CloudCommand

Cyber Security

CloudCommand GmbH chr.schumacher@gmx.tm

Queries & SQLi

CloudCommand GmbH chr.schumacher@gmx.tm

AGENDA

01 Wiederholung SQL

02 Konzept SQL Injection

03 SQL Injection Typen

01 Wiederholung SQL

Datenbanktypen

- Es gibt verschiedene Datenbanktypen
- Einer davon ist die relationale Datenbank → Nutzt SQL
- Beispiele sind MySQL, MariaDB und MSSQL
- Andere Datenbanken sind z.B. KeyValue-Stores, Row-based und Dokumenten Datenbanken
- Beispiele sind hier Redis, Cassandra und MongoDB

- Fokus hier auf relationalen Datenbanken, da komplexe Abfrage-Syntax ausgenutzt werden kann

Wichtige Befehle

- SQL Kommandos sind aus Verben und Werten aufgebaut
- Die meistgenutzten Verben sind:
  - SELECT
  - INSERT
  - UPDATE
  - DELETE
- Werte stellen das dar was verändert werden soll, z.B.:
  - Den Tabellennamen
  - Die Spalte/Zeile
  - Den Feldinhalt
  - Die ID

Beispiel Query

- Elemente in <> sind durch Werte zu ersetzen
- “SELECT * FROM <Tabelle>;”
- “SELECT <Spalte>, <Spalte2> FROM <Tabelle>;”
- “SELECT * FROM <Tabelle> WHERE id=<Zahl>;”
- “SELECT * FROM <Tabelle> WHERE <Spalte> > 30;”
- Zwischenaufgabe: Nur durch den Kontext, was geben diese Abfragen zurück?

Beispiel Query 2

- Elemente in <> sind durch Werte zu ersetzen
- “INSERT INTO <Tabelle> <Spalte> VALUES <Wert1>;”
- “INSERT INTO <Tabelle> (<Spalte1>,<Spalte2>,<Spalte3>) VALUES (<Wert1>, <Wert2>, <Wert3>);”
- “UPDATE <Tabelle> SET <Spalte1> = <Wert1> WHERE <Spalte2> = <Wert2>;”
- “UPDATE <Tabelle> SET <Spalte> = <Spalte> + <Wert>;”
- “DELETE FROM <Tabelle> WHERE <Spalte> = <Wert>;”
- “DELETE FROM <Tabelle>;”
- Zwischenaufgabe: Was geben diese Abfragen zurück?

02
Konzept SQL Injection

Die Idee

- Ein Textfeld auf einer Webseite wird direkt als Variable in eine SQL Abfrage eingesetzt
- Beispiel für eine Abfrage, <Input> ist der Inhalt unseres Textfeldes:
  SELECT * FROM “Angebote” WHERE “Name” LIKE ”%<Input>%”;
- Wenn die Webseite das Input nicht “reinigt” können wir durch durch cleveres Manipulieren des Inputs beliebige Abfragen auf der Datenbank durchführen.
- Zuerst beenden wir den ersten Abruf mit einem “;
- Jetzt können wir beliebiges abfragen wie z.B. SELECT * FROM geheime_tabelle;
- Oft wird das Output direkt mit ausgegeben.

Die Realität

- Moderne Web Frameworks reinigen alles Input, und bauen SQL Abfragen dynamisch zusammen, nachdem Sonderzeichen entfernt wurden
- Oft kann man das Output nicht direkt sehen. Wenn das Output nur als Zahl oder ähnliches (z.B. bei einem Aufrufzähler) dargestellt wird
- Hier gibt es andere Methoden, die auf Timing Differenzen setzen (Komplexe Anfragen dauern länger, was das gelingen des Angriffs prüfbar macht)
- Manuelles SQLi ist möglich, aber lässt sich mit Werkzeug vereinfachen → sqlmap

03
SQL Injection Typen

Klassische

- Die einfachste Art der SQL Injection
- Nutzt ein Input Feld oder einen Teil der URL
- Zeigt Ergebnisse direkt auf der Webseite/in der Applikation an indem Normale Ausgabe neu genutzt wird
- Signifikant einfacher wenn die SQL Abfrage im Source Code bekannt ist

Blind

- Hier kann der Angreifer die Ausgabe nicht sehen.
- Seiteneffekte müssen als Erfolgsindikator genutzt werden
- Oft Zeit, hier baut mal den SLEEP Befehl ein, und vergleicht die Zeit bis die Antwort kommt
- Aber auch Bedingungen können genutzt werden, wenn z.B. auf eine andere Seite weitergeleitet wird, basierend auf der Abfrage

Fehlerbasiert

- In manchen Systemen sind Fehlermeldung in Produktivsystem nicht deaktiviert
- So kann man Fehler in SQL Abfragen auslösen und anhand der Fehlermeldung Informationen über die Datenbank erhalten
- Diese Informationen können für weitere Angriffe genutzt werden

Second-Order

- Deutlich komplizierter
- Wir schreiben normal in die Datenbank, z.B. in ein Kommentarfeld
- Eine andere Applikation die auch die Datenbank nutzt liest dieses Feld aus und nutzt es in einer weiteren Abfrage.
- So können wir potenziell ein anderes System angreifen
- Nur selten realistisch

CloudCommand

CloudCommand GmbH chr.schumacher@gmx.tm