Ludwig-Maximilians-Universität München
Institut für Informatik
Lehr- und Forschungseinheit für Kommunikationssysteme und Systemprogrammierung
Prof. Dr. Helmut Reiser

MNM TEAM
Munich Network Management Team

IT-Sicherheit im Wintersemester 2024/2025
Übungsblatt 4

Besprechung: Do, 21.11.2024

Achtung: Zur Bearbeitung einiger Übungsaufgaben ist es notwendig sich über den Vorlesungsinhalt hinaus, durch Internet- und Literaturrecherche mit dem Thema zu beschäftigen.

Aufgabe 1: (T) Rootkits

Nachdem ein Angreifer erfolgreich Zugang zu einem IT-System, etwa durch das Ausnutzen einer dort vorhandenen Schwachstelle, erlangen konnte, wird dort meist ein Rootkit installiert.

a. Es wird grundsätzlich zwischen zwei Varianten von Rootkits unterschieden: User-Mode- und Kernel-Mode-Rootkits. Erläutern Sie diese kurz.

b. Wie unterscheidet sich ein Rootkit von anderer Malware, z.B. Viren, Würmer und Trojanischen Pferden?

c. Rootkits verfügen im Allgemeinen über eine sogenannte *Dropper*-Komponente. Welchem Zweck dient diese Komponente. Was wird unter einem *Multistage Dropper* verstanden?

d. Charakteristisch für Rootkits sind sogenannte Anti-Forensik-Maßnahmen. Erläutern Sie folgende Maßnahmen

   - Data Destruction
   - Data Concealment
   - Data Fabrication

Aufgabe 2: (T) XSS & SQL-Injection

a. In der Vorlesung wurden drei verschiedene Arten von *Cross-Site-Scripting* (XSS) vorgestellt. Welche der Varianten besitzt das höchste Bedrohungspotential?

b. Zum Schutz vor XSS existieren verschiedene Maßnahmen, welche einen Angriff verhindern oder dessen Auswirkungen verringern sollen. Beschreiben Sie kurz die folgenden Techniken:

   – Eingabevalidierung
   – Content Security Policy
   – HttpOnly

c. Beschreiben Sie wie SQL-Injection funktioniert und wie Sie eine Anfrage formulieren würden, um die Loginmaske einer Webseite zu umgehen und auf den Account **administrator** zuzugreifen, von der Sie wissen, dass die Passwörter mit folgendem SQL-Query überprüft werden:

```
'SELECT uid FROM users WHERE username = "' + $username + '" AND
password = "' + $password + '"'
```

   Sie können dabei die Parameter `$username` und `$password` über das Formular frei eingeben.

d. Beschreiben Sie sowohl abstrakt als auch konkret, wie Sie den Query aus der vorhergehenden Teilaufgabe verändern müssen, um SQL-Injection zu verhindern. Sie können sich bei der konkreten Beschreibung eine gängige Programmiersprache für das Web (z.B. PHP) aussuchen.

Aufgabe 3: (T) Security Code Review 101

Viele Verwundbarkeiten von Anwendungen gehen auf Unachtsamkeiten während der Programmierung zurück. Ein (unabhängig und teils automatisiert durchgeführtes) Code Review zielt darauf Security-Aspekte bereits zur Programmierzeit zu verbessern.

Nutzen Sie das OWASP *Secure Coding Dojo*, um typische Fehler bei Eingabenvalidierung, Speicheroperationen etc. zu identifizieren und Ihren eigenen Programmierstil zu verbessern!
`https://owasp.org/SecureCodingDojo/codereview101/`

Aufgabe 4: (T) Common Vulnerability Scoring System 3 (CVSSv3)

Für diese Aufgabe soll die folgende Schwachstellenbeschreibung verwendet werden, die über die vier Teilaufgaben hinweg modifiziert wird. Änderungen in einer der Teilaufgaben gelten auch in den darauf folgenden Teilaufgaben. (d.h. Änderungen in Teilaufgabe b) gelten auch für Teilaufgaben c) und d)).

In einer weit verbreiteten Webanwendung, die in ihrem Unternehmen als Kundenportal zur Verwaltung von Softwarelizenzen verwendet wird und daher öffentlich im Internet zugänglich sein muss, existiert eine cross-site request forgery (CSRF) Schwachstelle. Durch diese Schwachstelle können Angreifer aus der Ferne Aktionen mit den Rechten des angegriffenen Benutzers ausführen, wenn der Benutzer eine aktive Session hat und dazu gebracht werden kann, einen schädlichen Link zu öffnen.

Hinweis: Geben Sie bei den Aufgaben, bei denen explizit CVSS-Berechnungen gefordert sind, nicht nur deren Ergebnisse an, sondern begründen Sie auch die von Ihnen gewählten Optionen.

a. Beschreiben Sie kurz wie ein Angriff per CSRF üblicherweise funktioniert.

b. Berechnen Sie mithilfe des unter `https://www.first.org/cvss/calculator/3.1` verfügbaren CVSSv3-Calculators für die beschriebene Schwachstelle den CVSSv3 Base-Score. Vergleichen Sie diesen mit dem über `https://nvd.nist.gov/cvss.cfm?calculator&version=2` berechneten CVSSv2 Base-Score.

c. Die beschriebene Schwachstelle wurde am selben Tag auch auf der Security-Mailingliste *Full-Disclosure* publiziert und deren Ausnutzbarkeit anhand eines Proof-of-Concept (POC) bewiesen. Der Hersteller der Webanwendung hat die Schwachstelle nun auch offiziell bestätigt, aber bislang nur einen Workaround veröffentlicht. Wie verändert sich dadurch der CVSSv3 Base- bzw. Temporal-Score?

d. Bereits am nächsten Tag tauchte in einschlägigen Foren ein Exploit für diese Schwachstelle auf. Dieser besitzt keine besonderen Voraussetzungen und ist somit in jeder Situation funktional. Wie verändert sich dadurch der CVSSv3 Base-/Temporal-Score aus Aufgabe c)?

Aufgabe 5: (H) Buffer-Overflow

Angreifer nutzen oftmals Schwachstellen in lokal installierten Applikationen.

a. Erläutern Sie, was bei einem Buffer-Overflow genau passiert und wie ein Angreifer diesen für einen Angriff ausnutzen könnte?

b. Beschreiben Sie den Unterschied zwischen einem klassischen Buffer-Overflow und einem return-to-libc Angriff.

c. Nennen und beschreiben Sie mindestens drei Schutzmaßnahmen, die zum Schutz vor Buffer-Overflows eingesetzt werden können.