus noch viele weitere Mitwirkende. Vielen Dank an alle, die sich bisher beteiligt haben. Für eine umfassende Liste all jener, die zu früheren Versionen beigetragen haben, konsultieren Sie bitte die jeweiligen Vorversionen.

Wenn eine Danksagung in der obigen 4.0-Gutschriftenliste fehlt, wenden Sie sich bitte an vanderaj@owasp.org oder stellen Sie ein Ticket bei GitHub ein, um in zukünftigen 4.x-Updates anerkannt zu werden.

Vorwort

Willkommen beim Application Security Verification Standard (ASVS) Version 4.0. Der ASVS ist ein von der Community getriebenes Vorhaben mit dem Ziel, ein Rahmenwerk von Sicherheitsanforderungen und -kontrollen zu schaffen. Dabei soll sich dieses auf die Definition der funktionalen und nicht-funktionalen Sicherheitskontrollen konzentrieren, die beim Entwurf, der Entwicklung und dem Testen moderner Webanwendungen und Webservices erforderlich sind.

ASVS v4.0 ist das Ergebnis seitens der Bemühungen der Community und des Feedbacks aus der Industrie des letzten Jahrzehnts. Wir haben versucht, die Einführung des ASVS zu erleichtern um dies für eine Vielzahl von

verschiedenen Anwendungsfällen während des gesamten Lebenszyklus der sicheren Softwareentwicklung zu zu ermöglichen.

Wir wissen, dass es höchstwahrscheinlich niemals einen 100%igen Konsens über den Inhalt eines Webanwendungsstandards, einschließlich des ASVS, geben wird. Die Risikoanalyse ist immer bis zu einem gewissen Grad subjektiv, was eine Herausforderung darstellt, wenn man versucht, alles in einem Standard zu verallgemeinern. Wir hoffen jedoch, dass die letzten Aktualisierungen in dieser Version ein Schritt in die richtige Richtung sind und die in diesem wichtigen Industriestandard eingeführten Konzepte weiter verbessern.

Was ist neu in 4.0

Die größte Änderung in dieser Version ist die Aufnahme der NIST 800-63-3 Richtlinien zur digitalen Identität, die moderne, stichhaltige und erweiterte Authentifizierungskontrollen einführt. Obwohl wir einen gewissen Rückschritt bei der Angleichung an einen fortgeschrittenen Authentifizierungsstandard feststellen, sind wir der Meinung, dass eine Angleichung der Standards unerlässlich ist, vor allem dann, wenn ein anderer angesehener evidenzbasierter Anwendungssicherheitsstandard bereits vorhanden ist.

Standards der Informationssicherheit sollten stets versuchen, die Anzahl der besonderen Anforderungen zu minimieren, so dass sich Unternehmen, die die Anforderungen erfüllen möchten, nicht über konkurrierende oder inkompatible Mechanismen entscheiden müssen. Sowohl die OWASP Top 10 2017 als auch nun auch der OWASP Application Security Verification Standard haben sich hinsichtlich Authentifizierung und Sitzungsmanagement an NIST 800-63 angelehnt. Wir ermutigen andere standardsetzende Gremien, mit uns, dem NIST und anderen zusammenzuarbeiten, um zu einem allgemein akzeptierten Satz von Anwendungssicherheitsmechanismen zu kommen, um die Sicherheit zu maximieren und gleichzeitig die Kosten für die Einhaltung von Vorschriften zu minimieren.

Alle Kapitel des ASVS 4.0 wurden von Anfang bis Ende vollständig neu durchnummeriert. Das neue Nummerierungsschema ermöglichte es uns, Lücken aus längst überholten Kapiteln zu schließen und längere Kapitel aufzuteilen, um die Anzahl der Kontrollen, die ein Entwickler oder ein Team einhalten muss, zu minimieren. Wenn eine Anwendung beispielsweise keine JSON Web tokens (JWT) verwendet, ist der gesamte Abschnitt über JWT im Sitzungsmanagement nicht relevant.

Neu in Version 4.0 ist eine umfassende Abbildung der Common Weakness Enumeration (CWE), eine der am häufigsten gewünschten Erweiterungen, die wir in den letzten zehn Jahren hatten. Die CWE-Zuordnung ermöglicht es Werkzeugherstellern und Anwendern von Software für Schwachstellenmanagement, die Ergebnisse anderer Werkzeuge und früherer ASVS-Versionen mit der Version 4.0 bzw. neueren Versionen abzugleichen. Um Platz für den CWE-Eintrag zu schaffen, mussten wir die Spalte "Seit" entfernen Da wird alle Kapitel neu durchnummeriert haben ist diese Angabe weniger sinnvoll als in früheren Versionen des ASVS. Nicht jeder Eintrag in der ASVS hat eine zugehörige CWE, und da die CWE sehr viele Dubletten aufweist, haben wir versucht, den am häufigsten verwendeten und nicht unbedingt den nächstliegenden Eintrag zu verwenden. Verifizierungskontrollen lassen sich nicht immer eindeutig auf entsprechende Schwachstellen abbilden. Wir begrüßen jedoch die laufende Diskussion über die Schließung dieser Lücke mit der CWE-Community und der Informationssicherheit im Allgemeinen.

Wir haben daran gearbeitet, die Anforderungen der OWASP Top 10 2017 und der OWASP Proactive Controls 2018 weitestgehend zu erfüllen oder gar zu übertreffen. Da die OWASP Top 10 2017 das absolute Minimum zur Vermeidung von Nachlässigkeiten darstellt, haben wir bewusst alle Top 10 Anforderungen, bis auf spezifische Anforderungen an die Protokollierung, als Kontrollen der Stufe 1 eingeführt. Damit wollen wir es den OWASP Top 10-Anwendern erleichtern, einen tatsächlichen Sicherheitsstandard zu erreichen.

Wir wollten sicherstellen, dass die Stufe 1 des ASVS 4.0 eine umfassende Obermenge der Abschnitte 6.5 des PCI DSS 3.2.1 für Anwendungsdesign, Kodierung, Tests, Codereviews für Sicherheit sowie Penetrationstests darstellt. Dies erforderte zusätzlich zu den bestehenden branchenführenden Anforderungen an die Anwendungs- und Web-Service-Verifikation auch die Abdeckung von Bufferoverflows und unsicheren Speicheroperationen in V5 und unsichere speicherbezogene Compilerflags in V14.

