Safuat Hamdy

Der OWASP Application Security Verification Standard

Ein Praxisbericht

Verschiedene Normen, Gesetze und Verträge verlangen, dass Informationen während der Verarbeitung sowie in der Ablage angemessen geschützt werden. Doch was bedeutet jeweils „angemessen"? Eine Vorgehensweise nach allgemeiner Best-Practice ist oft zu unscharf, während eine Vorgehensweise etwa nach *Common Criteria* in vielen Fällen weit über das Ziel hinaus schießen würde. Für Webanwendungen bietet sich eine Vorgehensweise nach dem *OWASP Application Security Verification Standard* (ASVS) an. Der Beitrag stellt die praktischen Erfahrungen des Autors vor.

1 Problemstellung

Inzwischen werden viele Geschäftsprozesse über Webanwendungen und Webportale abgewickelt. Ob Online-Banking, Online-Shopping, Online-Medien oder Soziale Netzwerke – alle diese Anwendungen beruhen darauf, dass der Nutzer über seinen Browser und eine Internet-Verbindung mit einem Anwendungsserver kommuniziert.

Ein Ansatz zum Nachweis der Sicherheit dieser Anwendung liegt in einer Untersuchung auf Schwachstellen, d. h. eine Anwendung wird beispielsweise auf die Schwachstellen nach OWASP Top 10 abgeklopft. Der Nachteil bei dieser Vorgehensweise ist, dass nie ein vollständiger Nachweis erbracht werden kann. Werden keine Schwachstellen gefunden, dann kann das daran liegen, dass der Tester nicht lange genug gesucht hat oder nicht erfahren genug ist.

Eine Quellcodeanalyse kann hier mehr Licht ins Dunkel bringen, ist aber aufwändig und oft nicht möglich, weil der Quellcode beispielsweise nicht offengelegt ist oder weil der Quellcode so mit komplexen Frameworks verzahnt ist, dass eine genaue Untersuchung aus wirtschaftlicher Sicht zu aufwändig wäre. Umfangreicher Quellcode entzieht sich aus demselben Grund ebenfalls einem manuellen Review.

Dr. Safuat Hamdy

ist Security Consultant bei der Secorvo Security Consulting GmbH. Schwerpunkte: Sicherheits- und Webanwendungsaudits, sichere Softwareentwicklung und forensische Analysen.
E-Mail: safuat.hamdy@secorvo.de

Auch der Einsatz von Analysetools bietet nur mäßige Erleichterung. Klassische Schwachstellenscanner konzentrieren sich auf das Eingabeverhalten der untersuchten Anwendung, können aber Fehler in der Anwendungslogik prinzipiell nicht oder nur rudimentär erkennen. Darüber hinaus können sich auch hier komplexe Frameworks, etwa auf AJAX-Grundlage, als Stolperstein herausstellen. Dasselbe gilt für automatisierte Quellcodescanner. Wenn der eingesetzte Quellcodescanner die verwendeten Frameworks nicht „versteht", dann können keine sinnvollen Analyseergebnisse erwartet werden.

Anstelle der Suche nach Schwachstellen bietet sich eine Suche nach Indizien für wirksam implementierte Sicherheitsmechanismen an.

2 Der ASVS

Der OWASP *Application Security Verification Standard* 2009 (im Folgenden kurz: ASVS) bietet den Rahmen für eine Bewertung der Sicherheit einer Anwendung. Auch hier bilden lediglich Indizien die Grundlage der Bewertung. Anders als bei einem klassischen Penetrationstest wird hier jedoch nicht nach ggf. schwer zu findenden Schwachstellen gesucht, sondern nach relativ einfach nachweisbaren Merkmalen.

