Wehrhaft

Thomas Schreiber

Immer mehr Geschäftsprozesse stehen und fallen mit der Verfügbarkeit entsprechender Webanwendungen. Deren Sicherheit sollten Unternehmen systematisch angehen und nicht auf die leichte Schulter nehmen.

Herkömmliche Firewalls bieten keinerlei Schutz gegen Angriffe auf Webanwendungen. Oder anders gesagt: Eine Schwachstelle in einer Webanwendung kann von einem Angreifer ungehindert ausgenutzt werden, wenn die einzige Barriere zwischen seinem Browser und dem Webserver eine herkömmliche Firewall ist – gegenwärtig der Normalzustand bei fast allen Webangeboten.

Mit den Webanwendungen steht unter Umständen der gesamte Server oder mehr ohne Schutz im Netz. Denn bestimmte Schwachstellen erlauben den Zugriff über die Webanwendung bis ins interne Netzwerk hinein. Dieses große Loch mit einer nachgeschalteten Application Firewall, auch Webshield genannt, zu stopfen, ist nur eine, aber bei Weitem nicht die einzige notwendige Maßnahme, die solche Anwendungen und die durch sie umgesetzten Geschäftsprozesse absichert.

Ein Ebenenmodell wie in der Tabelle „Klassifizierung von Schwachpunkten“ soll bei der Strukturierung des weiten Feldes der Web Application Security helfen (siehe auch [1]). Es setzt auf die Netzwerkebene fünf weitere Schichten auf. Die Systemebene (1) befasst sich mit der Sicherheit der zur Realisierung der Webanwendung benötigten Software, die Technologieebene (2) behandelt Fragen wie die Wahl des richtigen Authentifizierungsverfahrens und der Softwarearchitektur, Ebene 3 ist der Bereich der Kodierung und damit der Programmierfehler. Hier sind so namhafte Schwachstellen wie Cross-Site Scripting und SQL-Injection angesiedelt.

Auf der Logikebene (4) geht es um die Art, wie die Fachprozesse in der Anwendung abgebildet sind und auf der semantischen (5) schließlich um den Schutz vor Täuschung und Betrug. In der rechten Spalte sind jeweils Beispiele für typische Schwachstellen oder Angriffstechniken genannt.

Altlasten überprüfen

Wer mit einem Onlineshop, einem Unternehmensportal oder einer eigenen geschäftlichen Anwendung im Web präsent ist, besitzt bereits eine Altlast. Denn sind diese Anwendungen nicht nach vorgegebenen Sicherheitsrichtlinien erstellt worden, ist die Wahrscheinlichkeit groß, dass sie Schwachstellen in sich tragen – je älter eine Webanwendung ist, desto mehr.

Ein Unternehmen sollte sich daher als Erstes Klarheit über den Stand der Sicherheit auf Anwendungsebene verschaffen. Das kann ein umfassender Penetrationstest aller Anwendungen und Webauftritte leisten. In der Regel ist es aber nicht erforderlich, alle gefundenen Schwachstellen für viel Geld zu beseitigen. Den Schutzbedarf der Anwendung und das Risiko einer Ausnutzung der Schwachstelle sollte der Verantwortliche gegen die Kosten zur Behebung abwägen.

Weiter sollte ein Unternehmen die Gelegenheit nutzen, die eine oder andere ohnehin nur noch selten genutzte Webanwendung endlich auszurangieren. Andere Anwendungen erstellt man vielleicht gleich ganz neu oder übernimmt ihre Funktion in eine für die Zukunft geplante neue Applikation. Tritt schließlich doch der schlimmste Fall ein, und man muss mit Schwachstellen fertig werden, obwohl man die Webanwendung nicht mehr anfassen möchte, so bietet sich hier unter Umständen und ausnahmsweise die Application Firewall als Retter an (siehe unten). Sie ist so einzustellen, dass sie die betreffenden Angriffsmuster blockiert.

Nach diesen einleitenden Maßnahmen gilt es, die Sicherheit auf Anwendungsebene im Unternehmen nachhaltig zu verankern. Zunächst zu den grundlegenden Maßnahmen.