Wir haben damit auch die Umstellung des ASVS weg von den monolithischen, lediglich serverseitigen Kontrollen hin zur Bereitstellung von Sicherheitskontrollen für alle modernen Anwendungen und APIs abgeschlossen. In Zeiten funktionaler Programmierung, von Serverless APIs, mobiler Anwendungen, der Cloud,

Containern, von CI/CD, DevSecOps, Federation und weiteren können wir die moderne Anwendungsarchitektur nicht länger ignorieren. Moderne Anwendungen sind ganz anders konzipiert als diejenigen, die bei der Veröffentlichung des ursprünglichen ASVS im Jahr 2009 umgesetzt wurden. Die ASVS muss immer weit in die Zukunft blicken, damit wir unserem primären Publikum - den Entwicklern - fundierte Ratschläge geben können. Wir haben alle Anforderungen, die davon ausgehen, dass Anwendungen auf Systemen einer einzigen Organisation ausgeführt werden, genau beschrieben oder entfernt.

Aufgrund der Größe des ASVS 4.0 und unseres Wunsches, die Basis für alle weiteren ASVS-Bemühungen zu werden, haben wir die mobile Sektion zugunsten des Mobile Application Security Verification Standard (MASVS) aufgegeben. Der Anhang zum Internet der Dinge (IoT) wird ebenfalls in einer zukünftigen IoT-ASVS Version erscheinen, welches vom OWASP IoT Projekts betreut wird. Eine erste Vorschau auf den IoT-ASVS haben wir in Anhang C aufgenommen. Wir danken sowohl dem OWASP Mobile Team als auch dem OWASP IoT-Projektteam für ihre Unterstützung des ASVS und freuen uns darauf, in Zukunft mit ihnen zusammenzuarbeiten, um ergänzende Standards zu schaffen.

Und schließlich haben wir weniger wirksame Kontrollen enttarnt und entfernt. Im Laufe der Zeit wurde die ASVS zu einem umfassenden Satz von Kontrollen, aber nicht alle Kontrollen sind gleichwertig hinsichtlich der Herstellung sicherer Software. Diese Bemühungen zur Beseitigung von Elementen mit geringer Auswirkung könnten noch weiter gehen. In einer künftigen Ausgabe des ASVS wird das Common Weakness Scoring System (CWSS) dazu beitragen, die wirklich wichtigen Kontrollen und die Kontrollen, die ausgemustert werden sollten, weiter zu priorisieren.

Ab der Version 4.0 wird sich der ASVS ausschließlich darauf konzentrieren, der führende Webanwendungs- und Servicestandard zu sein, der traditionelle und moderne Anwendungsarchitekturen sowie agile Sicherheitspraktiken und die DevSecOps-Kultur abdeckt.

Verwendung des ASVS

Der ASVS verfolgt zwei Hauptziele:

- Organisationen bei der Entwicklung und Wartung sicherer Anwendungen zu unterstützen.
- Sicherheitsdienstleistern, Anbietern von Sicherheitswerkzeugen und Verbrauchern die Möglichkeit zu geben, deren Anforderungen und Angebote dazu aufeinander abzustimmen.

Stufen des Application Security Verification Standards

Der Application Security Verification Standard definiert drei Verifizierungsstufen für Sicherheit, wobei jede Stufe immer weiter in die Tiefe geht.

- Die Stufe 1 ASVS ist für niedrige Sicherheitsstufen gedacht und ist vollständig durch Penetrationstests verifizierbar.
- Die Stufe 2 des ASVS ist für alle Anwendungen gedacht, die schützenwerte sensible Daten enthalten, und ist die empfohlene Stufe für die meisten Anwendungen.
- Die Stufe 3 des ASVS ist für die Anwendungen mit der höchsten Kritikalität gedacht - also Anwendungen, die Transaktionen von hohem Wert durchführen, sensible medizinische Daten enthalten bzw. alle Anwendungen, die ein Höchstmaß an Vertrauen erfordern.

Jede ASVS-Stufe enthält eine Liste von Sicherheitsanforderungen. Jede dieser Anforderungen kann auch auf sicherheitsspezifischen Features und Eigenschaften abgebildet werden, die von Entwicklern in die Software eingebaut werden müssen.

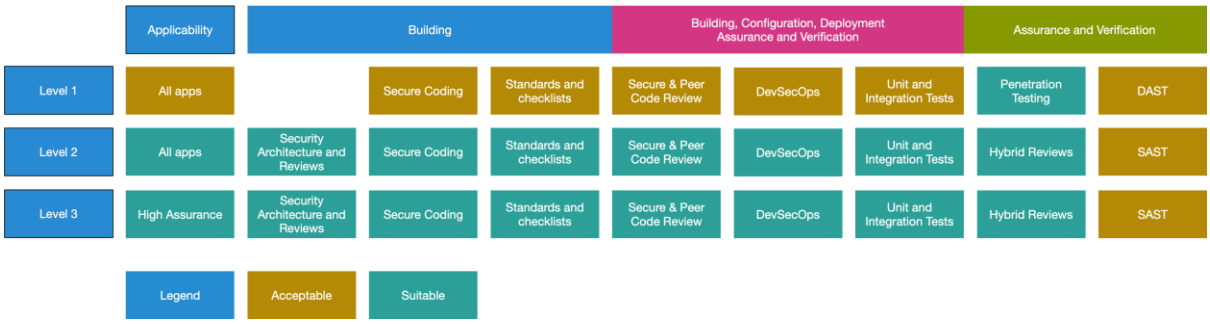

Abbildung 1 - Stufen des OWASP Application Security Verification Standard 4.0

Die Stufe 1 ist die einzige Stufe, die vollständig von Menschen mittels Penetrationtests prüfbar ist. Alle anderen Stufen erfordern zusätzlich Zugang zu Dokumentation, Quellcode, Konfiguration und den am Entwicklungsprozess beteiligten Personen. Aber selbst wenn die Stufe 1 "Black Box"-Tests (ohne Dokumentation und Quellcode) zulässt, ist dies keine wirksame Sicherungsmaßnahme und es sollte proaktiv von deren alleinigen Verwendung abgeraten werden. Böswillige Angreifer haben sehr viel Zeit zur Verfügung, während für die meisten Penetrationstests nur ein paar Wochen zur Verfügung stehen. Die Verteidiger müssen Sicherheitskontrollen einbauen, alle Schwachstellen schützen, finden und beheben und böswillige Akteure innerhalb einer angemessenen Zeit aufspüren und darauf reagieren. Böswillige Akteure haben im Wesentlichen unendlich viel Zeit und benötigen nur eine einzige undichte Verteidigung, eine einzige Schwachstelle oder eine fehlende Erkennung, um erfolgreich zu sein. Black-Box-Tests, die hektisch oft erst am Ende der Entwicklung oder womöglich gar nicht durchgeführt werden, können dieser Asymmetrie unmöglich mit Erfolg begegnen.

