Entwicklung einer mobilen Applikation als Referenz für automatisierte statische Code-Analysen

Studiengang: BSc in Informatik | Vertiefung: IT-Security
Betreuer: Prof. Hansjürg Wenger
Experte: Andreas Dürsteler

Software-Lösungen zur statischen Analyse erlauben die Prüfung von Quellcode auf sicherheitskritische Schwachstellen ohne Ausführung der eigentlichen Applikation. Das Ziel unserer Arbeit ist festzustellen, auf welchem Stand heutige Lösungen im Bereich der automatisierten statischen Quellcode-Analyse für mobile Applikationen hinsichtlich ihrer Sicherheit sind. Die eingebauten Sicherheitsrisiken sind der Organisation OWASP und den «Mobile Top 10» entnommen.

Fragestellung

Welchen Grad an Sicherheit bietet uns moderne Applikationen, welche statische Sicherheitstests durchführen? Welche Art von sicherheitskritischen Fehlern erkennt eine kommerziell erhältliche Software tatsächlich? Gibt es ein Muster, welche Art von Sicherheitslücken erkennt werden können? Und welche Fehler erkennt die Analyse-Software?

Vorgehensweise

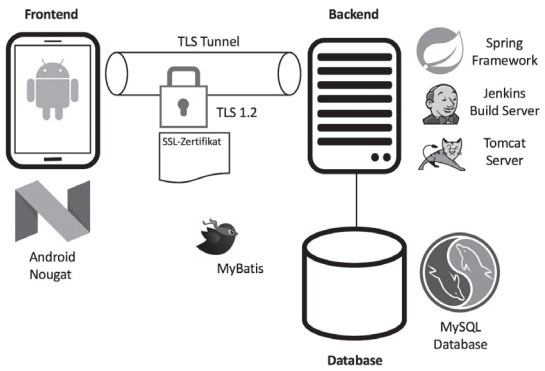
Um dies beurteilen zu können, haben wir eine mobile Applikation mit Backend aus einem Vorprojekt abgeändert und verschiedene Applikationszweige erstellt. In jedem der Applikationszweige ist jeweils eine Sicherheitslücke eingebaut. Die Nachweisbarkeit der Fehler reicht von trivialen Konfigurationsschwächen («gebrochene» Verschlüsselungsverfahren) bis zu schwieriger detektierbaren Prozessfehlern. Den Beweis einer tatsächlichen Schwäche erbringen wir sowohl durch «Unit»-Tests als auch durch Integrationstests. Die Art der Schwäche stammt aus dem OWASP «Mobile Top 10» Projekt. Gemäss dem «Magic Quadrant» von Gartner ist die evaluierte Standardprodukt führend im Bereich der statischen Sicherheits-Quellcode-Analyse. Daher haben wir in erster Linie die kritischeren Risiken vorgezogen, welche von deren «Scan»-Mustern auch erkannt werden können. Als Grundlage diente uns dafür die offizielle Produkt-Dokumentation. Gemäss der Beschreibung soll es dem Produkt beispielsweise möglich sein, Fehler in den Konfigurationen wie auch mangelhafte Geschäftsprozesse zu erkennen. Es wurde zudem darauf geachtet, dass die verwendeten Frameworks und Programmiersprachen durch die kommerzielle Software unterstützt sind.

Fazit

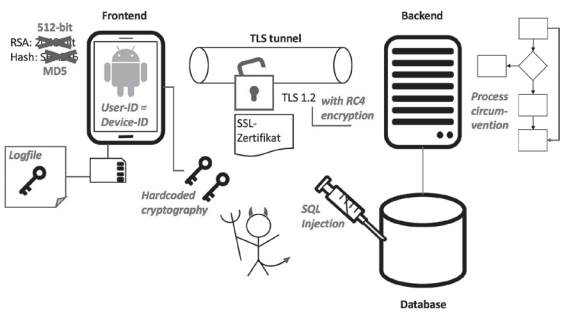
Die im Kontext dieser Arbeit verwendeten Werkzeuge für die Quellcode-Analyse haben mit der Standardkonfiguration fundamentale Sicherheitslücken unserer Referenzapplikation nicht erkannt. Die verwendeten Technologien sollten gemäss Herstellerdokumentation mit den Analyse-Werkzeugen kompatibel sein und die OWASP «Mobile Top 10» abdecken. Es könnte sein, dass eine nicht vollständige Unterstützung der von uns verwendeten Programmiersprachen, Bibliotheken und Frameworks der Grund für die fehlgeschlagene Suche darstellt. Gefährlich ist hierbei die falsche Sicherheit, die entsteht. Das Quellcode-Analyse-Werkzeug würde den Benutzer nämlich auch nicht über die möglicherweise vorliegende Inkompatibilität informieren.

Claudio Emmanuel Deluca

Michael Sebastian Zemp



Frontend und Backend tauschen untereinander asymmetrisch verschlüsselte Informationen in einem TLS-Kanal aus



Die Applikation enthält gezielt eingebaute Schachstellen sowohl im Frontend wie auch im Backend