Blatt 2 - Kennwortsicherheit

Kolja Hopfmann, Jonas Sander

00/00/0000

---

1 Sicherheit lokaler Rechner

1.1

1.2

1.3

2 Sichere Speicherung von Kennwörtern

2.1

1. Zuerst wurde in das Verzeichnis von rcracki navigiert.

```
cd webadmin/Rainbowtables/rcracki_mt_0.7.0_Linux:x86_64
```

Hier wurde dann rcracki auf der Datei mit den Passwörtern ausgeführt.

```
./cracki -l [password txt] [path to rainbow table]
```

Hierbei wurden Passwort Nummer 4: ulardi und Passwort Nummer 5: avanti gefunden. 2. Die Restlichen Passwörter konnten mit der Verwendeten Rainbowtable nicht geknackt werden. Dies ist darauf zurück zu führen, dass die Passwörter nicht als Wörter in der Tabelle enthalten sind. Ein erneuter Versuch mit einer anderen Rainbowtable könnte weitere Passwörter knacken,

jedoch ist dies keineswegs sicher.
Für das Abspeichern der MD5-Hashes aller alphanumerischen Passwörtern der Länge 1-7 wäre ein Speicher von $\sum_1^7(62^i \cdot 32byte) = 1,1 \cdot 10^{14}Byte = 110TB$ nötig. Dies ist um den Faktor 10^4 größer als die verwendete Rainbowtable und deutlich zu groß zur Speicherung.

2.2

Unsere Lösung war auf dem Alphabet bestehend aus 0-9 und a-z iterativ alle möglichen Strings durch zu probieren. Hierfür haben wird eine Java Klasse geschrieben, die in einer Schleife von 0 bis $36^7 - 1$ iteriert. In Jeder Iteration wird ein String zusammengesetzt, dieser mit dem Salt gehasht und das Ergebnis des Hashes mit dem in der Passwort-Datei gefundenem Hash verglichen. Hierbei werden erst alle 1-Stelligen, dann alle 2-Stelligen Passwörter durchlaufen etc. Hiermit wurde das Passwort s1v3s gefunden. Die Datei bruteforce.java befindet sich im Anhang.

2.3

Die erstellte Java Datei Useradmin.java befindet sich im Anhang.

3 Forensische Wiederherstellung von Kennwörtern

3.1

Aus den empfohlenen Quellen zu Blatt 2 unter "Zum sicheren Umgang mit Passwörtern im Speicher" geht hervor, dass viele Anwendungen Passwörter als Klartext im Arbeitsspeicher ablegen. Dadurch ist es Möglich das Passwort (für kurze Zeit) selbst nach ausschalten des Systems aus dem Speicher zu extrahieren. Desweiteren Dient eine Swap-Partition als zusätzlicher Arbeitsspeicher. So ist es Möglich aus der Swap-Partition Passwörter zu extrahieren.

3.2

Unter grml wurde der Inhalt der VM gemountet:

```
mount /dev/sda1
```

Unter */home/user/.bash_history* befinden sich die letzten commands welcher der user in der letzten bash-session in die shell eingetippt hat.

3.3

Da der Admin zunächst mit Jedit die Server-File editiert hatte haben wir zunächst in den Jedit config-Files gesucht, unter *home/.jedit*. Da war zu erkennen dass das alte Passwort "Flugentenfederkiel/991199" war, mit dem User "bloguser".

Daraufhin wurde versucht mit dem Programm photorec, Teile der Swap-Parition wiederherzustellen:

```
lsblk
photorec /dev/sda5
```

Die wiederhergestellten Dateien wurden aufgrund Platzmangel der grml-CD unter */dev/sda*1 abgelegt. Nun wurde versucht die Dateien nach dem String "PASSWORD" zu untersuchen da dies in der Server-Datei vor dem tatsächlichen Passwort stand.

```
find . -name "*.elf" | xargs grep -E 'PASSWORD'
find . -name "*.txt" | xargs grep -E 'PASSWORD'
find . -name "*.java" | xargs grep -E 'PASSWORD'
grep -a f0402684.elf
grep -a f0853144.elf
grep -a f0932456.elf
grep -a f0052256.elf
```

In der Datei f0853144.elf befand sich das Passwort als Klartext: Kindergeburtstag/119911

4 Unsicherer Umgang mit Passwörtern in Java

Der Passwortmanager verwendet sowohl für Encryption als auch Decryption die selbe Funktion. Dieser wird entweder das Klartextpasswort oder das Verschlüsselte Passwort übergeben. Kommt man in den Besitz der Verschlüsselten Passwörter, kann man sie also einfach über den Passwortmanager entschlüsseln.