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

MNM TEAM
Munich Network Management Team

IT-Sicherheit im Wintersemester 2014/2015
Übungsblatt 5

Abgabetermin: 25.11.2014 bis 12:00 Uhr

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

Die schriftlichen Lösungen aller mit **H** gekennzeichneten Aufgaben sind **vor Beginn** der jeweils nächsten Übungsveranstaltung abzugeben (über Uniworx als Einzelabgabe). Während des Semesters werden vier Übungsblätter ausgewählt, korrigiert und bewertet. Bei vier als korrekt bewerteten Lösungen (mind. 75% der erreichbaren Punkte) erfolgt ein Bonus von zwei Drittel Notenstufen auf die Klausurnote, bei nur drei oder zwei richtigen Lösungen erhalten Sie einen Notenbonus von einer Drittel Notenstufe.

Aufgabe 11: **(H)** Rootkits (6 Punkte)

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

a. Man unterscheidet grundsätzlich zwei Varianten von Rootkits: 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 versteht man unter einem *Multistage Dropper*?

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

   - Data Destruction
   - Data Concealment
   - Data Fabrication

e. Wie beurteilen Sie die Erkennung von Rootkit-Software durch eine signatur-basierte Anti-Virus-Software. Würde der Einsatz einer heuristischen (Anomalie-basierten) Erkennung helfen?

Aufgabe 12: (H) XSS (3 Punkte)

a. Erstellen Sie eine Webseite, die einen in der URL der Seite übergebenen Parameter namens *?search=...* in den Seiteninhalt übernimmt und als `<p>`*parameter*`</p>` ausgibt. Wie verhält sich das Programm, wenn der Parameter *search* den Wert `<script>alert("XSS Attack!")</script>` enthält?

b. Löst das folgende Konstrukt das Problem? `<p><![CDATA[` *parameter* `]]></p>`
Warum (nicht)?

c. Welche Möglichkeiten bestehen prinzipiell Filter gegen XSS-Attacken zu umgehen? Nennen Sie mindestens drei Möglichkeiten.

Aufgabe 13: (H) SQL-Injection (2 Punkte)

Gegeben ist das folgende Szenario:
Auf einem Webserver befindet sich ein Webformular, mit dessen Hilfe eine angebundene Datenbank abgefragt werden kann. Die eigentliche Abfrage übernimmt dabei ein CGI-Skript auf dem Webserver. Das Resultat der Abfrage ist abhängig von einem ID Feld, das vom Webformular gesetzt wird und mittels einem Parameter in der URL an das CGI-Skript weitergereicht wird. Ein valider Aufruf des CGI-Skripts lautet z.B. *http://webserver/cgi-bin/query.cgi?ID=86*.

a. Das Skript generiert folgenden SQL Query String: ”`SELECT produkt FROM artikel WHERE ID=`” + ID + ”`;`”
Modifizieren Sie den Paramter in der URL des CGI-Skriptes so, dass die Abfrage normal ausgeführt wird und anschließend die gesamte Tabelle Artikel gelöscht wird!

b. Das Skript generiert folgenden SQL Query String: ”`SELECT produkt, preis FROM artikel WHERE ID like '%`” + ID + ”`%';`”
Modifizieren Sie den Paramter in der URL des CGI-Skriptes so, dass die Abfrage normal ausgeführt wird und anschließend der Preis aller Produkte deren ID auf 2 endet halbiert wird!

c. Das Skript generiert folgenden SQL Query String: ”`SELECT produkt, preis FROM artikel WHERE ID=`” + ID + ”`;`”
Modifizieren Sie den Paramter in der URL des CGI-Skriptes so, dass die Abfrage normal ausgeführt wird und zusammen mit allen Kennungen der Datenbank samt ihren Passwort Hashes ausgegeben wird!

d. Wie können die beschriebenen Attacken verhindert werden?