Anwendungen absichern

Falls noch nicht geschehen, ist im Unternehmen ein Bewusstsein dafür zu schaffen, dass Anwendungssicherheit eine neue Aufgabe ist und vielerlei strukturelle Veränderungen nach sich zieht („Awareness“). Alle Beteiligten sind zu schulen, die Relevanz auf fachlicher Ebene ist herauszustellen und der Verantwortungsbereich IT-Sicherheit um dieses Element zu ergänzen.

Ein weiterer Aspekt ist die Organisation. So hat sich in vielen Unternehmen IT-Sicherheit als zentrale Stelle außerhalb der Fachabteilungen etabliert. Die Fachabteilung ist nicht aktiv in die sicherheitsgebenden Prozesse einbezogen: Sie nutzt die sichere Infrastruktur, unterliegt den Sicherheitsvorschriften und gibt die produktiven Anwendungen an den Betrieb ab. Anwendungssicherheit erfordert einen weitergehenden Beitrag der Fachabteilung, denn eine Webanwendung ist fast immer die Abbildung fachlicher Prozesse. Nur die Fachabteilung kann die fachlichen Abhängigkeiten von der Sicherheit der jeweiligen Anwendung beurteilen. Das gilt ganz besonders für die Ebenen der Logik und Semantik (siehe Tabelle „Klassifizierung…“). Ähnlich wie beim funktionalen Test einer Anwendung ist die Fachabteilung auch bei den Sicherheitsbelangen in die entsprechenden Prozesse einzubeziehen.

Die zu verteilenden Verantwortlichkeiten im Unternehmen lassen sich anhand des Ebenenmodells (siehe Tabelle „Zuständigkeiten in den Entstehungsphasen“) bestimmen. Ebene 1 ist Sache des Betriebs, Netzwerk- und Systemkenntnisse sind die benötigten Fähigkeiten. Ebene 2 wird von der Fachabteilung und Entwicklern mit Unterstützung des Betriebs verantwortet. Um Ebene 3 kümmert sich die Entwicklungsabteilung und Ebene 4 ist vorwiegend von der Fachabteilung, die eine Anwendung benötigt, zu steuern. Ebene 5 schließlich ist hinsichtlich der allgemeinen Grundsätze von der Zentrale und in puncto anwendungsspezifische von der Fachabteilung zu verantworten. Die rechte Spalte fasst die Zuständigkeiten für Planung, Umsetzung und Betrieb (Plan, Build, Run) einer Anwendung zusammen.

In Verträgen verankern

Es empfiehlt sich, Anwendungssicherheit zum Bestandteil der Werkverträge mit externen Dienstleistern zu machen. Sicherheitsmängel sind genauso wie funktionale Mängel als Fehler anzusehen und im Rahmen der Gewährleistung zu beheben. Auf Unternehmensseite dürfen Ausschreibungsbedingungen für Softwareprojekte Sicherheit allerdings nicht nur fordern, sie müssen sie auch fördern, indem das Budget dafür Spielraum lässt. Erhält der Billigste den Zuschlag, ist die Verlockung auf Anbieterseite groß, einen Preisvorteil durch Auslassung der Sicherheit zu erreichen.

Eine Application Firewall dient wie eine Netzwerkfirewall als zentraler Kontrollpunkt: Beide inspizieren den Datenstrom zwischen Browser und Server und lassen nur das durch, was die Regelbasis erlaubt. Auch wenn die Verlockung groß ist, mit einer Application Firewall lässt sich das Problem der unsicheren Webanwendungen nicht lösen – außer in dem oben unter Altlasten beschriebenen Sonderfall. Dazu ist die Webkommunikation zu komplex, sind die prinzipiellen Möglichkeiten von *http*-Filtern zu begrenzt. Die Erwartungen an eine Application Firewall sollten daher auch nicht zu hoch gesteckt werden.

