Juliane Mathes

»Agieren statt reagieren: Sicherheit bei der Entwicklung von Webanwendungen

DIE AUTORIN

Dipl.-Inf. Juliane Mathes ist Senior Security Analyst bei der TÜV Rheinland i-sec. In dieser Funktion berät sie Unternehmen und Organisationen, um ihre Webapplikationen und Online-Portale gegen Angriffe von innen und außen zu sichern.

Zugriff auf Daten, Passwörter im Seitenquelltext und ausführliche Fehlermeldungen mit Datenbankinformationen: Was sich für IT-Sicherheitsverantwortliche wie ein Schreckensszenario anhört, stellt sich bei der Sicherheitsanalyse von Webapplikationen als Realität heraus. Daher gilt für die Entwicklung von Webapplikationen: Es ist nie zu früh (und oft zu spät), sicherheitsrelevante Aspekte von Anfang an in die Konzeption einzubeziehen.

Foto/Illu: © Kobes - Fotolia.com

Mit den „OWASP Top 10 2011“ hat das Open Web Application Security Project im April erneut die derzeit zehn größten Risiken für Webanwendungen vorgestellt. Das Ziel ist, Programmierer, Softwarearchitekten und Organisationen zu sensibilisieren, aufzuklären und die Sicherheit von Webapplikationen durch Erklärung der zehn wichtigsten Stolperfallen ins Bewusstsein zu rufen, denn größtenteils sind die Top 10 relativ simple Sicherheitsfehler: Datenbanklücken oder XSS-Schwachstellen etwa. Oft sind Session-Tokens oder Identitäten, das Hochladen von Daten oder die Ablage von Dateien, Verzeichnisse, Datenbankeinträge und digitale Schlüssel nicht hinreichend vor Angriffen geschützt.

Jahr für Jahr belegen bei den OWASP Top Ten die gleichen Themen die vorderen Plätze. Zu Recht – sicherheitsrelevante Schwachstellen werden im Entwicklungsprozess von Webapplikationen oft zu spät aufgedeckt. Meist befindet sich die Applikation schon in der Testphase oder sogar im Livebetrieb.

Sicherheitsrelevante Aspekte sollten jedoch frühzeitig in die Entwicklung neuer Webapplikationen eingebunden werden, um nachträglichen Mehraufwand oder gar Schadensfälle zu vermeiden. Erfahrungen zeigen, dass die Kosten zur Fehlerbehebung mit fortschreitender Entwicklungsstufe von der Analyse bis zur Implementierung um den Faktor 10 ansteigen. Die Gründe für Sicherheitslücken in Webapplikationen sind vielfältig – und bekannt.

Unsicheres Design

Im Design einer Applikation spiegeln sich Prozessabläufe und Funktionalitäten wider. Ein unübersichtliches, unstrukturiertes Design führt nicht nur zu mangelnder Usability, sondern es ist auch fehleranfälliger als ein gut konzipiertes und bietet mehr Angriffsflächen. Sicherheitslücken können nicht oder nur schwer entdeckt und noch schwerer behoben werden.

Fehlende Sicherheitsbrille

Webapplikationen zeigen häufig Schwachstellen, die auf Sicherheitsmängel im Code hinweisen. Im Design und der Implementierung müssen Sicherheitsaspekte wie Ein- und Ausgabevalidierung oder Berechtigungsprüfungen durchgängig berücksichtigt werden. Bei den Applikationsverantwortlichen und den Entwicklern sollte eine gewisse Sensibilität für das Thema IT-Sicherheit aufgebaut werden. Ein Angreifer betrachtet eine Applikation meist mit anderen Augen als der klassische Benutzer. Die Sichtweise des Angreifers zu kennen, ist ein Vorteil, der in die Entwicklung einfließen sollte. Schwachstellen in der Applikation können weiterhin durch falsche Prioritäten und höhere Entwicklungskosten entstehen. Jede weitere Anforderung in der Roadmap bedeutet zunächst einmal potenziell zusätzliche Entwicklungskosten und eine mögliche Verlängerung des Zeitplans.