In den letzten 30 Jahren haben Black-Box-Tests immer wieder bewiesen, daß dabei kritische Sicherheitsprobleme übersehen werden, die dann zu immer massiveren Verletzungen der Datensicherheit führen. Wir befürworten nachdrücklich den Einsatz einer breiten Palette von Verfahren zur Sicherheitssicherung und Verifizierungsmaßnahmen. Dies umfasst die Ablösung klassischer Penetrationstests durch quellcodegeführte (hybride) Penetrationstests auf Stufe 1, mit vollem Zugang zu den Entwicklern und der Dokumentation während des gesamten Entwicklungsprozesses. Die Finanzaufsichtsbehörden tolerieren keine externen Finanzprüfungen ohne Zugang zu den Bilanzbüchern, Mustertransaktionen oder den Personen, welche die Kontrollen durchführen. Industrie und Regierungen müssen den gleichen Standard an Transparenz im Bereich der Softwareentwicklung fordern.

Wir empfehlen nachdrücklich die Verwendung von Sicherheitswerkzeugen innerhalb des Entwicklungsprozesses. DAST- und SAST-Werkzeuge können von der Build-Pipeline kontinuierlich verwendet werden, um leicht zu aufzuspürende Sicherheitsprobleme, die niemals vorhanden sein sollten, zu finden ("Low Hanging Fruits").

Automatisierte Werkzeuge und Online-Scans können nicht mehr als die Hälfte der ASVS ohne menschliche Hilfe abdecken. Wenn eine umfassende Testautomatisierung für jeden Build erforderlich ist, dann wird eine Kombination aus benutzerdefinierten Unit- und Integrationstests zusammen mit durch den Build initiierten Online-Scans verwendet. Tests der Geschäftslogik und der Zugriffskontrolle sind nur mit menschlicher Unterstützung möglich. Diese sollten in Unit- und Integrationstests umgesetzt werden.

Wie man diesen Standard verwendet

die beste Möglichkeit, den Application Security Verification Standard zu nutzen, ist diesen als Blaupause zur Erstellung einer Checkliste für die sichere Entwicklung zu verwenden, die speziell auf Ihre Anwendung, Plattform oder Organisation zugeschnitten ist. Durch die Anpassung des ASVS an Ihre Anwendungsfälle wird der Schwerpunkt auf die Sicherheitsanforderungen gelegt, die für Ihre Projekte und Umgebungen am wichtigsten sind.

Stufe 1 - Erste Schritte - automatisiert und Ausgangspunkt für weitere Stufen

Eine Anwendung erreicht ASVS Stufe 1, wenn sie sich angemessen gegen einfach zu entdeckende Sicherheitslücken der Anwendung verteidigt, die in den OWASP Top 10 und anderen ähnlichen Checklisten beschrieben sind.

Die Stufe 1 ist das absolute Minimum, das alle Anwendungen anstreben sollten. Es ist auch hilfreich als erster Schritt innerhalb eines mehrstufigen Prozesses oder wenn Anwendungen keine sensiblen Daten speichern bzw. verarbeiten und daher nicht die strengeren Kontrollen der Stufen 2 oder 3 benötigen. Kontrollen der Stufe 1 können entweder automatisch durch Werkzeuge oder ohne großen Aufwand manuell ohne Zugriff auf den Quellcode überprüft werden. Wir betrachten Stufe 1 als das für alle Anwendungen erforderliche Minimum.

Bedrohungen für die Anwendung werden höchstwahrscheinlich von Angreifern ausgehen, die einfache wenig aufwendige Techniken verwenden, um schnell zu findende und leicht ausnutzbare Schwachstellen zu identifizieren. Dies steht im Gegensatz zu einem entschlossenen Angreifer, der seine Energie gezielt für die Anwendung einsetzt. Wenn die von Ihrer Anwendung verarbeiteten Daten einen hohen Wert haben, wird man nur selten mit einer Überprüfung der Stufe 1 aufhören.

Stufe 2 - Der Standard für die meisten Anwendungen

Eine Anwendung erreicht ASVS Stufe 2 (d.h. den Standard), wenn sie die meisten Risiken, die heutzutage mit Software verbunden sind, angemessen abwehrt.

Stufe 2 stellt sicher, dass wirksame Sicherheitskontrollen vorhanden sind und innerhalb der Anwendung verwendet werden. Stufe 2 ist in der Regel für Anwendungen geeignet, die signifikante Business-zu-Business Transaktionen abwickeln. Einschließlich solcher, die Informationen aus dem Gesundheitswesen verarbeiten, geschäftskritische oder sensible Funktionen implementieren oder andere sensible Vermögenswerte verarbeiten, oder für Branchen, in denen die Integrität ein kritischer Aspekt zum Schutz ihres Geschäfts ist, wie z.B. die Spielindustrie, um Betrüger und Spiel-Hacks zu vereiteln.

Bedrohungen von Anwendungen der Stufe 2 sind in der Regel qualifizierte und motivierte Angreifer, die sich auf bestimmte Ziele konzentrieren und dabei Werkzeuge und Techniken einsetzen, die gut trainiert und wirksam sind, um Schwachstellen in Anwendungen zu entdecken und auszunutzen.

Stufe 3 - Anwendungen mit sehr hohen Sicherheitsanforderungen

ASVS Stufe 3 ist die höchste Verifizierungsstufe innerhalb des ASVS. Diese Stufe ist in der Regel Anwendungen vorbehalten, die ein erhebliches Maß an Sicherheitsverifikation erfordern, wie z.B. solche, die in Bereichen des Militärs, der Gesundheit und Safety, kritischen Infrastrukturen usw. zu finden sind.

Für Organisationen kann die ASVS-Stufe 3 nötig werden, wenn deren Anwendungen kritische Funktionen ausführen, bei denen ein Ausfall die Operationen der Organisation und sogar ihre Überlebensfähigkeit erheblich beeinträchtigen könnte. Nachstehend finden Sie ein Beispiel für die Anwendung von ASVS Stufe 3. Eine Anwendung erreicht ASVS-Stufe 3 (Fortgeschrittene Stufe), wenn sie sich angemessen gegen fortgeschrittene Sicherheitslücken einer Anwendung wehrt und auch die Prinzipien eines guten Sicherheitsdesigns verfolgt.

Eine Anwendung auf ASVS-Stufe 3 erfordert eine gründlichere Analyse der Architektur, der Codierung und des Testprozesses als alle anderen Stufen. Eine sichere Anwendung wird auf sinnvolle Weise modularisiert (um die Robustheit, Skalierbarkeit und vor allem den Aufbau von Sicherheitsschichten zu erleichtern), und jedes Modul (getrennt nach Netzwerkverbindung und/oder physischer Instanz) kümmert sich um seine eigenen Sicherheitsverantwortlichkeiten (*Defense in Depth*), die angemessen dokumentiert werden müssen. Zu den Verantwortlichkeiten gehören Kontrollen zur Gewährleistung der Vertraulichkeit (z.B. Verschlüsselung), Integrität (z.B. Transaktionen, Eingabevalidierung), Verfügbarkeit (z.B. entsprechende Lastverteilung), Authentifizierung (auch zwischen Systemen), Unabstreitbarkeit, Autorisierung und Auditierung (Protokollierung).