Bestimmte Klassen von Schwachstellen auf Anwendungsebene lassen sich grundsätzlich nicht per Application Firewall schützen. Dazu gehört insbesondere alles, was sich auf der logischen und der semantischen Ebene befindet. Die Leistung reicht somit zwar nicht an die einer herkömmlichen Firewall heran, als „second line of defense“ sind Application Firewalls jedoch ein wichtiger Bestandteil einer umfassenden Sicherheitsstrategie für Webanwendungen. Derzeit ist ihre Verbreitung allerdings noch gering.

Das richtige Maß

Anwendungssicherheit ist kein Selbstzweck. Jede Maßnahme, die der Verantwortliche um der Sicherheit willen ergreift, ist in Relation zu setzen zur Gefährdungslage. Eine Bank ist einem anderen Bedrohungspotenzial ausgesetzt als ein Suchdienst. Ein Unternehmen, dessen Geschäftsmodell vollständig auf einer Webanwendung beruht, trägt ein höheres Risiko als ein produzierendes, das das Web lediglich zur Effizienzsteigerung nutzt. Unabhängig von einzelnen Webanwendungen sollte sich ein Unternehmen zunächst grundsätzlich einordnen und ein allgemeines Maß für den Schutzbedarf definieren. Die Schutzmaßnahmen und den für einzelne Webanwendungen zu treibenden Aufwand kann es dann im Rahmen dieser Metrik bestimmen. Der Hauptschauplatz, auf dem der Kampf gegen unsichere Webanwendungen ausgetragen werden muss, ist der Softwareentwicklungsprozess, die Wurzel des Übels. Wie die Klassifizierungstabelle zeigt, ist die Implementierungsebene, also die Kodierungsphase, dabei zwar ein wichtiger Ansatzpunkt, aber bei Weitem nicht der einzige.

Durch die Vorgabe von Programmierrichtlinien (Secure Coding Guidelines), die Web Application Security berücksichtigen, kann man eine Vielzahl von Fehlerquellen schon im Vorfeld ausschließen. Beispielsweise lassen sich eine Reihe von Schwachstellen, die das Eindringen in fremde Benutzeraccounts ermöglichen, durch die Befolgung einiger einfacher Regeln für die Programmierung des Sessionhandlings vermeiden. Der Best-Practices-Guide des Bundesamtes für Sicherheit in der Informationstechnik oder der OWASP Guide zur Web Application Security (siehe Kasten „Internetadressen zur Webanwendungssicherheit“) halten eine Vielzahl solcher Maßnahmen bereit.

Solche Handbücher bilden eine gute Basis für die Erstellung unternehmensspezifischer Secure Coding Guidelines, sind jedoch noch auf das eigene Umfeld abzustimmen. Je umfassender der Sicherheitsverantwortliche die eingesetzten Webtechniken, Frameworks, System- und Infrastrukturkomponenten in eine solche Guideline einbezieht, desto wirksamer wird das Ganze. Ob die Einhaltung auch zu überprüfen ist, steht auf einem anderen Blatt. Im ersten Schritt der Einführung von Web Application Security kann man darauf sicher verzichten, es ist schon viel erreicht, wenn man sie in den Verträgen mit externen Dienstleistern oder internen Stellen als verbindlich verankert.

Begehen die Entwickler in der Design-Phase eines Softwareprojekts Fehler, die sich erst im Feldtest als in dieser Form nicht brauchbare Funk-

iX-TRACT

- Dass Webanwendungen manipulierbar und gelegentlich sogar Einfallstore in die Firmen-IT sind, ist noch zu wenigen Verantwortlichen bewusst.
- Nur die Symptome zu bekämpfen, etwa mit speziellen Firewalls, reicht nicht aus. Websicherheit muss auf jeder Ebene ansetzen – ob bei Planung, Umsetzung oder Anwendungsbetrieb.
- Lassen Budget und Zeitplan kein ausführliches Konzept zu, empfiehlt es sich, zumindest einige elementare Sicherheitsmaßnahmen zu ergreifen.

tionen herausstellen, kann es sehr teuer werden, das nachträglich zu beheben. Genauso verhält es sich mit der Web Application Security: Eine Schwachstelle, die man im abschließenden Penetrationstest entdeckt, kann entweder nur notdürftig behoben werden und ist damit potenziell unsicher oder aber die Behebung führt zu hohen Kosten und Verzögerungen.