Fehlende Definition von Rechten und Pflichten

Ein weiteres Risiko für die Anwendungssicherheit entsteht, wenn die Entwicklung ausgelagert wird. In dem Fall ist das naheliegende Ziel des Auftraggebers, eine qualitativ hochwertige, sichere Anwendung zu erhalten. Das Ziel des Auftragnehmers dagegen ist häufig eine schnelle und günstige Entwicklung mit dem Fokus auf Funktionalität und Anwenderfreundlichkeit, da diese beiden Merkmale dem Auftraggeber sofort ins Auge fallen – im Gegensatz zu fehlenden Sicherheitsmerkmalen, die meist erst später bemerkt werden. Daher kommt der Sicherheitsaspekt bei externen Entwicklungen häufig zu kurz.

Zu teuer?

Bei näherer Untersuchung stellt sich heraus, dass gerade das Kosten-argument ein Scheinargument ist: Die nachträgliche Behebung von Schwachstellen erzeugt wesentlich höhere Kosten, als eine rechtzeitige konzeptionelle Berücksichtigung sicherheitsrelevanter Risiken verursacht hätte. Außerdem geht eine späte Korrektur fast immer mit einer Verzögerung im „Go live“ einher. Und im schlimmsten Fall hat es dann schon einen Sicherheitsvorfall (Incident) gegeben, der einen hohen Imageschaden verursacht hat.

Sicherheitsrelevante Aspekte lassen sich bei der Neuentwicklung von Webapplikationen in die einzelnen Schritte des Entwicklungsprozesses integrieren. Welche Schritte sind das und was ist zu tun?

DESIGN

Das Sicherheitslevel der Webapplikation wird basierend auf dem Schutzbedarf der Applikation und der verarbeiteten Daten und Informationen festgeschrieben. Aus dem Sicherheitsanspruch an die Webapplikation resultieren die Sicherheitsanforderungen. Diese werden wie andere nicht-funktionale Anforderungen überwacht. So wird sichergestellt, dass ihre Umsetzung nicht von der Einschätzung einzelner Entwickler abhängt, sondern konsequent verfolgt wird. Wenn die Software extern entwickelt wird, ermöglicht das sogenannte „Tracken“

(Zurückverfolgen) der Anforderungen eine Überwachung des Software-Entwicklungspartners.

Basierend auf dem Design-Entwurf werden Bedrohungsszenarien (Threat-Modelle) entwickelt, um die potenziellen Gefahren für die geplante Webapplikation analysieren zu können: Was sind schützenswerte Daten, was die potenziellen Angriffspunkte der Anwendung und wie würde ein Angreifer vorgehen, um die Applikation zu kompromittieren? Dabei wird das Modell systematisch erweitert, um sicherzustellen, dass der größte Teil potenziell möglicher Angriffe auch wirklich berücksichtigt wird.

Die Analyse der einzelnen Bedrohungen und ihres zu erwartenden Schadenspotenzials bietet die Grundlage für die Auswahl und Priorisierung von Sicherheitsmaßnahmen. Als weiteres Auswahlkriterium ist das Kosten-Nutzen-Verhältnis zu nennen. Im Anschluss an das Threat-Modelling werden die Sicherheitsmaßnahmen definiert, die umzusetzen sind.

IMPLEMENTIERUNG

Rollen- und Rechtekonzept

Um geeignete Sicherheitsmaßnahmen in die Applikation zu implementieren, wird zunächst ein Rollen- und Rechtekonzept erstellt. Die Aufgaben, Eigenschaften und Rechte der Benutzer werden anhand der Benutzerrollen definiert, die einzelnen Benutzern oder Benutzergruppen zugeordnet werden können. Dies erleichtert die Rechteverwaltung, da bei einer Änderung der Rechtestruktur nur die Rechte der Benutzerrolle angepasst werden müssen und nicht die Rechte jedes einzelnen Benutzers.

Wenn sich der Status eines Benutzers ändert, wird ihm eine andere Rolle zugewiesen. Ein Benutzer kann mehrere Rollen haben, seine Rechte ergeben sich dann durch eine Kombination der Rechte aller Rollen. Webapplikationen nutzen das Rollen- und Rechtekonzept zur Darstellung einer an die Benutzerrollen angepassten grafischen Benutzeroberfläche.

Beispiel für eine Datenpanne:

Im Juni 2011 enthielt der Abbuchungstext der fälligen Kfz-Steuer auf dem Kontoauszug steuerpflichtiger Bürger in Baden-Württemberg neben dem Namen des Steuerpflichtigen, seinem Kfz-Kennzeichen und seiner Steuersumme weitere Details: die Steuernummer und Umsatzsteuer anderer Bürger und Unternehmen sowie ihre Religionszugehörigkeit.
Der Fehler war nach dem Update der Software für das Lastschriftverfahren aufgetreten. Offensichtlich war die Aktualisierung vor dem Einspielen ins System nicht genug getestet worden.