Da auch eine erfolgreiche Prüfung nach ASVS keinen Beweis für die Sicherheit der Anwendung liefert, ist es natürlich möglich, dass auch eine ASVS-konforme Anwendung Schwachstellen enthält, denn wie jedes Audit ist auch eine Prüfung nach ASVS nur eine – wenn auch systematische – Stichprobe. Die Gründlichkeit, mit der die einzelnen Prüfpunkte abgearbeitet werden, wird durch den abgestimmten Zeit- und Budgetrahmen beschränkt. Es geht aber auch hier nicht darum, eine perfekte Sicherheit nachzuweisen, sondern mit angemessenem Aufwand abzuschätzen und zu dokumentieren, welches Sicherheitsniveau eine Anwendung erreicht.

Die derzeit gültige Version des ASVS stammt aus dem Jahr 2009 und befindet sich derzeit in Überarbeitung.

2.1 ASVS im Detail

Den Kern des ASVS bilden 14 Prüfgebiete (*Verification Requirements*), die jeweils eine Reihe von Prüfpunkten enthalten. Es werden im Rahmen einer Prüfung jedoch nicht pauschal alle Prüfpunkte abgearbeitet. Stattdessen wird beim ASVS über einen Prüflevel die Prüftiefe definiert, so dass mit steigender Prüftiefe auch das Vertrauen in die Sicherheit einer Anwendung steigt. Rein konzeptionell ist dies mit den *Evaluation Assurance Levels* (EAL) der *Common Criteria* vergleichbar – je höher der Prüflevel, desto höher die Zusicherung an die Sicherheit, aber auch desto höher der Prüfaufwand.

Der ASVS unterschiedet vier Level 1 bis 4, die hierarchisch angeordnet sind, d. h. die Prüfpunkte eines Levels sind in allen höheren Leveln vollständig enthalten. Der ASVS charakterisiert die einzelnen Level folgendermaßen [3]:

- Level 1 – *Automated Verification*: Dieser Level ist angemessen für Anwendungen, bei denen geringfügige Anforderungen an die Sicherheit gestellt werden
- Level 2 – *Manual Verification*: Dieser Level ist angemessen für Anwendungen, die beispielsweise personenbezogene Daten oder Kreditkartentransaktionen verarbeiten. Hierunter fallen viele typische Webanwendungen.
- Level 3 – *Design Verification:* Dieser Level ist angemessen für Anwendungen, die besonders sensitive Daten verarbeiten, etwa Patientendaten.
- Level 4 – *Internal Verification*: Dieser Level ist angemessen für Anwendungen, die kritische Infrastrukturen oder lebenserhaltende Systeme steuern.

Die Level 1 und 2 sind jeweils noch in die Unterlevel 1A und 1B bzw. 2A und 2B geteilt. Ein Bestehen nach diesen Leveln stellt ein jeweils schwächeres Ergebnis dar, als ein Bestehen nach den Leveln 1 bzw. 2.

Die Level 3 und 4 sind in der Praxis eher selten anzutreffen. Erstens ist der Prüfaufwand zum Bestehen nach diesen Leveln signifikant höher, als etwa für Level 2, lohnt sich also nur für besonders kritische Anwendungen. Zweitens muss eine Anwendung zum Bestehen nach Level 3 oder 4 bereits vom Design her an den entsprechenden Anforderungen nach ASVS entwickelt werden. Eine ad-hoc-Prüfung nach ASVS ist nur bis zu Level 2 sinnvoll. Auch dies ist in ähnlicher Form bereits von den *Common Criteria* her bekannt.

Das Plus von Level 3 im Vergleich zu Level 2 besteht vor allem in einer Prüfung der Sicherheitsarchitektur der Anwendung. Dennoch ist der Einschluss von einigen Prüfpunkten aus Level 3 durchaus betrachtenswert (siehe Abschnitt 3), während die Prüfpunkte aus Level 4 im Wesentlichen darauf abzielen, die Schadcodefreiheit der Anwendung nachzuweisen.

Eine weitere Parallele zu den *Common Criteria* besteht darin, dass der ASVS definiert *was* zu prüfen ist, aber nicht *wie*. Hier kann der OWASP *Testing Guide* [4] teilweise eine Handreichung geben; in einem Blog von Herman Stevens wurde der (unvollständige) Versuch einer Abbildung gemacht [6].

