Blatt Nr. 2 (Ausgabe: 14. April 2016, Abgabe: 04. Mai 2016)

Kennwortsicherheit

Übungsaufgabe 1. Sicherheit lokaler Rechner

Aufgabe 1.1 Zugriff auf /etc/passwd und /etc/shadow des Webservers

Überblick über die VM.

- *Blatt2-Admin-PC.vmwarevm* wurde aus */home/vmware* nach */home/ss16g07/vmware* kopiert
- wurde über *File -> Open...* importiert (die virtuelle Festplatte wurde eingelesen)
- VM wurde gestartet; beim Boot wurde danach gefragt, ob die VM kopiert oder verschoben wurde; nach Aufgabe wurde "kopiert" ausgewählt

Booten von der CD

- *grml*-iso wurde aus */home/vmware* nach */home/ss16g07/vmware* kopiert
- Neues Image wurde in den VM-Einstellungen in das CD-Laufwerk eingelegt
- Ebenfalls unter den VM-Einstellungen wurde das CD-Laufwerk verbunden
- Während des Bootens der VM wurde das BIOS aufgerufen, um sicherzustellen, dass von der CD gebootet wird

Einlesen und durchsuchen der Root-Partition

- Wählen des deutschen Tastaturlayouts mit $d \rightarrow$ *Enter*
- mounten der Festplatte mit *mount -r /dev/sda1*
- nach */etc/fstab* Festplatte nun */mnt/sda1* zugreifbar
- Auslesen der Dateien */etc/shadow* und */etc/passwd* mit *cat*
  - *passwd* enthält Einträge der folgenden Form: [1]
    - *<Nutzername>*:x[1]:*<Nutzer ID>*:*<Gruppen ID>*:*<Nutzer ID Info>*:*<home-Verzeichnis>*:*<Shell>*
  - *shadow* enthält Einträge der folgenden Form: [2]
    - *<Nutzername>*:*<verschlüsseltes Password>*:*<Tag der letzen Passwortänderung>*[2]:*<minimaler Zeitabstand zwischen Passwortänderungen>*[3]:*<maximaler Zeitabstand zwischen Passwortänderungen>*[4]:*<Warnungszeitraum für auslaufende Passwörter>*:*<Zeit nach der ein Password ausläuft>*[5]:*<Zeit, die seit der Inaktivität des Accounts vergangen ist>*
- es gibt die Benutzer *webadmin* und *georg*
- Herausfinden der Nutzergruppen mit *cat group | grep <Benutzername>*[6]
  - georg : admin georg
  - webadmin : adm dialout cdrom plugdev lpadmin webadmin sambashare

[1] *x* (bei Ubuntu 14: *) indiziert, dass ein verschlüsseltes Passwort für diesen Nutzer in */etc/shadow* vermerkt ist
[2] in Tagen seit dem 1. Jan 1970
[3] Zeit, bis das Passwort wieder geändert werden kann
[4] Zeitpunkt, an dem das Passwort verfällt
[5] nach Inaktivität des Accounts
[6] Durch die | wird die Ausgabe von *cat group* an den *grep*-Befehl weitergegeben, der alles herausfiltert, was nicht zum ihm angegebenen Parameter passt

Aufgabe 1.2 Auslesen von Kennwörter

- salting: Hinzufügen einer zufälligen Zeichenkette ("salt")
- hashing: Umrechnung der Daten in Hash-Werte[1]

Installieren und Verwendung von *john*

- John wurde installiert mit *apt-get install john*, es konnte jedoch nicht authentifiziert werden
- Einfaches Ausführen von *john* zeigt die Hilfe-Seite
- Eingabe des Befehls *john –incremental –users=webadmin /mnt/sda1/etc/shadow*, um das Passwort von *webadmin* im *incremental*-Mode zu ermitteln
- Nach 5 Minuten wurde eine manuelle Terminierung durchgeführt

Wörterbuchangriff

- es wurde in das home Verzeichnis navigiert damit wieder Schreibzugriff besteht
- mit *wget http:download.openwall.net/pub/wordlists/all.gz* wurde ein Wörterbuch runtergeladen
- durch *gunzip all.gz* wurde das Wörterbuch entpackt
- und mit *john –wordlist=all –users=webadmin /mnt/sda1/etc/shadow* der Angriff gestartet
- nach 21.01 sec war das Passwort herausgefunden: mockingbird

Aufgabe 1.3 Setzen von neuen Kennwörtern

- Das Passwort von georg ist nicht ohne weiteres ermittelbar, weil es wahrscheinlich nicht im Wörterbuch steht
- zum unmounten von sda1 wurde *umount /dev/sda1* eingegeben
- zum erneuten mounten wurde *mount -w /dev/sda1* eingegeben
- zum Ändern des root Verzeichnsises mit shell Wechsel wurde *chroot /mnt/sda1/ /bin/sh* eigegeben
- nun wurde das Passwort von georg auf *1* gesetzt: *passwd georg 1*
- es wurde das sytem durch *exit* gefolgt von *shutdown -r now* neu gestartet und sich als georg eingeloggt