Sicherheit früh berücksichtigen

Je früher die Verantwortlichen im Entwicklungsprozess die Weichen für die Sicherheit stellen, desto nachhaltiger und günstiger können sie sie erreichen. Der Penetrationstest kurz vor Inbetriebnahme als einzige sicherheitsgebende Maßnahme ist die denkbar ungünstigste Variante. Termin und Kostendruck lassen es in der Regel nicht zu, dass der Eingriff über hastiges Flickwerk hinausgeht und am Ende eine rundum sichere Webanwendung online geht.

Ein Softwareprojekt kann sich viele Schwierigkeiten zum Schluss ersparen, wenn es einen in Webanwendungssicherheit erfahrenen Softwarearchitekten bereits in der Phase der Anforderungsdefinition („Requirement“) und des Entwurfs hinzuzieht. Sind die Möglichkeiten für einen solchen ganzheitlichen Ansatz nicht gegeben, sollten zumindest vor Beginn der Umsetzung das Fach- und IT-Konzept in einem Workshop mit einem Experten auf Herz und Nieren geprüft werden.

Abbildung 1 zeigt den typischen Projektverlauf für die Entwicklung einer Webanwendung. Jede Phase hat Ansatzpunkte für das Erhöhen und Überprüfen der Sicherheit. Das beginnt schon in der Definitionsphase: Neben Einsatz- sind auch Missbrauchsszenarien zu definieren. Je klarer das Bild potenziellen Missbrauchs ist, desto besser kann der Entwickler im grundlegenden Entwurf der Anwendung darauf reagieren.

Klassifizierung von Schwachpunkten

	Ebene	Inhalt	Beispiele
5	Semantik	Schutz vor Täuschung und Betrug	Phishing-Schutz, Schutz vor Informationspreisgabe
4	Logik	Absicherung von Prozessen und Workflows als Ganzes	„Passwort vergessen“-Funktion, Benutzer-Lock-Out
3	Implementierung	Vermeiden von Programmierfehlern, die zu Schwachstellen führen	Cross-Site Scripting, SQL-Injection
2	Technologie	richtige Wahl und sicherer Einsatz von Technologie	Verschlüsselung, Authentifizierung
1	System	Absicherung der auf der Systemplattform eingesetzten Software	Known Vulnerabilities, Konfigurationsfehler
0	Netzwerk und Host	Absicherung von Host und Netzwerk	

Zuständigkeiten in den Entstehungsphasen

	Ebene	Fähigkeiten/Kenntnisse	Stelle	Funktion
5	Semantik	Corporate Identity und Unternehmenskommunikation	Fachabteilung (fordert an), Zentrale	Plan
4	Logik	Kenntnisse der Geschäftsprozesse	Fachabteilung (fordert an), Zentrale	Plan
3	Implementierung	Softwareentwicklungskenntnisse	Entwickler (setzt um)	Build
2	Technologie	allgemein IT-Sicherheit	Fachabteilung, Entwickler, Betrieb	Build
1	System	Netzwerk- und Systemadministration	Betrieb	Run
0	Netzwerk und Host	Netzwerk- und Systemadministration	Betrieb	Run

Zusammen mit den Informationen aus Schutzbedarfs- und Risikoanalyse ergibt sich daraus „Security by Design“. Sicherheitsreviews der so entstehenden Dokumente sollten gewährleisten, dass die Entwickler nichts übersehen haben. Die Entwicklungsphase, das eigentliche Coding, können Werkzeuge begleiten, die neben der Qualität die Sicherheit erhöhen. Außer kommerziellen Tools wie Fortify, Codeassure oder Ounce Code existiert für Java-Entwickler das frei verfügbare LAPSE (erhältlich unter suif.stanford.edu/~livshits/work/lapse), das sich in Eclipse integrieren lässt.