Ein weiteres Element des ASVS ist eine Definition des Prüfberichts, die sich jedoch nicht grundlegend von anderen formalen Anforderungen an vergleichbare Prüfberichte unterscheidet. Für die Erstellung ASVS-konformer Berichte stellt OWASP eine Berichtsvorlage zur Verfügung.

Anders als etwa für *Common Critera* gibt es kein Zertifikat, das die Konformität zum ASVS bescheinigt. Zum einen, weil es keine akkreditierten Prüfungsorgane gibt (und dies seitens OWASP auch nicht vorgesehen ist). Zum anderen halte ich dies inhaltlich nicht für sinnvoll: Da sich eine Prüfung immer auf ein bestimmtes Release bezieht, wäre eine ASVS-Zertifizierung von häufig wechselnden Releases von Webanwendungen wirtschaftlich nicht darstellbar. Damit stellt sich aber auch die Frage nach dem Sinn einer ASVS-Prüfung, da jedes Prüfergebnis nach einem Release-Wechsel streng genommen obsolet wird. Selbst wenn alle Sicherheitsmechanismen beim Release-Wechsel unberührt bleiben, kann eine Änderung oder Erweiterung der Anwendungslogik zu Schwachstellen führen, die vorher nicht vorhanden waren.

Zum Nachweis einer *nachhaltig* sicheren Softwareentwicklung sollten hierfür ergänzend die Entwicklungsprozesse überprüft werden [2], also etwa, ob die Entwickler die notwendige Kompetenz haben und ob im *Systems Development Life-Cycle* (SDLC) Sicherheit angemessen berücksichtigt wird. Aber prozessorientierte Normen, wie etwa die ISO 27001, verlangen auch, dass regelmäßig eine Messung der Wirksamkeit der ergriffenen Maßnahmen vorgenommen wird, und dies kann im Fall von Webanwendungen bedeuten, eine Anwendungsprüfung nach ASVS durchzuführen.

Da Sicherheit kein Zustand ist, sondern ein kontinuierlicher Prozess, lässt sich somit festhalten, dass eine einzelne Prüfung nach ASVS ohne weiteren Kontext nur als eine Station in dem Sicherheitsprozess verstanden werden sollte. Ist noch kein solcher Prozess etabliert, dann bietet sich eine ASVS-Prüfung als Ausgangspunkt für die Initiierung eines solchen Prozesses an.

3 Vorgehensweise

Typischerweise werden vor allem Prüfungen für bestehende Anwendungen nach Level 1A und 2A nachgefragt und durchgeführt, da eine Prüfung nach Level 3 oder 4 für bestehende Anwendungen häufig ohne Weiteres nicht möglich ist – zumindest wenn ein Bestehen nach ASVS das Ziel ist. Die Haupthindernisse zum Vorgehen nach Level 1B oder 2B sind die Verfügbarkeit des Quellcodes der zu untersuchenden Anwendung sowie der Aufwand für die dafür benötigten Analysetools.

Der Aufwand für eine Prüfung nach Level 1A oder 2A ist relativ moderat. Für eine Prüfung nach Level 2A sind insgesamt 56 Prüfpunkte in 11 der 14 Prüfgebiete zu untersuchen; für eine Prüfung nach Level 1A sind nur 22 Prüfpunkte aus 9 Prüfgebieten zu untersuchen. Würde man eine vollständige Prüfung nach Level 2 durchführen, dann wären insgesamt 92 Prüfpunkte aus 12 Prüfgebieten zu bearbeiten.

Einige der Prüfpunkte aus Level 2B können auch ohne Einblick in den Quellcode geprüft werden; dafür würde beispielsweise ein Interview mit dem verantwortlichen Anwendungsadministrator ausreichen. Beispiele hierfür sind *Verify that account passwords are salted […] and hashed before storing.* (V 2.13) oder *Verify that password hashes are salted when they are created* (V 7.4).