Anwendung des ASVS in der Praxis

Verschiedene Bedrohungen haben ihren Ursprung in unterschiedlichen Motivationen. Einige Branchen verfügen über einzigartige Informations- und Technologievorteile und bereichsspezifische regulatorische Anforderungen.

Organisationen wird empfohlen, ihre spezifischen Risikomerkmale auf der Grundlage ihrer Geschäftsart eingehend zu prüfen und auf Grundlage dieser Risiken und Geschäftsanforderungen die dafür geeignete ASVS-Ebene zu bestimmen.

Prüfung und Zertifizierung

Die Haltung der OWASP zu ASVS-Zertifizierungen und Vertrauenssiegeln

OWASP als herstellerneutrale, gemeinnützige Organisation zertifiziert derzeit keine Anbieter, Verifizierer oder Software für ASVS.

Alle derartigen Zusicherungen, Vertrauenssiegel oder Zertifizierungen werden von OWASP nicht offiziell überprüft, registriert oder zertifiziert, so dass eine Organisation, die sich auf eine solche Prüfung verlässt, vorsichtig sein muss, was das Vertrauen in eine dritte Partei oder ein Vertrauenssiegel betrifft, das eine ASVS-Zertifizierung für sich beansprucht.

Dies sollte aber Organisationen auch nicht daran hindern, derartige Dienstleistungen anzubieten, solange sie keine offizielle OWASP-Zertifizierung beanspruchen.

Leitfaden für zertifizierende Organisationen

Der Application Security Verification Standard kann für volltransparente Prüfungen von Anwendungen verwendet werden, einschließlich des offenen und ungehinderten Zugangs zu Schlüsselressourcen wie Architekten und Entwickler, Projektdokumentation, Quellcode, authentifizierter Zugang zu Testsystemen (inklusive des Zugangs zu Konten in jeder Rolle), insbesondere für Verifikationen der Stufen 2 und 3.

In der Vergangenheit haben Penetrationstests und Überprüfungen von sicherem Code ausnahmslos Probleme beinhaltet - das heißt, dass nur fehlgeschlagene Tests im Abschlussbericht erscheinen. Eine zertifizierende Organisation muss in jedem Bericht folgende Angaben mit aufnehmen:

- Den Umfang der Verifizierung (insbesondere, wenn eine Schlüsselkomponente außerhalb des Umfangs liegt, wie z.B. die SSO-Authentifizierung)
- Eine Zusammenfassung der Verifizierungsergebnisse, einschließlich der bestandenen und nicht bestandenen Tests, mit klaren Angaben zur Lösung der fehlgeschlagenen Tests.

Bestimmte Prüfungsanforderungen sind möglicherweise nicht auf die zu prüfende Anwendung anwendbar. Wenn Sie beispielsweise Ihren Kunden eine zustandslose Service-Schicht API ohne eine zugehörige Client-Implementierung zur Verfügung stellen, sind viele der Anforderungen in *V3 Session Management* nicht direkt anwendbar. In solchen Fällen kann eine zertifizierende Organisation immer noch die volle ASVS-Konformität beanspruchen, muss aber in jedem Bericht klar einen Grund für die Nichtanwendbarkeit solcher ausgeschlossener Prüfungsanforderungen angeben.

Die Aufbewahrung von detaillierten Arbeitsaufzeichnungen, Screenshots oder Videos, Skripts zur verlässlichen und wiederholten Ausnutzung eines Problems und elektronischen Aufzeichnungen von Tests, wie z.B. das Abfangen von Proxy-Protokollen und zugehörigen Notizen, wie z.B. einer Bereinigungsliste, gilt als Industriestandard und kann für die kritischsten Entwickler als Beweis für die Ergebnisse nützlich sein. Es reicht nicht aus, einfach ein Tool laufen zu lassen und über die Fehler zu berichten; dies liefert (überhaupt) keinen ausreichenden Beweis dafür, dass alle Probleme auf einer Zertifizierungsebene gründlich getestet und geprüft worden sind. Im Falle von Streitigkeiten sollte es genügend Sicherheitsbeweise geben, um nachzuweisen, dass jede einzelne verifizierte Anforderung tatsächlich auch getestet wurde.

Testmethoden

Zertifizierende Organisationen können die geeignete(n) Prüfmethode(n) frei wählen, sollten diese jedoch in einem Bericht aufführen.

Je nach der zu prüfenden Anwendung und den Prüfungsanforderungen können unterschiedliche Prüfmethoden verwendet werden, um ein hinreichendes Vertrauen in die Ergebnisse zu gewinnen. Die Überprüfung der Wirksamkeit von Eingabevalidierungsmechanismen einer Anwendung kann beispielsweise entweder mittels eines manuellen Penetrationstests oder mit Hilfe von Quellcodeanalysen durchgeführt werden.

Die Rolle automatisierter Sicherheitswerkzeuge

Der Einsatz von Werkzeugen für automatisierte Penetrationstests wird empfohlen, um eine möglichst hohe Testabdeckung zu erreichen.

Es ist nicht möglich, die vollständige ASVS-Prüfung nur mit automatisierten Penetrationstesttools alleine abzudecken. Während eine große Anzahl der Anforderungen in Stufe 1 mit automatisierten Tests überprüft werden kann, ist die überwiegende Mehrheit der Anforderungen nicht für automatisierte Penetrationstests geeignet.

Bitte beachten Sie, dass die Grenzen zwischen automatisierten und manuellen Tests mit zunehmender Reife in der Anwendungssicherheitsindustrie immer mehr verschwimmen. Automatisierte Werkzeuge werden häufig von Experten manuell optimiert, und manuelle Tester nutzen oft eine Vielzahl automatisierter Werkzeuge.

Die Rolle von Penetrationstests

In Version 4.0 haben wir uns entschieden, die Stufe 1 vollständig penetrationstestbar zu machen, ohne Zugriff auf Quellcode, Dokumentation oder Entwickler zu erfordern. Zwei Protokollierungselemente, die zur Einhaltung der OWASP Top 10 2017 A10 erforderlich sind, erfordern Interviews, Screenshots oder eine andere Sammlung von Beweisen, wie sie auch in der OWASP Top 10 2017 erforderlich sind. Das Testen ohne Zugang zu den notwendigen Informationen ist jedoch keine ideale Methode der Sicherheitsüberprüfung, da die Möglichkeit versäumt wird, ein Code-Review zu etablieren, Bedrohungen und fehlende Kontrollen zu identifizieren und einen weitaus gründlicheren Test in kürzerer Zeit durchzuführen.