Gezielte Code-Inspektion

Grundlage für die Entwicklung bilden die schon angesprochenen Secure Coding Guidelines. Ein Code Review wäre für kleinere oder nicht so schutzbedürftige Anwendungen sicher über das Ziel hinausgeschossen, aber die stichprobenartige oder auf einige neuralgische Punkte beschränkte gezielte Inspektion von Code-Teilen (zu nennen ist hier insbesondere der Bereich der Datenvalidierung) kann auf effiziente Weise potenzielle Sicherheitslöcher frühzeitig ans Licht bringen und so rechtzeitig die weitere Entwicklung beeinflussen. Penetrationstests vor der Produktivschaltung und das Überwachen des Anwendungsbetriebs und auch des versuchten Missbrauchs schließlich runden diese Maßnahmen ab.

Jede Änderung und jeder Release-Wechsel im weiteren Leben der Anwendung sollte erneut die genannten Sicherheitsmaßnahmen durchlaufen. Ein Tipp am Rande: Ein

E-Business treibendes Unternehmen sollte sicherstellen, dass von außen eintreffende Sicherheitshinweise, wie gefundene Schwachstellen, direkt zur Sicherheitsabteilung gelangen. Sonst findet man sich mit seiner Webanwendung unter Umständen auf einer imageschädigenden Full-Disclosure-Website wieder (www.heise.de/security/news/meldung/80204).

Webanwendungen durchlaufen vor dem Produktivbetrieb in der Regel qualitätssichernde Maßnahmen. Funktionale Tests und Untersuchungen der Laststabilität gehören dazu, die Prüfung der Sicherheit erfolgt bisher jedoch nur selten. Das ist zwingend zu ändern. Das Durchlaufen einer Sicherheitsanalyse auf Anwendungsebene sollte für jede Webanwendung in ihrer endgültigen Infrastruktur zu einem festen Bestandteil des Freigabeprozesses werden.

Ein Anwendungspenetrationstest ist jedoch deutlich anders als ein herkömmlicher, auf die Infrastruktur zielender. Das fängt mit dem Grad der Werkzeugunterstützung an: Auch wenn Hersteller von Webscannern es noch so sehr beteuern, Web-Application-Security-Analysen können bisher mit automatischen Tools nur unzureichend durchgeführt werden [2]. Ein Experte, der die Webanwendung manuell sowie mit Werkzeugen analysiert, ist unerlässlich. Er sollte fundierte Softwareentwicklungskenntnisse mitbringen. Erfahrungen mit Tests auf Netzwerkebene reichen nicht aus.

Internetadressen zur Webanwendungssicherheit

Open Web Application Security Project (OWASP):	www.owasp.org
Web Application Security Consortium:	www.webappsec.org
Maßnahmenkatalog und Best Practices für die Sicherheit von Webanwendungen des Bundesamtes für Sicherheit in der Informationstechnik (BSI):	www.bsi.de/literat/studien/websec/WebSec.pdf
OWASP-Guide zur Web Application Security:	www.owasp.org/index.php/Category:OWASP_Guide_Project

Damit ein solcher Test Schwachstellen in der Anwendung selbst aufdecken kann, sind alle vorgeschalteten Filter, insbesondere Application Firewalls, zu deaktivieren. Die Durchführung in der Zielumgebung ist Voraussetzung dafür, dass der Test auch Einflüsse erkennt, die nicht in der Anwendung zu suchen sind: etwa die fehlerhafte Konfiguration einer Systemkomponente oder sonstige Umgebungseinflüsse. Bei der Planung dieser letzten Phase darf man den Zeitfaktor nicht vergessen. Denn fördert der Test inakzeptable Schwachstellen zutage, sollte genügend Spielraum zu ihrer Behebung bis zur geplanten Inbetriebnahme vorhanden sein.

Womit nun beginnen, wenn der Rahmen der finanziellen oder organisatorischen Möglichkeiten eng gesteckt ist? Besser als gar nichts zu tun ist es, sich auf einige Mindestmaßnahmen zu beschränken.