Selbst der Einschluss einzelner Prüfpunkte aus Level 3 ist u. U. relativ leicht möglich: Anforderungen wie *Verify that all input data ist canonicalized […] prior to validation* (V 5.8) ist in jedem Fall sinnvoll, wenn der Quellcode vorliegt.

Der ASVS sieht durchgehend eine Unterstützung durch Tools bei der Untersuchung vor. Welche Tools und wie diese eingesetzt werden, überlässt der ASVS dem Prüfer, entscheidend ist hier in erster Linie die Nachvollziehbarkeit. Ich selbst neige mittlerweile dazu, für Prüfungen nach Level 2B oder 2 auf Quellcodescanner zu verzichten und stattdessen die neuralgischen Codestellen manuell zu begutachten. Damit soll nicht von dem Einsatz von Quellcodescannern abgeraten werden; wichtig ist, dass der Einsatz solcher Werkzeuge im Entwicklungsprozess (und hier im QA-Prozess) verankert sein muss – eine ad-hoc-Verwendung solcher Tools ist meist inhaltlich nicht sinnvoll und wirtschaftlich nicht darstellbar.

Rein konzeptionell besteht der wesentliche Unterschied zwischen Level 1A und 2A laut ASVS darin, dass bei Level 1A ein automatisierter Schwachstellenscan stattfindet, dessen Ergebnisse manuell verifiziert werden, während für Level 2A aktiv eine manuelle Prüfung stattfindet. Dies ist auch dem Umstand geschuldet, dass die Prüfpunkte für Level 2 bzw. 2A teilweise überhaupt nicht oder nicht gut automatisiert überprüft werden können.

Das größte praktische Hindernis bestand am Anfang meiner Praxis darin, die einzelnen Prüfpunkte mit Leben zu füllen, d. h. exakte Kriterien festzulegen, wann ein Punkt erfüllt ist und wann nicht. Für zahlreiche Prüfpunkte ist das Testkriterium mehr oder weniger offensichtlich, teilweise leistet hier auch der OWASP *Testing Guide* [4] Hilfestellung, wobei auch da eine gewisse Transferleistung erbracht werden muss. Einige Prüfpunkte sind leider nicht immer unmittelbar verständlich formuliert (siehe folgender Abschnitt), so dass ich vor allem am Anfang gelegentlich die ASVS-Mailingliste bemühen musste. Glücklicherweise sind auch einige ASVS-Autoren auf der Mailingliste aktiv, die mich stets mit nützliche Hinweisen und Erklärungen versorgt haben.

Für die Untersuchung reicht ein erweiterbarer Browser wie Mozilla Firefox oder Google Chrome mit Erweiterungen wie Web Developer, einem guten Cookie Manager und verschiedenen weiteren Add-Ons, sowie ein Proxy wie Burp oder ZAP. Zusätzlich haben sich Tools wie wget, curl oder der openssl s_client als nützlich erwiesen. Hin und wieder kommt man auch um das Schreiben eines Skripts nicht herum, um einen Prüfschritt zu automatisieren. Selbst der Einsatz von Metasploit wäre hier denkbar, wurde von mir aber noch nicht erprobt.

Zur Nachvollziehbarkeit der Untersuchung wurden alle Prüfschritte protokolliert. Die bisherigen Erfahrungen mit ASVS-Prüfungen haben gezeigt, dass für jeden Prüfpunkt ein eigenes Proxy-Protokoll angelegt werden sollte. Proxys wie Burp erlauben es, den Zustand einer Proxy-Sitzung abzuspeichern und später wieder aufzunehmen. Der Vorteil dieser Vorgehensweise liegt darin, dass man in dem Protokoll nicht suchen muss, welcher Schritt zu welchem Prüfpunkt gehört. Wird ein Screenshot gemacht, um die Wirkung eines Schrittes zu dokumentieren, dann sollte ein Bezug zu dem Eintrag im entsprechenden Proxy-Protokoll gemacht werden, etwa durch eine entsprechende Benennung der Datei.