Wenn möglich, ist bei der Durchführung einer Bewertung nach den Stufen 2 oder 3 der Zugang zu Entwicklern, Dokumentation, Code und Zugriff auf eine Testanwendung mit nicht produktiven Daten erforderlich. Penetrationstests, die auf diesen Ebenen durchgeführt werden, erfordern diese Zugriffsebene, die wir als *Hybride Überprüfungen* oder *Hybride Penetrationstests* nennen.

Andere Verwendungszwecke des ASVS

Abgesehen von der Verwendung zur Bewertung der Sicherheit einer Anwendung haben wir eine Reihe anderer möglicher Anwendungen für den ASVS identifiziert.

Detaillierter Leitfaden für eine Sicherheitsarchitektur

Eine der häufigeren Anwendungen des Application Security Verification Standards ist die Verwendung als Ressource für Sicherheitsarchitekten. In der *Sherwood Applied Business Security Architecture (SABSA)* fehlt eine Vielzahl an Informationen, die für eine gründliche Überprüfung einer sicheren Anwendungsarchitektur erforderlich sind. ASVS kann verwendet werden, um diese Lücken zu füllen, indem Sicherheitsarchitekten bessere Kontrollen für häufige Probleme, wie z.B. Best Practices für Datenschutz und Strategien für Eingabevalidierung, wählen können.

Als Ersatz für Standard Secure Coding Checklisten

Viele Organisationen können von der Übernahme des ASVS profitieren, indem sie sich für eine der drei Stufen entscheiden oder einen *Fork* des ASVS erstellen und die Anforderungen für jede Anwendungsrisikostufe domänenspezifisch ändern. Wir fördern diese Art der Abzweigung, solange die Rückverfolgbarkeit gewährleistet ist. Das heisst, wenn eine Anwendung die Anforderung 4.1 bestanden hat, soll dies für die Verzweigung dasselbe bedeuten wie für den sich weiter entwickelnde Standard.

Als Leitfaden für automatisierte Unit- und Integrationstests

Der ASVS ist so konzipiert, dass er in hohem Maße testbar ist - mit der einzigen Ausnahme der Anforderungen an die Architektur und Vermeidung bösartigen Codes. Durch die Erstellung von Unit- und Integrationstests, die auf spezifische und relevante Fuzzing- und Missbrauchsfälle testen, wird die Anwendung mit jedem einzelnen Build nahezu *selbstverifizierend*. Beispielsweise können zusätzliche Tests für die Testsuite eines Login-Controllers erstellt werden, die den Benutzernamen auf gebräuchliche Standardbenutzernamen, *Account Enumeration*, *Brute-Forcing*, LDAP- und SQL-Injektionen und XSS-Angriffe testen. In ähnlicher Weise sollte ein Test für das Kennwort übliche Standard-Kennwörter, Kennwortlänge, Null-Byte-Injektionen, Entfernen des Kennwortparameters, XSS und mehr umfassen.

Für Trainings zur sicheren Entwicklung

Der ASVS kann auch dazu verwendet werden, Merkmale sicherer Software zu definieren. Viele Trainingskurse zur "sicheren Kodierung" sind einfach ethische Hacking-Kurse mit nur einem kurzen Ausflug zu *Secure Coding*. Dies hilft den Entwicklern aber nicht unbedingt, sichereren Code zu schreiben. Stattdessen können sichere

Entwicklungskurse den ASVS verwenden, wobei der Schwerpunkt hier auf den *ProActive Controls* des ASVS liegt und nicht auf den OWASP Top 10 der Fehler, die man nicht machen sollte.

Als Treiber für die Sicherheit agil entwickelter Anwendungen

Der ASVS kann innerhalb eines agilen Entwicklungsprozesses als Rahmenwerk verwendet werden, um spezifische Aufgaben z.B. innerhalb eines Sprints zu definieren, die vom Team implementiert werden müssen, um ein sicheres Produkt zu erhalten. Ein Ansatz könnte sein: Beginnen Sie mit Stufe 1, prüfen Sie ihre Anwendung oder das System gemäß den ASVS-Anforderungen für die spezifizierte Stufe, finden Sie heraus, welche Kontrollen fehlen und erfassen Sie spezifische Tickets/Aufgaben im Backlog. Dies hilft bei der Priorisierung bestimmter Aufgaben (z.B. beim Grooming/Refinement) und macht die Sicherheit im agilen Prozess sichtbar. Dies kann auch zur Priorisierung von Revisionsprüfungen und Review-Tasks in der Organisation verwendet werden. Das kann bedeuten, dass eine bestimmte ASVS-Anforderung für ein bestimmtes Teammitglied ein Treiber für ein Codereview, ein Refactoring oder eine Auditierung sein kann und als *Technical Debt* im Backlog sichtbar wird, das zeitnah erledigt werden sollte.

Als Rahmen für die Steuerung von Dienstleistern zur Erstellung sicherer Software

ASVS ist ein großartiges Rahmenwerk, um bei der sicheren Softwarebeschaffung oder der Beschaffung von kundenspezifischen Entwicklungsdienstleistungen zu unterstützen. Der Auftraggeber kann einfach die Anforderung stellen, dass die Software, die er beschaffen bzw. beauftragen möchte, auf ASVS-Ebene *X* entwickelt werden muss, und vom Anbieter den Nachweis verlangen, dass die Software die ASVS-Ebene *X* erfüllt. Dies funktioniert optimal, wenn es mit dem Anhang zum OWASP-Vertrag für sichere Software (*OWASP Secure Software Contract Annex*) kombiniert wird.

V1: Anforderungen für Architektur, Design und die Bedrohungsmodellierung

Steuerungsziel

Die Sicherheitsarchitektur ist in vielen Organisationen fast zu einer verschollenen Kunst verkommen. Die Tage des Enterprisearchitekten sind im Zeitalter von DevSecOps größtenteils vorbei. Der Bereich der Anwendungssicherheit muss hier verlorenes Terrain aufholen, agile Sicherheitsprinzipien übernehmen und gleichzeitig anerkannte sichere Architekturprinzipien wieder in die Software-Praxis einführen. Architektur ist nicht die Implementierung, sondern die Art und Weise, über ein Problem nachzudenken, das potenziell viele verschiedene Lösungsmöglichkeiten hat, und nicht nur eine einzige "richtige" Antwort besitzt. Nur allzu oft wird gerade die Sicherheit als unflexibel angesehen und verlangt von den Entwicklern, Code auf eine bestimmte Art und Weise zu reparieren, obwohl die Entwickler vielleicht eine viel bessere Möglichkeit kennen, das Problem zu lösen. Es gibt nicht die eine einfache Lösung für die Architektur, und so zu tun, als ob es nicht so wäre, ist ein Bärendienst für die Softwareentwicklung.