Erste Schritte zur Sicherheit

Wichtig ist zunächst, die Anwendungssicherheit in das Unternehmen hineinzutragen („Awareness“). Schulungen und Informationsveranstaltungen, zumindest für Projektleiter und Prozessverantwortliche, sind ein guter Anfang. Die Gründung eines Kompetenzteams, bestehend aus interessierten Mitarbeitern aus den unterschiedlichen Bereichen, verleiht dem Thema eine zusätzliche Eigendynamik.

Unerlässlich ist auch die Sicherheitsuntersuchung bestehender Webanwendungen. Mit den gefundenen Problemen kann man durchaus pragmatisch umgehen (siehe „Altlasten“).

Zu den Mindestanforderungen gehört ebenso die Entwicklung von verbindlichen Secure Coding Guidelines auf der Basis bestehender Best-Practices-Empfehlungen. Das kann durch das Kompetenzteam oder einen erfahrenen Dienstleister erfolgen. Im Werkvertrag mit externen Softwarehäusern ist die Anwendungssicherheit einzufordern.

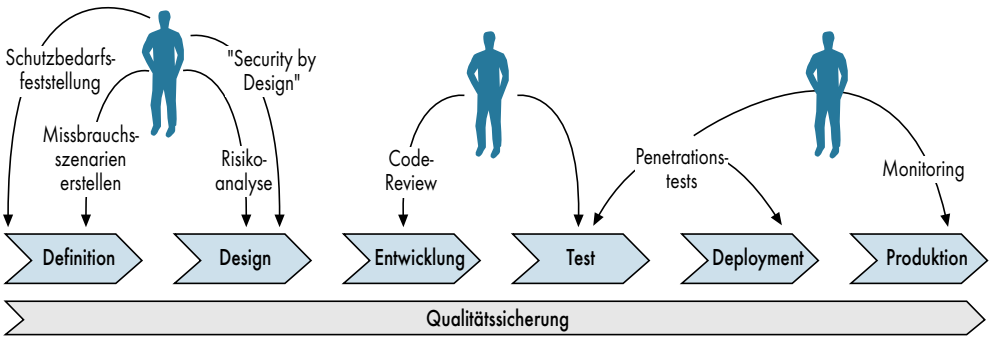
Und schließlich schadet es nicht, eine zweite Sicherungslinie einzurichten. Zum Einstieg bietet sich das frei verfügbare Apache-Modul *mod_security* (www.modsecurity.org) an, das man getrost als vollwertige Application Firewall bezeichnen kann. Um mod_security zum Einsatz zu bringen, bedarf es noch nicht einmal eines eigenen Rechners oder Anpassungen an der Infrastruktur, da es, zumindest in einer Apache-Umgebung, auf demselben Host wie die zu schützende Webanwendung betrieben werden kann.

Fazit

Angreifer verlegen sich angesichts mittlerweile sehr gut abgesicherter Netzwerke immer stärker auf die Anwendungsebene. Die Tore sind hier zumeist noch weit offen, das Bewusstsein für die Gefahren ist noch zu wenig ausgeprägt. Unternehmen sind daher gut beraten, wenn sie die Web Application Security als ernstzunehmende Disziplin in ihre IT-Security-Konzepte aufnehmen und mit deren Umsetzung zügig beginnen. (ur)

THOMAS SCHREIBER

ist Berater für Web Application Security und Geschäftsführer der SecureNet GmbH in München.



In jeder Phase der Entwicklung einer Webanwendung kann man Maßnahmen ergreifen, die der Verbesserung der Sicherheit dienlich sind (Abb. 1).

Literatur

[1] Sicher auf allen Ebenen; Bundesamt für Sicherheit in der Informationstechnik und SecureNet GmbH; www.bsi.bund.de/literat/forumkes/kes0305.pdf

[2] Holger Peine; Internetsicherheit; Lochdetektor; Websicherheitswerkzeuge auf dem Prüfstand; *iX* 10/2006, S. 62