Aus der praktischen Erfahrung erweist es sich als sinnvoll, sich zunächst einen Überblick über die zu prüfende Anwendung (in ASVS-Terminologie als „TOV" – Target of Verification – bezeichnet) zu verschaffen und parallel dazu einen konkreten Auditplan zu erstellen. Bei Portalen, die mehrere Anwendungen vereinigen, sollte darauf geachtet werden, dass der Scope der Untersuchung nicht verlassen wird.

In der Explorationsphase (in der Literatur gelegentlich auch als „Abtastung" bezeichnet) sollte darauf geachtet werden, wo Eingaben (auch versteckte Eingaben) und Parameter übergeben werden können. Für Anwendungen, die eine Benutzeranmeldung erfordern, sollte mithilfe des Cookie-Managers und anderer geeigneter Tools eruiert werden, welcher Session-Mechanismus verwendet wird. Bei einigen von mir untersuchten Anwendungen hat sich das als nicht trivial zu beantwortende Frage herausgestellt, da dort Mischformen aller möglichen Mechanismen im Einsatz waren. In einem Fall war der Session-Mechanismus derart komplex, dass, obwohl keine konkrete Schwachstelle zu finden war, dies als Finding berichtet wurde. Auch diese Explorationsphase sollte aufgezeichnet werden, da in der Praxis gelegentlich Auffälligkeiten auftraten, die später nicht mehr rekonstruiert werden konnten.

Im nächsten Schritt wendet man sich der konkreten Überprüfung der einzelnen Erfordernisse des ASVS zu. Hierbei ist es wichtig, dass der Betreiber der Anwendung eventuell vorgeschaltete *Web Application Firewalls* (WAFs) oder *Intrusion Detection/Prevention Systems* (IDS/IPS) abschaltet, da andernfalls die vorgelagerten Schutzmechanismen anstelle der Anwendung geprüft werden, darüber hinaus vermeidet man damit, bei der Abarbeitung der Prüfpunkte ausgelöste Sperrungen der IP-Adresse, von der aus die Prüfung erfolgt, aufheben oder ablaufen zu lassen. Daher ist es auch bei einer ASVS-Prüfung sinnvoll, ein Testsystem zur Verfügung zu haben.

4 Kritik am ASVS

Einige Kritikpunkte sind bereits im vorigen Abschnitt angerissen worden. Sie stellen nicht den ASVS an sich in Frage; die Kritik dreht sich eher um Formulierungen oder den Sinn, die Durchführbarkeit und die Zuordnung einzelner Prüfpunkte.

Der ASVS ist ein Dokument, das sich noch in der Entwicklung befindet. Der Standard wurde 2009 das erste Mal veröffentlicht, und nach drei Jahren Erfahrung in der Praxis haben sich verschiedene Punkte ergeben, die einer Überarbeitung bedürfen. Das ist ein ganz normaler Prozess. Trotz der Kritik an einzelnen Punkten besteht jedoch kein Zweifel daran, dass der ASVS grundsätzlich einen wertvollen Beitrag zur Anwendungssicherheit leisten kann. Im Folgenden gehe ich auf verschiedene Kritikpunkte im Detail ein.