Eingabe-/Ausgabevalidierung

Alle Möglichkeiten der Dateneingaben müssen überprüft und validiert werden. Dies betrifft ebenso Eingabefelder (Name, E-Mail-Adresse) wie auch Hidden Fields und Auswahlkomponenten. In der Ausgabevalidierung wird geprüft, dass nur valide Inhalte an den Webbrowser ausgeliefert werden. Das Thema der Validierung kann durch den Einsatz gängiger Frameworks in der Entwicklung zentral geregelt werden. Zu beachten ist, dass die Möglichkeiten des Frameworks durchgängig in der Applikation verwendet werden müssen. Auch Flash-Komponenten oder die schnell erstellte Wetteranzeige in der Applikation können als Einfallstor für Angriffe dienen.

Session-Management

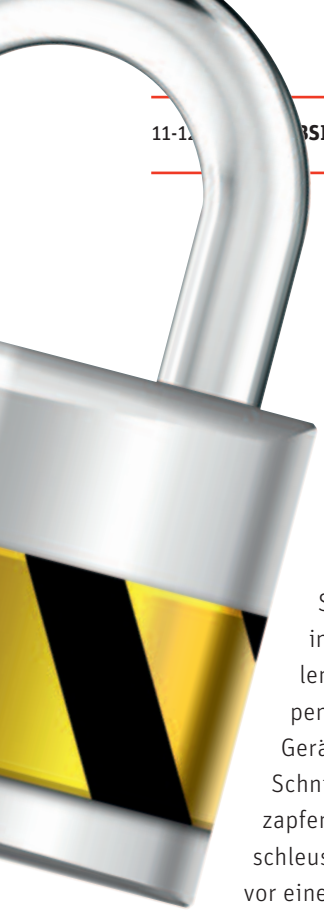
Die aktiven Benutzersitzungen in der Applikation und die zugehörigen Berechtigungen werden über Sessions abgebildet. Die Gültigkeit der Sitzungen muss bei jedem Request durch die Applikation geprüft werden. Session-Cookies müssen bei An- und Abmeldung gewechselt werden. Weiterhin empfiehlt sich der Schutz der Kekse durch das Setzen von Flags, wie zum Beispiel des HTTP-Only- oder des Secure-Flags.

Datenbank

Für Zugriffe auf die Datenbank wird ein Regelwerk aufgesetzt, damit auch große Datenmengen effizient, widerspruchsfrei und dauerhaft gespeichert werden können. Die Datenbank muss Daten in unterschiedlichen, bedarfsgerechten Darstellungsformen für Benutzer und Anwendungsprogramme bereitstellen. Das Datenbankmanagementsystem (DBMS) verwaltet und organisiert die strukturierte Speicherung der Daten und kontrolliert alle lesenden und schreibenden Zugriffe auf die Datenbank.

Dateiablage

Auch die Regeln für die Dateiablage müssen auf Sicherheitslücken hin abgeklopft werden. Welchen Sicherheitslevel hat welche Datei? Allgemeine Informationen, interne, vertrauenswürdige und geheime Dokumente werden nach bestimmten Kriterien eingestuft und entsprechend abgelegt und verschlüsselt. Was geschieht mit einem Datenüberhang, wenn der Speicherplatz nicht mehr für die Daten ausreicht? Wie wird mit Warenkörben verfahren, die gefüllt, aber nicht bestellt oder abgeholt wurden?



Systeme

Im nächsten Schritt werden die Systeme auf Sicherheitslücken hin untersucht: Wie sieht das Netzwerkdesign aus? Welche Schnittstellen gibt es intern zwischen Filialen, Abteilungen, Gruppen? Wo können mobile Geräte über offene Schnittstellen Daten anzapfen oder Malware einschleusen? Ist das System vor einem Denial-of-Service-Angriff geschützt? Könnten DDos-Attacken (Distributed Denial of Service) die Server lahmlegen und den Webshop unerreichbar machen?