Eine spezifische Implementierung einer Webanwendung wird wahrscheinlich während ihrer gesamten Lebensdauer kontinuierlich überarbeitet, aber die Gesamtarchitektur wird sich wahrscheinlich nur selten ändern, sondern sich nur langsam weiter entwickeln. Die Eine Sicherheitsarchitektur unterscheidet sich hier nicht - wir brauchen heute eine Authentifizierung, wir werden morgen eine Authentifizierung benötigen, und wir werden sie in fünf Jahren brauchen. Wenn wir heute fundierte Entscheidungen treffen, können wir viel Aufwand, Zeit und Geld sparen, wenn wir architekturkonforme Lösungen auswählen und wiederverwenden. Vor einem Jahrzehnt beispielsweise wurde die Multi-Faktor-Authentifizierung nur selten implementiert.

Hätten Entwickler in ein einziges, sicheres Identitätsprovider-Modell investiert, wie z.B. eine föderierte SAML-Identität, könnte der Identitätsprovider einfach aktualisiert werden, um neue Anforderungen wie die NIST 800-63-Konformität einzubeziehen, ohne die Schnittstellen der ursprünglichen Anwendung zu ändern. Immer wenn viele Anwendungen die gleiche Sicherheitsarchitektur und damit die gleiche Komponente nutzen, profitieren sie alle gleichzeitig von diesem Upgrade. SAML wird jedoch nicht immer die beste oder geeignetste Authentifizierungslösung bleiben - es muss möglicherweise gegen andere Lösungen wie z.B. OpenID Connect ausgetauscht werden, wenn sich die Anforderungen ändern. Änderungen wie diese sind entweder kompliziert, so kostspielig, dass eine komplette Neuentwicklung erforderlich ist, oder ohne eine Sicherheitsarchitektur schlichtweg unmöglich.

In diesem Kapitel behandelt der ASVS die wichtigsten Aspekte jeder soliden Sicherheitsarchitektur:

- Verfügbarkeit
- Vertraulichkeit
- Verarbeitungsintegrität
- Nichtabstreitbarkeit
- Datenschutz

Alle dieser Sicherheitsprinzipien müssen als Bestandteil in allen Anwendungen integriert sein. Ein *Shift-Left* ist entscheidend, beginnend mit der Befähigung der Entwickler durch Checklisten für die sichere Kodierung, Mentoring und Schulungen, Kodierung und Tests, Aufbau, Bereitstellung, Konfiguration und Betrieb und endend mit unabhängigen Tests, um sicherzustellen, dass alle Sicherheitskontrollen vorhanden und funktionsfähig sind. Früher war der letzte Testschritt alles, was wir als Industrieunternehmen umgesetzt haben, aber das reicht nicht mehr aus, wenn Entwickler zehn- oder hundertmal am Tag Code in die Produktion schieben. Anwendungssicherheitsexperten müssen mit agilen Techniken Schritt halten. Das bedeutet, dass sie Entwicklerwerkzeuge einsetzen, den Code lernen und mit den Entwicklern zusammenarbeiten müssen, anstatt das Projekt monatelang mit Sicherheitsblockaden zu überhäufen, während alle anderen Projektbeteiligten längst weiter gearbeitet haben.

V1.1 Anforderungen an einen sicheren Softwareentwicklungslebenszyklus

#	Beschreibung	L1	L2	L3	CWE
1.1.1	Überprüfen Sie die Verwendung eines sicheren Softwareentwicklungslebenszyklus, der die Sicherheit in allen Entwicklungsphasen berücksichtigt. (C1)		✓	✓	
1.1.2	Überprüfen Sie bei jeder Designänderung oder Sprintplanung den Einsatz von Bedrohungsmodellen (*threat models*), um Bedrohungen zu identifizieren, Gegenmaßnahmen zu planen, angemessene Reaktionen auf Risiken zu erleichtern und Sicherheitstests anzuleiten.		✓	✓	1053
1.1.3	Vergewissern Sie sich, dass alle Userstories und Features funktionale Sicherheitseinschränkungen enthalten, wie z.B. *Als Benutzer sollte ich mein Profil anzeigen und bearbeiten können. Ich sollte nicht in der Lage sein, das Profil anderer Personen anzuzeigen oder zu bearbeiten*.				
	✓	✓	1110		
1.1.4	Überprüfen Sie die Dokumentation und Begründung aller Vertrauensgrenzen, Komponenten und wichtigen Datenflüsse der Anwendung.		✓	✓	1059
1.1.5	Überprüfen Sie die Definition und Sicherheitsanalyse der High-Level-Architektur der Anwendung und aller angeschlossenen Remotedienste. (C1)		✓	✓	1059
1.1.6	Überprüfen Sie die Implementierung zentralisierter, einfacher (Wirtschaftlichkeit des Designs), geprüfter, sicherer und wiederverwendbarer Sicherheitskontrollen, um doppelte, fehlende, unwirksame oder unsichere Kontrollen zu vermeiden. (C10)		✓	✓	637
1.1.7	Überprüfen Sie die Verfügbarkeit einer Checkliste für sichere Kodierung, Sicherheitsanforderungen, von Leitfäden oder Richtlinien für alle Entwickler und Tester.		✓	✓	637

V1.2 Anforderungen an die Authentifizierungsarchitektur

Beim Entwurf der Authentifizierung spielt es keine Rolle, ob Sie über eine starke, hardwarebasierte Multi-Faktor-Authentifizierung verfügen, wenn ein Angreifer gleichzeitig ein Konto zurücksetzen kann, indem er

lediglich ein Callcenter anruft und allgemein bekannte Fragen beantwortet. Beim Identitätsnachweis müssen **alle** Authentifizierungswege die gleiche Stärke aufweisen.

#	Beschreibung	L1	L2	L3	CWE
1.2.1	Überprüfen Sie, ob die Kommunikation zwischen Anwendungskomponenten, einschließlich APIs, Middleware und Datenschichten, authentifiziert ist und individuelle Benutzerkonten verwendet. (C3)		✓	✓	306
1.2.2	Vergewissern Sie sich, dass die Anwendung einen einzigen geprüften Authentifizierungsmechanismus verwendet, der bekanntermaßen sicher ist, auf eine stärkere Authentifizierung ausgedehnt werden kann und über eine ausreichende Protokollierung und Überwachung verfügt, um Kontomissbrauch oder -verletzungen zu erkennen.				
	✓	✓	306		
1.2.3	Vergewissern Sie sich, dass alle Authentifizierungspfade und Identitätsmanagement-APIs eine einheitliche Stärke der Authentifizierungssicherheitskontrolle implementieren, so dass es keine schwächeren Alternativen für das Risiko der Anwendung gibt.				
		✓	✓	306	

V1.3 Architektonische Anforderungen an das Sitzungsmanagement

Dies dient als ein Platzhalter für zukünftige architektonische Anforderungen.

