Künstler Burgy Zapp

von Don Denoncourt

Ein Freund nahm kürzlich eine Stelle in einem Unternehmen an, das iSeries Anwendungen unter WebSphere entwickelte. Als OS/400 und Java Experte hatte er das Gefühl, hier einen wertvollen Beitrag leisten zu können – bis zu dem Zeitpunkt, als das Unternehmen entschied, auf ein Sun Solaris System zu wechseln. Ich persönlich glaubte, dieser Wechsel würde meinem Freund eine exzellente Möglichkeit bieten, seine Kenntnisse auf ein neues Betriebssystem auszudehnen. Er allerdings konnte meine Begeisterung nicht teilen. Er hatte das Gefühl, seine Fähigkeiten auf einer ungewohnten Plattform nicht effektiv einsetzen zu können. Bald nach der Entscheidung kündigte er und wechselte zu einem anderen Unternehmen, das seine Entwicklungen auf einem iSeries System mit Java betrieb.

Ich stimme nicht mit der Überzeugung meines Freundes überein, dass es schwierig sei, vorhandenen OS/400 Skill auf ein anderes System zu übertragen. Leistungsfähige Betriebssysteme (voll multitaskingfähig, Multi-User-Plattform) zeichnen sich durch ein ähnliches Verhalten aus. Hat man einmal ein solides Verständnis eines solchen Systems entwickelt, gestaltet sich der Wechsel auf eine andere Plattform mit vergleichbaren Fähigkeiten relativ einfach. Man kann und sollte durchaus seine vorhandenen OS/400 Fähigkeiten auch auf andere Systemumgebungen portieren.

Eine der Plattformen, die man sicherlich erforschen sollte, ist Linux. Linux wurde, wie sicherlich bekannt, seitens IBM als strategisches Betriebssystem für die Zukunft spezifiziert. Bereits mit V5R1 kann Linux in einer logischen Partition betrieben werden. Mit dem Erlernen der Basisfunktionen von Linux lassen sich Linux-basierte Anwendungen relativ einfach im Unternehmen implementieren, Anwendungen, die in OS/400 in dieser Form nicht vorhandene Funktionen (wie z.B. Netzwerk-Security oder Bereitstellung von Multimedia-Inhalten) abdecken.

Analoge Funktionen, unterschiedliche Bezeichnungen

Wie schafft man es also, sich beim Arbeiten in einer Linux Umgebung komfortabel zu fühlen, wenn man so viele Jahre damit verbracht hat, AS/400 und iSeries Umgebungen zu erlernen und beherrschen. Der Schlüssel zur Erfolg ist das Bilden von Analogien zwischen beiden Plattformen! Viele der wesentlichen AS/400 Grundprinzipien (Benutzer, Benutzergruppen, Bibliothekslisten, Ausgabewarteschlangen, aktive Jobs) haben ein entsprechendes Linux Gegenstück. Die Linux

Bezeichnungen hierfür mögen von den vertrauten OS/400 Bezeichnungen abweichen; dennoch, wenn man ein solides Verständnis der Arbeitsweise dieser OS/400 Grundprinzipien entwickelt hat, fällt es nicht schwer, die gleichen Konzepte in einer Linux Umgebung mit den entsprechenden Anweisungen anzuwenden. Abbildung 1 zeigt eine Auflistung korrespondierender OS/400 und Linux Funktionen.

OS/400 verwendet Bibliotheken, um Objekte wie ausführbare Programme, Datenbankdateien, Indizes und sogar wiederum Bibliotheken darin zu speichern. Jeder Objekttyp verfügt über einen eigenständigen Satz von Anweisungen mit authentischem Informationsgehalt zum Erstellen, Löschen und Verändern individueller Objekte.

Die Objektorientierung unter Linux ist bei weitem nicht so differenziert gestaltet. Linux kennt ausschließlich Dateien – selbst ausführbare Programme sind schlicht Dateien. Üblicherweise reicht ein einziger Satz von Anweisungen aus, um alle Dateien und deren Existenzrechte zu verwalten. Die Dateien selbst enthalten keine authentischen Informationen. Statt dessen speichert das Linux Dateisystem für jede Datei einen minimalen Satz von Security-Attributen im Verzeichniseintrag ab. Andererseits verfügt Linux aber über eine hochflexible Verzeichnisstruktur. Verzeichnisse (das Gegenstück von OS/400 Bibliotheken) lassen sich in Baumstrukturen verschachtelt anlegen: Eine Datei befindet sich in einem Verzeichnis, das sich wiederum in einem Verzeichnis befindet, usw. Das Konzept ist identisch mit dem der PC-Verzeichnisstruktur (genau genommen haben die PC Softwareentwickler die Idee der verschachtelten Verzeichnisse von Unix übernommen).

