htw saar

Studiengang Kommunikationsinformatik
Prof. Dr.–Ing. Damian Weber
Dipl.-Inf. Marion Bohr
Sarah Theobald, M.Sc.

Systemmanagement und Sicherheit

6. Übung

In dem vom Internet abgeschirmten Testnetz `172.16.0.0` sind virtuelle Maschinen `play01.local` bis `play100.local` eingerichtet, die vom Rechner `isl-s-01` aus erreichbar sind. Jedem Team ist eine Maschine `playnn` zugeordnet.

Das root–Passwort und Ihre Zuordnung zur virtuellen Maschine erhalten Sie in Ihrer Übungsgruppe.
Loggen Sie sich als `root` auf dem Ihnen zugeordneten `play`–Rechner ein.
Auf den `play`–Rechnern gibt es die Editoren `ee`, `vi` und `nano`, wobei `ee` der einfacher zu bedienende Editor sein könnte (die dort eingeblendete Hilfe verwendet das ∧–Symbol für die Strg–Taste).

Nachdem Ihr Benutzername in der Datei `vm.conf` von einem der Praktikumsbetreuer eingetragen wurde, erfolgt das Starten der virtuellen Maschine von `isl-s-01` aus mittels `play -s`

Das Beenden der virtuellen Maschine erfolgt (innerhalb der VM) mittels

```
shutdown -p now
```

oder alternativ außerhalb der VM mittels

```
play -p
```

von `isl-s-01` aus.

Aufgabe 1 (User–Accounts (1))

Legen Sie sich auf der play–Maschine einen eigenen persönlichen Login–Account an. Benutzen Sie hierfür Ihren Benutzernamen aus dem STL–Labor.

Siehe auch das FreeBSD-Handbook *Users and Basic Account Management*.

Setzen Sie für die folgende Aufgabe für `joe` Ihren Benutzernamen ein. Jedes Teammitglied soll dies für seinen Loginnamen durchführen.

Hier nun *beispielhaft* die Vorgehensweise für den Benutzer `joe`

- legen Sie eine neue Gruppe `joe` an,
- legen Sie einen neuen Benutzer `joe` an, dieser soll als Hauptgruppe die Gruppe `joe` haben; hierbei ist `vipw` zu benutzen
- geben Sie `joe` ein Homeverzeichnis und ein Passwort
- testen Sie den neuen Account mit Hilfe von `su` und `id`
- loggen Sie sich als `joe` von `isl-s-01` aus auf dem `play`-Rechner ein
- schreiben Sie als `joe` ein C–Programm, das die numerische User–ID und den Inhalt der Umgebungsvariable `PATH` ausgibt.

Legen Sie einen weiteren Benutzer `joex` mit Hilfe des `adduser` Kommandos an.

Legen Sie einen weiteren Benutzer `joey` mit Hilfe des `pw` Kommandos an.

Aufgabe 2 (User–Accounts (2))

Legen Sie mit Hilfe des `adduser`–Kommandos (Option `-f`) und einem von Ihnen geschriebenen Skript 50 Benutzer an, die (wie `joe` in Aufgabe 1) nun

- `joe01`,...,`joe50` heißen,
- Passwörter `passjoe01`,...,`passjoe50` haben,
- Homeverzeichnisse `/home/joe01`,...,`/home/joe50` besitzen
- als Login-Shell `/bin/tcsh` benutzen.

Bei dieser Aufgabe ist das `jot` Programm hilfreich. Siehe etwa das Ergebnis von

```
jot -w %02d 20 1 20
```

Aufgabe 3 (User–Accounts (3))

Erweitern Sie Ihr Skript aus Aufgabe 2, damit es sicherere Passwörter generiert.

Wenn Sie sicherere Passwörter generieren möchten, nutzen Sie

```
openssl rand -base64 6
```

Aufgabe 4 (at–Kommando)

Benutzen Sie als User `joe01` das at–Kommando, um einen Befehl in der Zukunft ausführen zu lassen. Mit dem `mailx` Kommando können Sie die e–Mail lesen, die die Standardausgabe des ausgeführten Kommandos enthält.

Aufgabe 5 (crontab–Kommando)

Benutzen Sie als User `joe01` das crontab–Kommando, um einen Befehl periodisch ausführen zu lassen.

Aufgabe 6 (limit/ulimit)

Diese Aufgabe kann auf dem `play`–Rechner oder dem `isl`–Rechner gelöst werden.

Mit Hilfe der Shell–Kommandos `limit` (aus der C–Shell) bzw. `ulimit` (aus der Bourne–Shell) können Prozesslimits gesetzt werden. Sie finden eine Beschreibung zu den Kommandos innerhalb der Manualpages zu `csh` und `sh`.

Setzen Sie ein Filesize–Limit, das Sie mutwillig mit dem `yes`–Kommando überschreiten wollen.

Setzen Sie ein CPU-Time–Limit, das Sie mutwillig mit einem eigenen C–Programm überschreiten wollen. Das C–Programm soll das `SIGXCPU` Signal abfangen und sich freiwillig beenden.

Setzen Sie die Limits mit beiden Programmen (`limit` und `ulimit`).

Zählen Sie die Anzahl Ihrer aktiven Prozesse mit Hilfe von `ps` und `wc`. Setzen Sie ein knapp darüber liegendes Limit für die maximale Anzahl an Prozessen, das Sie dann durch im Hintergrund zu startende Prozesse zu überschreiten versuchen.

Beachten Sie hierbei, dass

- `root` die maximale Anzahl von Prozessen beliebig überschreiten darf
- die Überschreitung dieses Limits in der `/var/log/messages` Datei protokolliert wird

Aufgabe 7 (Service aktivieren (inetd/telnet))

Für dieses Aufgabe benötigen Sie den `play`–Rechner mit `root`–Zugang. Es empfiehlt sich, hierfür ein Terminalfenster offenzuhalten.

Auf dem `play`–Rechner mit `root`–Zugang ist der `inetd` Server installiert. Dies ist der Internet–Superserver, der Ports für konfigurierte Dienste öffnet und beim Eintreffen einer Client–Nachricht für den Aufruf des entsprechenden Protokollservers sorgt.

Sorgen Sie dafür, daß er beim Booten aufgerufen wird. Hierzu setzen Sie die Umgebungsvariable `inetd_enable` in `/etc/rc.conf`.

Erzwingen Sie durch ein geeignetes `shutdown`–Kommando, daß die Maschine nach 1 Minute bootet. Überprüfen Sie danach, daß der `inetd` Server nach dem Booten läuft.

Editieren Sie die Datei `/etc/inetd.conf` derart, daß die Zeile für den Service `telnet` aktiv ist. Senden Sie dem `inetd` Server ein HUP–Signal, damit er seine Konfigurationsdatei erneut liest.

Überprüfen Sie mittels eines mittels `netstat`, daß der telnet–Port vorhanden ist.

Überprüfen Sie den `telnet`–Service mit einem Login–Vorgang mittels `telnet` von `isl-s-01` (beachten Sie, daß ein `root`–Login via `telnet` nicht möglich ist).

Aktivieren Sie weitere Services

- daytime-Service
- FTP-Service
- talk-Service

und überprüfen Sie erst deren Lauffähigkeit erst mit `netstat` und danach deren Funktionalität mit einem geeigneten Client.