V1.4 Anforderungen an die Architektur der Zugriffskontrolle

#	Beschreibung	L1	L2	L3	CWE
1.4.1	Vergewissern Sie sich, dass vertrauenswürdige Durchsetzungspunkte, z.B. an Gateways für Zugangskontrolle, Servern und *serverlosen* Funktionen, die Zugangskontrollen durchsetzen. Setzen Sie niemals Zugriffskontrollen nur auf dem Client durch.		✓	✓	602
1.4.2	Überprüfen Sie, ob die gewählte Lösung zur Zugangskontrolle flexibel genug ist, um die Anforderungen der Anwendung zu erfüllen.		✓	✓	284
1.4.3	Überprüfen Sie die Durchsetzung des Prinzips der wenigsten erforderlichen Privilegien (*least privilege*) bei Funktionen, Dateien, URLs, Controllern, Diensten und anderen Ressourcen. Dies impliziert den Schutz vor Spoofing und die Erweiterung von Privilegien.		✓	✓	272
1.4.4	Vergewissern Sie sich, dass die Kommunikation zwischen Anwendungskomponenten, einschließlich APIs, Middleware und Datenschichten, mit den am wenigsten erforderlichen Berechtigungen (*least privilege*) durchgeführt wird.				
	(C3)	✓	✓	272	
1.4.5	Überprüfen Sie, dass die Anwendung einen einzigen und gut erprobten Zugriffskontrollmechanismus für den Zugriff auf geschützte Daten und Ressourcen verwendet. Alle Anfragen müssen diesen einzigen Mechanismus durchlaufen, um Kopieren und Einfügen oder unsichere Alternativpfade zu vermeiden.				
	(C7)	✓	✓	284	

1.4.6	Stellen Sie sicher, dass eine attribut- oder merkmalsbasierte Zugriffskontrolle verwendet wird, wobei der Code die Berechtigung des Benutzers für ein Merkmal/Datenelement und nicht nur seine Rolle prüft. Die Berechtigungen sollten weiterhin über Rollen zugewiesen werden. (C7)		✓	✓	275

V1.5 Anforderungen an die Eingabe- und Ausgabe-Architektur

In Version 4.0 haben wir uns von dem Begriff *server-seitig* als etablierte Vertrauensgrenze verabschiedet. Die Vertrauensgrenze ist immer noch ein Thema - Entscheidungen über nicht vertrauenswürdige Browser oder Client-Geräte zu treffen, können weiterhin umgangen werden. In den heutigen Standardarchitekturen hat sich jedoch der Punkt der Vertrauensdurchsetzung (*Trust Enforcement*) dramatisch verändert. Wenn in dem ASVS der Begriff *vertrauenswürdige Serviceschicht* verwendet wird, meinen wir daher jeden vertrauenswürdigen Durchsetzungspunkt, unabhängig vom Standort, wie z.B. einen Microservice, eine *serverless* API, eine serverseitige, eine vertrauenswürdige API auf einem Clientgerät, usw.

#	Beschreibung	L1	L2	L3	CWE
1.5.1	Vergewissern Sie sich, dass die Eingabe- und Ausgabeanforderungen klar definieren, wie die Daten auf der Grundlage des Typs, des Inhalts und der anwendbaren Gesetze, Vorschriften und anderen Richtlinien zu behandeln und zu verarbeiten sind.				
	✓	✓	1029		
1.5.2	Stellen Sie sicher, dass bei der Kommunikation mit nicht vertrauenswürdigen Clients keine Serialisierung verwendet wird. Wenn dies nicht möglich ist, stellen Sie sicher, dass angemessene Integritätskontrollen (und möglicherweise Verschlüsselung, wenn sensible Daten gesendet werden) durchgesetzt werden, um Deserialisierungsangriffe einschließlich der Injektion von Objekten zu verhindern.		✓	✓	502
1.5.3	Vergewissern Sie sich, dass die Eingabevalidierung auf einer vertrauenswürdigen Serviceschicht durchgesetzt wird. (C5)		✓	✓	602
1.5.4	Vergewissern Sie sich, dass die Ausgabekodierung in der Nähe des oder durch den Interpreter erfolgt, für den sie bestimmt ist. (C4)		✓	✓	116

V1.6 Anforderungen an die kryptographische Architektur

Anwendungen müssen mit einer Architektur für eine starke Kryptographie entworfen werden, um die Datenbestände gemäß ihrer Klassifizierung zu schützen. Alles zu verschlüsseln ist verschwenderisch, nichts zu verschlüsseln ist rechtlich fahrlässig. Es muss ein Gleichgewicht gefunden werden, in der Regel während des Architektur- oder High-Level-Designs, bei Design-Sprints oder Architektur-Spikes. Wenn Sie die Kryptographie nach und nach entwerfen oder nachrüsten, wird die sichere Implementierung unweigerlich viel mehr kosten, als wenn Sie sie gleich von Anfang an einbauen würden.

Architektonische Anforderungen sind der gesamten Codebasis inhärent und daher schwierig zu vereinheitlichen oder zu integrieren. Die architektonischen Anforderungen müssen bei den Kodierungsstandards während der gesamten Umsetzungsphase berücksichtigt werden und sollten während der Sicherheitsarchitektur, bei Peer- oder Code-Reviews oder bei Retrospektiven überprüft werden.

#	Beschreibung	L1	L2	L3	CWE
1.6.1	Vergewissern Sie sich, dass es eine explizite Richtlinie für die Verwaltung von kryptografischen Schlüsseln gibt und dass der Lebenszyklus eines kryptografischen Schlüssels einem Schlüsselverwaltungsstandard wie NIST SP 800-57 folgt.		✓	✓	320

1.6.2	Stellen Sie sicher, dass Verwender von kryptografischen Diensten Schlüsselmaterial und andere Geheimnisse schützen, indem Sie Key Vaults oder API-basierte Alternativen verwenden.		✓	✓	320
1.6.3	Vergewissern Sie sich, dass alle Schlüssel und Passwörter austauschbar sind und Teil eines genau definierten Prozesses zur Neuverschlüsselung sensibler Daten sind.		✓	✓	320
1.6.4	Vergewissern Sie sich, dass symmetrische Schlüssel, Kennwörter oder API-Geheimnisse, die von Kunden erzeugt oder mit ihnen geteilt werden, nur zum Schutz von Geheimnissen mit geringem Risiko, wie z.B. die Verschlüsselung des lokalen Speichers, oder für vorübergehende, kurzzeitige Verwendungen wie die Verschleierung von Parametern, verwendet werden. Das Teilen von Geheimnissen mit Kunden ist gleichzusetzen mit dem Teilen von Klartext und sollte auch architektonisch als solches behandelt werden.				
	✓	✓	320		

V1.7 Architektonische Anforderungen an Fehlerbehandlung, Protokollierung und Auditierung

