UEB 09

1

a

Eine Zugriffskontrollmatrix ist ein Modell, das in Computersystemen zur Darstellung der Rechte von einzelnen Benutzern (in diesem Fall Alice und Bob) auf bestimmte Ressourcen (in diesem Fall die drei Dateien) verwendet wird.

Die Matrix könnte so aussehen:

Benutzer / Datei	hello world.txt	check world.sh	print hello world.sh
Alice	R, W	R, X	X
Bob	R	R, W	-

Hinweis:

- R steht für Lesen (Read)
- W steht für Schreiben (Write)
- X steht für Ausführen (Execute)
- "-" zeigt an, dass kein Zugriff gewährt wurde

In dieser Matrix kann Alice die Datei "hello world.txt" lesen und schreiben, die Datei "check world.sh" lesen und ausführen, und sie kann die Datei "print hello world.sh" ausführen. Bob hingegen kann die Datei "hello world.txt" lesen, kann in "check world.sh" lesen und schreiben, aber hat keinen Zugriff auf "print hello world.sh".

b)

Access Control Lists (ACLs) sind Listen, die auf jeder Ressource basieren und definieren, welcher Benutzer welche Zugriffsrechte auf die Ressource hat. Hier sind die ACLs für die genannten Dateien:

hello world.txt

- Alice: Lesen, Schreiben
- Bob: Lesen

check world.sh

- Alice: Lesen, Ausführen

- Bob: Lesen, Schreiben

print hello world.sh

- Alice: Ausführen
- Bob: Kein Zugriff

c)

Im Gegensatz zu ACLs, die auf Ressourcen basieren, basieren Capabilities auf Benutzern. Sie definieren, welche Ressourcen von einem bestimmten Benutzer zugegriffen werden können und welche Operationen ausgeführt werden können. Hier sind die Capabilities für Alice und Bob:

Alice

- hello world.txt: Lesen, Schreiben
- check world.sh: Lesen, Ausführen
- print hello world.sh: Ausführen

Bob

- hello world.txt: Lesen
- check world.sh: Lesen, Schreiben
- print hello world.sh: Kein Zugriff

Bei beiden Modellen handelt es sich um verschiedene Ansätze zur Darstellung von Zugriffsrechten in Computersystemen, wobei ACLs den Zugriff auf der Basis einzelner Ressourcen steuern und Capabilities auf Benutzerbasis arbeiten.

2

Um die Situation mit Linux-Dateisystemberechtigungen umzusetzen, müssen wir zuerst die Dateien erstellen und dann die entsprechenden Berechtigungen setzen. Bitte beachten Sie, dass in Linux Dateinamen ohne Leerzeichen üblicher sind. Ich werde Unterstriche anstelle von Leerzeichen verwenden, um den Prozess reibungslos zu gestalten.

a) Erstellen Sie eine Datei `hello_world.txt` mit dem Inhalt "Hello, world!".

```
echo "Hello, world!" > hello_world.txt
```

b) Erstellen Sie eine Datei `check_world.sh` mit dem Inhalt `cat hello_world.txt | grep -q 'world' && echo "'world' found"`.

```
echo "cat hello_world.txt | grep -q 'world' && echo "'world' found"" >
```

```
check_world.sh
```

c) Erstellen Sie eine Datei `print_hello_world.sh` mit dem Inhalt `cat hello_world.txt`.

```
echo "cat hello_world.txt" > print_hello_world.sh
```

Nun setzen wir die Berechtigungen entsprechend der Anforderungen. In Linux repräsentieren die Ziffern 4, 2 und 1 die Berechtigungen zum Lesen, Schreiben und Ausführen. Der Eigentümer (user), die Gruppe (group) und andere (other) haben jeweils eine dieser Ziffern.

Zum Beispiel: Die Berechtigung 7 (4+2+1) bedeutet, dass der Eigentümer lesen, schreiben und ausführen kann. Die Berechtigung 6 (4+2) bedeutet, dass der Eigentümer lesen und schreiben kann, aber nicht ausführen.