Einige Prüfpunkte sind so unklar formuliert, dass man zumindest als Einsteiger Schwierigkeiten hat, die entsprechenden Punkte zu prüfen. Ein Beispiel hierfür ist *Verifiy that all server side errors are handled on the server* (V 8.2). Als ich mich zu Beginn meiner Praxis das erste Mal mit diesem Punkt auseinander setzen musste, fragte ich mich sofort, wo ein serverseitiger Fehler denn sonst behandelt werden soll? Eine Nachfrage auf der Mailingliste wurde damit beantwortet, dass die Formulierung tatsächlich etwas verunglückt sei: Gemeint ist mit dem Prüfpunkt, dass alle Fehlerbedingungen auf dem Server *angemessen* behandelt werden; die Formulierung legt das aber nicht unbedingt nahe. Ich muss gestehen, dass ich im Nachhinein nicht immer nachvollziehen kann, warum ich einen Punkt unklar fand, erinnere mich aber an teilweise längere Diskussionen mit Kollegen über verschiedene Punkte. Für Einsteiger wäre hier ein Kommentar zum ASVS hilfreich, der zu jedem Prüfpunkt beschreibt, was gemeint ist, und ggf. eine Prüfstrategie skizziert.

Andere Anforderungen wie *Verify that HTTP headers in requests […] contain only printable ASCII characters* (V 11.6) sind schlicht sinnlos; Punkte wie *Verify that all authentication controls fail securely* (V 2.6) oder *Verify that access control fails* securely (V 4.8) oder *Verify that all logging controls are implemented on the server* (V 8.3) finde ich – zumindest für Level 2A – fragwürdig. Für mich stellt sich hier die Frage, wie das ohne Einblick in den Quellcode überhaupt nachzuweisen ist. In der Praxis bin ich mit diesen Punkten so vorgegangen, dass ich nach Indizien für einen Verstoß dagegen gesucht habe. Sehr befriedigend ist das jedoch nicht, denn es entspricht mehr einer Suche nach Schwachstellen – der Idee des ASVS nach sollte jedoch andersherum nach Indizien der Erfüllung gesucht werden.

Darüber hinaus ist der Unterschied zwischen einigen Punkten nicht ohne weitere Erklärung klar. So erscheint mir die Abgrenzung zwischen *Function Access* (V 4.1) und *Service Access* (V 4.6) oder zwischen *File Access* (V 4.3) und *Data Access* (V 4.7) eher künstlich – zumindest hatte ich noch keine Anwendung zu prüfen, bei der diese Unterscheidung relevant gewesen wäre.

Aus Prüfer-Sicht hinterlässt der Verweis im Bericht von einem Punkt auf einen anderen das Gefühl, als hätte man nicht sorgfältig genug gearbeitet. Man sollte sich an dieser Stelle jedoch nicht scheuen, einen Punkt als nicht anwendbar auszuschließen.

Andere Kritiken am ASVS gehen in eine ähnliche Richtung. So schleppt der ASVS nach Ansicht von Stevens [5] noch einigen Ballast mit sich herum, der bei den Prüfern zu Verunsicherung oder Frust führt, weil einige Prüfanforderungen fragwürdig oder unklar sind. Konkret identifiziert Stevens diverse Punkte im ASVS, die seiner Ansicht nach nicht notwendigerweise ein Mehr an Sicherheit bringen würden sondern eher als *nice-to-have* gelten.

Als Beispiel führt Stevens das Logging an: Sieben Prüfpunkte des ASVS verlangen, dass bestimmte Ereignisse aufgezeichnet werden, jedoch mache Logging eine Anwendung nicht sicherer. Auch wenn Logging einen Beitrag zur Nachvollziehbarkeit von Ereignissen leistet, neige ich selbst dazu, Logging ebenfalls nicht als Sicherheitsmechanismus anzusehen. Eine Reihe weiterer Kritikpunkte bezieht Stevens auf die zu enge Vorstellung, wie eine sichere Anwendung auszusehen hat – entscheidend hierbei ist jedoch die Risikoanalyse des Betreibers.

Darüber hinaus identifiziert Stevens einige Punkte im ASVS, die sich eigentlich auf die Anforderungsphase im Entwicklungsprozess beziehen, und die somit (in dieser Form) aus dem ASVS gestrichen werden sollten – das gilt auch für das Logging. Ich selbst habe mir dazu noch kein abschließendes Urteil gebildet, aus meiner Sicht ist es eher entscheidend, dass die einzelnen Anforderungen des ASVS leicht und eindeutig nachprüfbar sind.