#	Beschreibung	L1	L2	L3	CWE
1.7.1	Vergewissern Sie sich, dass im gesamten System ein einheitliches Protokollformat und ein einheitlicher Protokollierungsansatz verwendet wird. (C9)		✓	✓	1009
1.7.2	Vergewissern Sie sich, dass die Protokolle sicher an ein, vorzugsweise entferntes, System zur Analyse, Erkennung, Alarmierung und Eskalation übertragen werden.				
(C9)		✓	✓		

V1.8 Architektonische Anforderungen für Datenschutz und Privatsphäre

#	Beschreibung	L1	L2	L3	CWE
1.8.1	Vergewissern Sie sich, dass alle sensiblen Daten identifiziert und in Schutzstufen eingeteilt werden.		✓	✓	
1.8.2	Vergewissern Sie sich, dass alle Schutzebenen mit einer Reihe von Schutzanforderungen verbunden sind, z.B. Verschlüsselungsanforderungen, Integritätsanforderungen, Aufbewahrung, Datenschutz und andere Vertraulichkeitsanforderungen, und dass diese in der Architektur angewendet werden.		✓	✓	

V1.9 Architektonische Anforderungen an die Kommunikation

#	Beschreibung	L1	L2	L3	CWE
1.9.1	Überprüfen Sie, ob die Anwendung die Kommunikation zwischen Komponenten verschlüsselt, insbesondere wenn sich diese Komponenten in verschiedenen Containern, Systemen, Standorten oder Cloud-Anbietern befinden. (C3)		✓	✓	319
1.9.2	Überprüfen Sie, dass die Anwendungskomponenten die Authentizität jeder Seite in einer Kommunikationsverbindung überprüfen, um "Person-in-the-Middle"-Angriffe zu verhindern. Beispielsweise sollten Anwendungskomponenten TLS-Zertifikate und -Ketten validieren.		✓	✓	295

V1.10 Architekturanforderungen hinsichtlich bösartiger Software

#	Beschreibung	L1	L2	L3	CWE
1.10.1	Vergewissern Sie sich, dass ein Quellcode-Kontrollsystem verwendet wird, mit Verfahren, die sicherstellen, dass beim Einchecken zugehörige Tickets referenziert werden. Das Quellcode-Kontrollsystem sollte über eine Zugriffskontrolle und identifizierbare Benutzer verfügen, um die Rückverfolgbarkeit von Änderungen zu ermöglichen.		✓	✓	284

V1.11 Architekturanforderungen an die Geschäftslogik

#	Beschreibung	L1	L2	L3	CWE
1.11.1	Überprüfen Sie die Definition und Dokumentation aller Anwendungskomponenten in Bezug auf die von ihnen bereitgestellten Geschäfts- oder Sicherheitsfunktionen.		✓	✓	1059
1.11.2	Vergewissern Sie sich, dass alle kritischen Abläufe der Geschäftslogik, einschließlich Authentifizierung, Sitzungsverwaltung und Zugriffskontrolle, keinen unsynchronisierten Zustand aufweisen.		✓	✓	362
1.11.3	Vergewissern Sie sich, dass alle kritischen Abläufe der Geschäftslogik, einschließlich Authentifizierung, Sitzungsverwaltung und Zugriffskontrolle, thread-sicher und resistent gegen Prüfzeit- und Nutzungszeit-Nebenläufigkeiten sind.			✓	367

V1.12 Architekturanforderungen an sicheres Hochladen von Dateien

#	Beschreibung	L1	L2	L3	CWE
1.12.1	Überprüfen Sie, ob die vom Benutzer hochgeladenen Dateien außerhalb des Web-Stammverzeichnisses gespeichert sind.		✓	✓	552
1.12.2	Vergewissern Sie sich, dass die vom Benutzer hochgeladenen Dateien - falls sie angezeigt oder von der Anwendung heruntergeladen werden müssen - entweder durch Octet-Stream-Downloads oder von einer nicht verwandten Domäne, wie z.B. einem Cloud File Storage Bucket, bereitgestellt werden. Implementieren Sie eine geeignete Content Security Policy (CSP), um das Risiko von XSS-Vektoren oder anderen Angriffen auf die hochgeladene Datei zu reduzieren.		✓	✓	646

V1.13 Architekturanforderungen für APIs

Dies dient als Platzhalter für zukünftige Architekturanforderungen.

V1.14 Architekturanforderungen für die Konfiguration

#	Beschreibung	L1	L2	L3	CWE
1.14.1	Überprüfen Sie die Verwendung eindeutiger oder spezieller Betriebssystemkonten mit niedriger Privilegierung für alle Anwendungskomponenten, Dienste und Server.				
(C3)		✓	✓	250	
1.14.2	Überprüfen Sie die Trennung von Komponenten unterschiedlicher Vertrauensebenen durch klar definierte Sicherheitskontrollen, Firewall-Regeln, API-Gateways, Reverse-Proxies, Cloud-basierte Sicherheitsgruppen				

	oder vergleichbare Mechanismen.				
	✓	✓	923		
1.14.3	Stellen Sie sicher, dass bei der Bereitstellung von Binärdateien auf nicht vertrauenswürdigen Geräten binäre Signaturen, vertrauenswürdige Verbindungen und verifizierte Endpunkte verwendet werden.		✓	✓	494
1.14.4	Vergewissern Sie sich, dass die Build-Pipeline vor veralteten oder unsicheren Komponenten warnt und entsprechende Maßnahmen ergreift.				
	✓	✓	1104		
1.14.5	Vergewissern Sie sich, dass die Build-Pipeline einen Build-Schritt enthält, um die sichere Bereitstellung der Anwendung automatisch zu erstellen und zu verifizieren, insbesondere wenn es sich bei der Anwendungsinfrastruktur um Software handelt, wie z.B. Build-Skripts für Cloud-Umgebungen.		✓	✓	
1.14.6	Vergewissern Sie sich, dass Anwendungsimplementierungen auf der Netzwerkebene angemessen sandboxed, containerisiert und/oder isoliert sind, um Angriffe zu verzögern und Angreifer davon abzuhalten, andere Anwendungen anzugreifen, insbesondere wenn sie sensible oder gefährliche Aktionen wie Deserialisierung durchführen. (C5)		✓	✓	265
1.14.7	Stellen Sie sicher, dass die Anwendung keine nicht mehr unterstützte, unsicheren oder veralteten clientseitigen Technologien wie NSAPI-Plugins, Flash, Shockwave, ActiveX, Silverlight, NACL oder clientseitige Java-Applets verwendet.		✓	✓	477

Referenzen

Für weitere Informationen siehe auch:

- OWASP Threat Modeling Cheat Sheet
- OWASP Attack Surface Analysis Cheat Sheet
- OWASP Threat modeling
- OWASP Secure SDLC Cheat Sheet
- Microsoft SDL
- NIST SP 800-57