Gemäß der Zugriffskontrollmatrix:

- Alice kann `hello_world.txt` lesen und schreiben, `check_world.sh` lesen und ausführen und `print_hello_world.sh` ausführen.
- Bob kann `hello_world.txt` lesen, `check_world.sh` lesen und schreiben und hat keinen Zugriff auf `print_hello_world.sh`.

Nehmen wir an, Alice und Bob gehören zur gleichen Gruppe, dann könnten die Berechtigungen wie folgt gesetzt werden:

```
chmod 640 hello_world.txt      # Alice (der Eigentümer) kann lesen und
schreiben, Bob (in der Gruppe) kann nur lesen
chmod 750 check_world.sh       # Alice kann lesen und ausführen, Bob kann
lesen und schreiben
chmod 700 print_hello_world.sh # Alice kann lesen, schreiben und
ausführen, Bob hat keinen Zugriff
```

d) Erstellen Sie die Benutzer Alice und Bob mithilfe des Werkzeugs `adduser`.

```
sudo adduser alice
sudo adduser bob
```

Bei diesen Befehlen werden Sie nach weiteren Informationen gefragt, wie z.B. dem Passwort für die Benutzer.

e) Erstellen Sie eine Gruppe `shared` mithilfe des Werkzeugs `addgroup`.

```
sudo addgroup shared
```

f) Fügen Sie Alice und Bob zur Gruppe `shared` hinzu mithilfe des Werkzeugs `usermod`.

```
sudo usermod -a -G shared alice
sudo usermod -a -G shared bob
```

Der Parameter `-a` steht für 'append' (hinzufügen) und `-G` steht für 'groups' (Gruppen). Also fügt `usermod -a -G` den Benutzer zu der angegebenen Gruppe hinzu, ohne ihn aus anderen Gruppen zu entfernen.

g) Wir haben bereits die Dateien erstellt und nun müssen wir die entsprechenden Berechtigungen setzen. Da wir die Benutzer und die Gruppe erstellt haben, müssen wir auch die Eigentümerschaft der Dateien auf Alice setzen und die Gruppe der Dateien auf `shared` setzen:

```
sudo chown alice:shared hello_world.txt
sudo chown alice:shared check_world.sh
sudo chown alice:shared print_hello_world.sh
```

Jetzt setzen wir die Berechtigungen entsprechend der Anforderungen.

```
sudo chmod 640 hello_world.txt       # Alice (Eigentümer) kann lesen und
schreiben, Gruppe (shared) kann nur lesen
sudo chmod 760 check_world.sh        # Alice kann lesen, schreiben und
ausführen, Gruppe (shared) kann lesen und ausführen
sudo chmod 700 print_hello_world.sh  # Alice kann lesen, schreiben und
ausführen, Gruppe (shared) und andere haben keinen Zugriff
```

h) Testen Sie die von Ihnen erstellten Dateisystemberechtigungen.

Wir können testen, ob die Berechtigungen korrekt gesetzt sind, indem wir versuchen, die Dateien als Benutzer Alice und Bob zu lesen, zu schreiben und auszuführen. Angenommen, die Passwörter für Alice und Bob sind jeweils `alicepassword` und `bobpassword`, dann können wir die `su` (switch user) Befehle verwenden, um die Tests durchzuführen.
Als Alice:

```
su - alice -c "cat hello_world.txt"                        # Sollte
"Hello, world!" ausgeben
su - alice -c "echo 'Goodbye, world!' >> hello_world.txt"  # Sollte
erfolgreich sein
su - alice -c "./check_world.sh"                           # Sollte
"'world' found" ausgeben, da 'world' in der Datei hello_world.txt
existiert
su - alice -c "./print_hello_world.sh"                     # Sollte den
Inhalt der Datei hello_world.txt ausgeben
```

Als Bob:

```
su - bob -c "cat hello_world.txt"                              # Sollte
"Hello, world!" ausgeben
su - bob -c "echo 'Goodbye, world!' >> hello_world.txt"    # Sollte einen
Fehler ausgeben, da Bob nicht schreiben darf
su - bob -c "./check_world.sh"                                 # Sollte
"'world' found" ausgeben, da 'world' in der Datei hello_world.txt
existiert
su - bob -c "./print_hello_world.sh"                           # Sollte einen
Fehler ausgeben, da Bob nicht auf die Datei zugreifen darf
```