Auch auf der Mailing-Liste wurden verschiedene Punkte oder Ansätze des ASVS kritisch diskutiert. Viele der genannten Kritikpunkte sind laut Aussage von Daniel Cuthbert derzeit Gegenstand der Überarbeitung des ASVS.

5 Fazit

Trotz einzelner Kritikpunkte am ASVS 2009 ist der ASVS eine gute Meßlatte zur Beurteilung der Sicherheit einer Anwendung. Da es keine ASVS-Zertifizierung gibt, kann man den ASVS vor allem als Best-Practice-Richtlinie zur Prüfung von Anwendungen auffassen. Das bedeutet, dass der Umfang und die Prüfung der Anforderungen des ASVS an die Bedürfnisse der konkreten Prüfung angepasst werden sollten, jedoch ohne den Charakter der Prüfung nach einem bestimmten Level unkenntlich zu machen. Dazu gehört, dass der Prüfer die Abweichungen der Anwendung vom ASVS als relevant oder irrelevant einstufen und ggf. die Restrisiken im Kontext des betrieblichen Anwendungsumfeld bewerten sollte.

Für Einsteiger wäre ein Guide hilfreich, der die einzelnen Prüfpunkte kommentiert und Hinweise gibt, wie die einzelnen Prüfpunkte adressiert werden sollten oder welche Kriterien zum Bestehen oder Nicht-Bestehen angelegt werden könnten. Dies würde die Lernkurve im Umgang mit dem ASVS deutlich angenehmer gestalten. Darüber hinaus würde sich damit eine Best Practice zur Anwendung des ASVS etablieren. Im Rahmen der offenen und öffentlichen Überarbeitung des Standards bietet es sich an, die eigenen Erfahrungen in die Verbesserung und Weiterentwicklung einfließen zu lassen.

Der ASVS ist ein Prüfstandard und eignet sich daher nur bedingt als Grundlage zur sicheren Softwareentwicklung. Die Kenntnis der ASVS-Prüfgebiete und -Punkte ist jedoch für Entwickler insofern interessant, als der ASVS einen Hinweis auf spätere Testfälle und Reviewgebiete gibt. Für die Verantwortlichen der Qualitätssicherung bietet der ASVS jedoch einen ausgezeichneten Rahmen für die Prüfung der Sicherheit einer Anwendung.

Gleichzeitig stellt der ASVS ein wertvolles Mittel für Anwendungsverantwortliche und Projektleiter dar, um die Sicherheit einer Anwendung ohne tiefere technische Kenntnisse bewerten zu können.

Verweise

[1] Kai Jendrian, *Überprüfung von Webanwendungen mit dem „OWASP Application Security Verification Standard 2009“*, DuD 3/2010, S. 138-142.

[2] Kai Jendrian, *Sicherheit als Qualitätsmerkmal mit OpenSAMM*, DuD 4/2012, S. 270-273.

[3] Sabha Kazerooni, Daniel Cuthbert, *OWASP Application Verification Standard 2009*, 2009, http://www.owasp.org/images/4/4e/OWASP_ASVS_2009_Web_App_Std_Release.pdf

[4] Matteo Meucci (Project Lead), *OWASP Testing Guide, v3*, http://www.owasp.org/images/5/56/OWASP_Testing_Guide_v3.pdf

[5] Herman Stevens: The Real World Against the OWASP ASVS, Juni 2011, http://blog.astyran.sg/2011/06/real-world-against-owasp-asvs.html

[6] Herman Stevens: Mapping the Skies, OWASP ASVS against Testing Guide (part one), März 2012 http://blog.astyran.sg/2012/03/mapping-skies-owasp-asvs-against.html

[7] Dave Wichers (Project Lead), OWASP Top 10, 2010, http://owasptop10.googlecode.com/files/OWASP%20Top%2010%20-%202010.pdf