Übungsaufgabe 2. Sichere Speicherung von Kennwörtern

Aufgabe 2.1 Angriffe mit Hashdatenbanken und Rainbow-Tables

- es wurde in das home Verzeichnis von webadmin navigiert durch *cd /home/webadmin/*
- Wechseln in das Unterverzeichnis *Rainbowtables/rcracki*
- Ausführung von *rcracki* mit *./rcracki <table-path> -l <password-file>*
- es konnten nicht alle Passwörter ermittelt werden.Vermutlich weil nicht alle Passwörter in der benutzten RainbowTable codiert waren
- eigene Programme sind immer gut, weil man weiß, was sie können, dementprecehnd dauert es aber auch lange, viel Umfang einzbauen
- diese Speicherung würde (da jedes Passwort einen Hash der Länge 128 Bit[4] generiert) $\Sigma_{i=1}^{7} 128^i$ Bit $\approxeq$ 71 Terabyte verbrauchen, während eine der gegebenen Rainbowtables nur ca. 40 MB groß ist

[1] Werte fester Länge, typischerweise hexadezimal codiert [3]

Aufgabe 2.2 Eigener Passwort-Cracker

Quellcode zu dieser Aufgabe zu finden unter **Anhang I: Erinnerungshilfe**

Passwort: s1v3s

Lösungsweg:

- Darstellung der Passwortstellen wie "Walzen"bei einem Zahlenschloss (Wert von a bis 9)
- Senden eines Ticks an die Walzen
- Erste Walze wird gedreht, bei Überlauf Weitergabe an nächste Walze (rekursiv über alle sechs Walzen)
- Startwert jeder Walze ist " "
- Abfragen des Walzenstandes (Char-Sequenz) nach jedem Tick
- abgefragten Stand mit salt konkatenieren und hashen
- Kombinationen reichen von "a" bis "999999"
- Abbruch, wenn gehashte Kombination mit gegebenem Hash übereinstimmt

Aufgabe 2.3 Eigene Kennwort-Speicherfunktion in Java

Quellcode zu dieser Aufgabe zu finden unter **Anhang II: Useradmin-Klasse**

Funktionalität:

- Speicherformat: `username:salt:password-hash`
- Speicherort: Festgelegt auf `~/Documents/passwoerter.txt`
  - Verbesserungsoption: Benutzer den Speicherpfad angeben lassen
- 5000-maliges hashen des Passwortes
- `checkUser` nimmt das Nutzer-Passwort-Tupel entgegen, hasht das Salt-Passwort-Tupel 5000 mal und vergleicht mit dem Nutzer-Hash-Tupel im Speicher
- *main*-Methode nimmt als zusätzliche Parameter Methodennamen und Nutzernamen entgegen
- Eingabeaufforderung für Passwort (vgl. Zusatzfrage, s.u.)
- Vorhandene Datensätze werden überschrieben

Zusatzfrage. Wenn das Passwort als Konsolenparameter übergeben wird, ist es in der Historie einzusehen und kann über `script` gespeichert werden. Aus diesem Grund wurde sich dazu entschieden, das Passwort über ein `JOPtionPane` im Programm abzufragen.

Übungsaufgabe 3. Forensische Wiederherstellung von Kennwörtern

Aufgabe 3.1

- Speicherorte: Swap, Registry, bash-history

Aufgabe 3.2

Es wurde die bash-Chronik ausgelesen mit `/mnt/sda1/home/user # cat .bash_history`.
Inhalt:

- reboot
- cat /etc/inittab
- man intro
- man ls
- exit
- vi /etc/sudoers
- top
- pico
- cd
- exit
- java
- cd Desktop
- jedit wp-config.php
- scp wp-config.php root@10.1.1.2:/var/www/wordpress/
- rm wp-config.php
- exit

Aufgabe 3.3

Es wurden folgende Passwörter gefunden:

- **user:** bloguser
  password: Flugentenfederkiel/991199
- **user:** blog_user
  password: DUzvAu22cKatsXyV
- **user:** bloguser
  password: Kindergeburtstag/119911

Der letzte Eintrag führte zum erfolgreichen Login.

Übungsaufgabe 4. Unsicherer Umgang mit Passwörtern in Java

- Mögliche Sicherheitslücke in Klasse `transport/HTTPTransport.java`
- entschlüsseltes Passwort wird in String gespeichert
- nach [5] sind Strings im Speicher auslesbar
- alternativ kann Prozess gedumped werden, womit später der Dump ausgelesen werden kann

Literatur

[1] http://www.cyberciti.biz/faq/understanding-etcpasswd-file-format/

[2] http://www.cyberciti.biz/faq/understanding-etcshadow-file/

[3] http://zeitstempel.hauke-laging.de/hashinfo.php

[4] http://de.wikipedia.org/wiki/Message-Digest_Algorithm_5

[5] http://www.foo.be/course/dess-20082009/davidoff-clearmem-linux.pdf

Anhang I: Erinnerungshilfe

Anhang II: Useradmin-Klasse