OWASP: API-Aufrufe richtig verwenden

Weckruf

Joshua Tiago

Code Reviews sind nur ein Mittel, mit denen Sicherheitsexperten die Risiken nicht nur mobiler Webanwendungen minimieren können. Weitere diskutierten Sicherheitsexperten aus aller Welt bei der zehnten OWASP AppSec USA.

In der Keynote des ersten Tages sprach der Sicherheitsexperte Bill Cheswick über die Entwicklung von Computersystemen in den letzten 40 Jahren. Viele der bereits vor 40 Jahren bekannten Risiken und Schwachstellen seien noch heute in modernen Betriebssystemen vorhanden. Während vor 100 Jahren ein Autofahrer noch viel mechanisches Vorwissen benötigte, um ein Fahrzeug sicher zu betreiben, könnten heutige Fahrzeuge von jedermann ohne Vorkenntnisse gefahren werden. Computersysteme setzen hingegen voraus, dass der Anwender für den sicheren Gebrauch über ein fundiertes Wissen zu IT-Sicherheitsthemen verfüge, so Cheswick. Abschließend sprach er sich für weitere Anstrengungen im Bereich Trusted Computing aus.

Welchen Herausforderungen sich Entwickler und Administratoren bei der Verwendung von HTTP-Security Headers in Webapplikationen stellen müssen, erklärte Kenneth Lee in seinem Vortrag „Lessons in Implementing HTTP Security Headers". Er stellte vor, welche Security Header es gibt (HSTS, X-XSS, Content-Security Policy et cetera), welche Funktionen sie haben und welche Browser welche Header unterstützen. So kann man mit dem Header X-Content-Security-Policy unterschiedliche Injection-Angriffe abwehren. Cross-Site Scripting lässt sich etwa dadurch verhindern, dass der Header das Ausführen von Inline Scripts unterbindet und stattdessen den gesamten JavaScript-Code in externe Dateien verschiebt. Findet nun ein Angreifer eine Cross-Site-Scripting-Schwachstelle in der Applikation, hat er keine Möglichkeit, den Code in die Seite einzubetten und zur Ausführung zu bringen.

In seinem Beitrag „Advanced Mobile Application Code Review Techniques" beschrieb Sreenarayan Ashokkumar ein Verfahren, mit dem man in kurzer Zeit Quellcode-Reviews für mobile Apps durchführen kann. Hierfür stellte er ein Werkzeug vor, das den Source-Code der jeweiligen App nach kritischen API-Aufrufen durchforstet. In einer langen Liste stellte er die relevanten API-Aufrufe der verschiedenen SDKs vor.

Riskante API-Aufrufe

Wenn Entwickler die beschriebenen API-Aufrufe unsachgemäß verwenden, kann das zu den in den „OWASP Top Ten Mobile Risks" beschriebenen Risiken führen – etwa „M10 Sensitive Information Disclosure": Standardmäßig erstellt iOS einen Screenshot der laufenden Applikation, wenn der Benutzer den Home-Button gedrückt hält. Macht er das innerhalb einer Banking-App, legt iOS 7 einen Screenshot von potenziell vertraulichen Daten auf dem Gerät ab. Um zu kontrollieren, ob ein Entwickler Gegenmaßnahmen getroffen hat, kann ein Prüfer im Code nach dem zugehörigen API-Aufruf suchen. In dem Fall: *applicationDidEnterBackground*. Dort sollte zum Beispiel *window.hidden* das Speichern des Screenshots verhindern.

Wie sich aktuelle Web Application Firewalls (WAF) in Bezug auf SQL-Injection austricksen lassen, stellte Robert Salgado in seinem Vortrag „New Exploitation and Obfuscation Techniques" vor. Dabei machte er sich zunutze, dass Datenbanksysteme Whitespaces in verschiedenen Formaten verstehen und decodieren. So kann ein Whitespace-Zeichen für MS SQL Server auf mehr als 30 verschiedene Weisen dargestellt werden. Viele der gängigen WAFs berücksichtigen dies in ihren Filterlisten nicht. (ur)