Labreport #2

Patrick Eickhoff, Alexander Timmermann

Aufgabe 1

1.1 Zugriff auf */etc/passwd* und */etc/shadow*

2. Das Booten des grml-ISO gestaltete sich schwieriger als erwartet. Da das Boot-Menü nur in einem sehr kurzen Zeitfenster zu aktivieren ist, mussten wir in der *.vmx*-Datei von VMWare die Zeile `bios.forceSetupOnce = True` hinzufügen. Auf diese Weise haben wir einmalig das Öffnen des Bios beim Bootvorgang erzwungen.

5. In der Datei *passwd* finden wir Einträge, wie z.B. <Beispiel>. Über die Manualpage (`man 5 passwd`) erfahren wir, das alle Einträge folgende Form besitzen:

   ```
   <login name>:
   <optional encrypted password>:
   <numerical user ID>:
   <numerical group ID>:
   <user name or comment field>:<user home directory>
   [ : <optional user command interpreter> ]
   ```

   Wenn im Passwort Feld ein `x` steht, finden wir das zugehörige Passwort verschlüsselt in der *shadow*-Datei. Das Format dieser Datei können wir ebenfalls über `man 5 shadow` ermitteln. In dieser Datei sind nun tatsächlich die Passwörter der Nutzer angegeben.

6. Der einzige Nutzer, der in Mitglied der *admin*-Gruppe ist, heißt georg (admin : georg).

1.2 Auslesen von Kennwörtern

1. Ein Passwort zu *hashen* bedeutet, das Passwort mittels einer *Hash-Funktion* zu verschlüsseln. Eine *Hash-Funktion* hat die Eigenschaften, das sie eine Einwegfunktion ist, und immer eine Ausgabe fixer Länge erzeugt. Zudem sollte sie auch kollisionsfrei sein. Da eine *Hash-Funktion* bei gleicher Eingabe immer den gleichen *Hash* ausgibt, ist es möglich Anhand einer großen Datenbank von *Hashes* die Eingabe zu ermitteln. Um dies zu verhindern, werden die Passwörter zusätzlich noch *gesaltet*.

Dies bedeutet nichts anderes, als das ein ein zusätzliches Wort vor oder hinter dem Passwort angehängt wird, bevor es *gehasht* wird. Auf diese Weise werden für jeden *Salt* unterschiedliche *Hashes* erzeugt.

4. Mittels `sudo john -mode:incremental -users:webadmin -format:crypt shadow` versuchen wir das Passwort des Nutzers *webadmin* per Bruteforce zu entschlüsseln. Nach 5 Minuten brechen wir den Versuch erfolglos ab. Vermutlich ist das Passwort so lang, dass ein Bruteforce- Angriff von enormer Dauer wäre.

5. Nachdem wir die Wörterbuch-Dateien mittels
`wget http://download.openwall.net/pub/wordlists/ all.gz`
heruntergeladen und über textttgunzip entpackt haben, können wir unseren Angriff beginnen. Mit dem Befehl:
`sudo john -mode:wordlist:/root/all -users:webadmin -format:crypt shadow`
starten wir den Wörterbuchangriff und finden heraus, das das Passwort des Nutzers *webadmin* **mockingbibrd** lautet.

1.3 Setzen von neuen Passwörtern

1. Vermutlich lässt sich das Passwort des Nutzers *georg* nicht einfach mit *John the Ripper* ermitteln, da es etweder gesaltet ist, zu lang oder einfach nicht in unserer Wörterbuch-Datei enthalten ist.

2. Mittels `mount -w dev/sda1` mounten wir das Image erneut mit Schreibzugriff.

3. Über den Befel `chroot . /bin/bash` öffnen wir im aktuellen Verzeichnis eine interaktive *bash-Shell*.

4. Mit `passwd georg` setzen wir nun das neue Passwort *passwort123*.

2 Sichere Speicherung von Kennwörtern

2.1 Angriffe mit Hashdatenbanken und Rainbow-Tables

1. Durch den Befehl `./rcracki` erfahren wir, dass *rcrack* mit einem Pfad zu den Rainbowtables, sowie der Datei, die entschlüsselt werden soll, ausgeführt wird. Mittels Optionen kann angegeben werden, um welche Art von Datei es sich handelt.

   ```
   ./rcrack rainbow_table_path [-h hash | -l hashlist | -f pwdumpfile]
   ```

   Nun starten wir unseren Angriff mit folgendem Befehl:

   ```
   ./rcrack ../md5_mixalpha-numeric-space_1-7_*/*.rti -l ~/geheime_kennwoerter.txt
   ```

   Wir erhalten folgende 5 Passwörter: ente,ball,borkeni,ulardi, avanti. 2 Hashes konnten nicht entschlüsselt werden.

2. Vermutlich wurden die beiden Passwörter, die nicht entschlüsselt werden konnten, gesaltet. Die Idee Programm zu schreiben, das MD5-Hashes für alle alphanumerischen Passwörter bis zur Länge 7 berechnet und abspeichert, ist jedoch auch keine passable Lösung. Alleine das Berechnen von

$$\sum_{i=0}^{7} 62^i = 3579345993195$$

Kennwörtern würde enorme Rechenleistung benötigen. Da MD5 immer Hashes der Länge 16-Byte generiert würde die Speicherung dieser

$$3579345993195 * 16B = 5,72695358910^{13}B = 53336GB = 52TB$$

an Festplattenspeicher in Anspruch nehmen. Die Rainbowtables aus der vorherigen Aufgabe haben hingegen nur ca. 12GB benötigt.

2.2 Eigener Passwort-Cracker

Um das Passwort zu entschlüsseln haben wir uns entschieden einfach einen Bruteforce-Angriff durchzuführen. Hierzu lassen wir uns einfach mittels for-Schleife und einer Funktion zum erzeugen des karthesischen Produkts über einem Alphabet mit einer gegebenen Länge i alle alphanumerischen, klein geschriebenen Kennwörter generieren. Jedes Kennwort wird dann gesaltet und mittels MD5-Funktion gehasht. Die Ausgabe der Funktion vergleichen wir dann mit dem gegebenen Hash. Nach einiger Iterationen finden wir tatsächlich heraus, dass das Passwort **s1v3s** lautet. (Für Code siehe *pwcrack.py*)

2.3 Eigene Kennwort-Speicherfunktion in Java