Härtung

Welche ihrer vielen Funktionen brauchen die Betriebssysteme, Datenbanken und Anwendungen gar nicht für ihren konkreten Einsatzzweck? Überflüssige Features machen die Angriffsfläche unnötig groß. Schwachstellen in Softwaremodulen, die gar nicht genutzt werden, können die Sicherheit des Gesamtsystems gefährden. In der Systemhärtung werden daher alle überflüssigen Funktionen entfernt oder deaktiviert, um die Wahrscheinlichkeit eines erfolgreichen Angriffs auf ein Minimum zu reduzieren.

Verschlüsselung

Sensible Daten und die Kommunikation müssen verschlüsselt werden, denn gerade in Webshops und anderen Online-Anwendungen ist die Gefahr des Missbrauchs groß: Hier sind die Schnittstellen nach außen, das weit offene Portal, die Verletzlichkeit von Personendaten und Bestellvorgängen, Bankdaten und Benutzerkonten so nahe beieinander wie sonst nirgends. In einem mehrstufigen Konzept muss daher festgelegt werden, welche Daten im Klartext für alle offen lesbar sind und welche automatisch vom System nur verschlüsselt abgelegt und übertragen werden. Die Verschlüsselung muss automatisiert werden, damit die Einhaltung der Regeln nicht vom Sicherheitsbewusstsein oder der Sorgfalt einzelner Anwender abhängt.

Prozesse

Eine Analyse der internen und externen Prozesse zeigt Schwachstellen in der Administration, der Supply Chain und anderen Geschäftsprozessen, die über Abteilungen und Betriebsgrenzen hinweggehen. An welchen Stellen müssen zusätzliche Vorkehrungen getroffen werden, damit die Geschäftsprozesse ihren Zweck erfüllen – nämlich auf die Kunden und den Markt ausgerichtet sind, die Kernkompetenzen des Unternehmens unterstützen und die Informationstechnologie zur Unterstützung nutzen – ohne dass Sicherheitslücken auftreten? Kurz gesagt: Die IT-Security muss so hoch wie möglich sein und darf doch das Business und seine Abläufe nicht behindern oder stören.

Sichere Programmierung

Für Webapplikationen relevant ist beispielsweise das Open Web Application Security Project OWASP. Es bietet eine gute Orientierung über Schwachstellen und Strategien zu deren Vermeidung.

Punktuell lässt das Entwicklerteam sich bei der Umsetzung einzelner Sicherheitsanforderungen oder der Implementierung besonders sicherheitskritischer Punkte von Experten unterstützen (3rd Level Support). Besonders kritische Punkte sind etwa die Benutzerauthentifizierung, die Session-Verwaltung oder Datenbankzugriffe. Diese Code-Abschnitte werden von externen Experten mit entsprechendem Fachwissen analysiert

Test

Die Testphase beginnt schon während der Implementierung. Es werden sicherheitsrelevante Testfälle (Cases) erstellt und ausgeführt. Dadurch wird geprüft, inwieweit die definierten Sicherheitsanforderungen umgesetzt wurden. Neben Blackbox-Analysen, in denen der Angriff auf die Applikation ohne Vorwissen und Authentifizierungsdaten erfolgt, werden Whitebox-Analysen im authentifizierten Kontext durchgeführt. Die Analysen werden durch Audits der Systeme und Prozesse abgerundet. Insgesamt ergibt sich eine detaillierte Prüfung des aktuellen Sicherheitsniveaus der Applikation.

Betrieb

Da Sicherheitsanalysen immer nur eine Momentaufnahme darstellen, wird durch regelmäßige Monitorings eine kontinuierliche Überwachung der Sicherheitsaspekte erreicht. Kritische Codeteile werden in einem Review gegen die jeweils aktuellen Bedrohungsszenarien geprüft. Das Rollout von Applikationsbestandteilen wird sicherheitstechnisch begleitet. Insgesamt lässt sich so eine gleichbleibende Qualität der Applikationssicherheit erreichen.

Ergebnis: Die Pole-Position im Wettbewerb

Die frühzeitige Integration von Sicherheitsanforderungen in den Entwicklungsprozess hat viele Vorteile: Vor allem gewährleistet sie eine höhere Software-Qualität, die nachweisbar weniger Sicherheitsschwachstellen aufweist – ein nicht zu unterschätzender Vorteil im Wettbewerb. Die Hinzuziehung von externem Spezialisten-Know-how – einem kombinierten Wissenspool aus Software-, Test- und Security-Kompetenz – schafft den eigenen Entwicklern Freiraum, sich auf ihre Kernkompetenzen zu fokussieren. ¶