Das Verschachteln von Verzeichnissen erfordert eine weit komplexere Dateibeschreibung, als dies bei OS/400 erforderlich ist. OS/400 kennzeichnet die Lage eines Objektes über eine einfache Bibliothek/Objekt-Beschreibung. Linux hingegen muss dem kompletten Pfad zu der Datei oder dem Verzeichnis vorhalten. Nachfolgend einige Beispiele üblicher Linux Verzeichnis-Pfade:

- /etc – In diesem Verzeichnispfad werden von Subsystemen (wie Networking oder Mail) benötigte Dateien gespeichert.
- /usr/local – Hier werden Anwendungsbibliotheken (wie Star Office, Java Development Kits, Web Application Server) gespeichert.
- /bin, /usr/bin und /usr/sbin – enthalten Dienstprogramme und Produkte
- /home beinhaltet Benutzerverzeichnisse und Dateien

Unter Linux lässt sich jede gewünschte Verzeichnisstruktur anlegen. Auch modifizierte Anweisungen lassen sich in diesen Verzeichnissen ablegen. Üblicherweise aber werden selbst erstellte Anweisungen in dem Verzeichnis /usr/local/bin abgelegt und Programmprodukte (z.B. Star Office) in Verzeichnissen unterhalb /usr/local.

Linux verwendet überdies ein Konzept mit der Bezeichnung Pfadliste (*path list*), das der in OS/400 implementierten Bibliotheksliste ähnlich ist. Die Pfadliste wird in der *global environment* Variablen $PATH gespeichert. Sie lässt sich mit der Anweisung *echo* anzeigen:

```
echo $PATH
/usr/local/bin:/usr/etc:/usr/sbin:
/usr/bsd:/sbin:/usr/bin /bin
```

Die Pfadliste ist schlicht eine Auflistung der Verzeichnispfade, so wie die OS/400 Bibliotheksliste eine Auflistung von Bibliotheksnamen ist. In beiden Fällen verfügt jede ausführbare Instanz – ein *job* unter OS/400 und ein *process* unter Linux – über eine eigenständige Bibliotheksliste oder Pfadliste. OS/400 Bibliothekslisten und Linux Pfadlisten unterscheiden sich jedoch in ihren Möglichkeiten. OS/400 verwendet Bibliothekslisten zum Auffinden jeglicher Objekte, Linux hingegen verwendet

Pfadlisten ausschließlich zum Auffinden ausführbarer Dateien. Hier ist OS/400 klar im Vorteil. Für einen Programmtest ist es beispielsweise möglich, eine Anwendung mit Testdaten in einer Testbibliothek arbeiten zu lassen, indem man einfach die Testbibliothek oberhalb der Produktionsbibliothek in die Bibliotheksliste einfügt. Eine vergleichbare Möglichkeit ist unter Linux nicht vorhanden.

Entsprechungen in der Security

Ich hatte zuvor schon auf die im Vergleich zu OS/400 unter Linux schlichtere Security-Implementierung angespielt. Unter Linux verfügt jede Datei und jedes Verzeichnis über einen Satz von drei Berechtigungsdefinitionen: eine für den Eigentümer, eine für eine benannte Gruppe von Benutzern und eine für den allgemeinen Zugriff. Es gibt nur drei mögliche Berechtigungsstufen: den Lesezugriff, den Schreibzugriff und die Berechtigung zur Ausführung. Die Anweisung ls -l erzeugt eine Verzeichnisauflistung in Langform. Werden Verzeichnisse in dieser Form angezeigt, erscheinen die Security-Attribute auf der linken Seite der Auflistung als eine Reihe von einstelligen Markierungen:

```
ls -l mydir
drwxr-xr-x root sys     33 Oct26 2000  bin
drwxrwxr-x tom  staff 4096 Oct11 08:40 cusdb
-rw-r--r-- tom  staff 9352 Nov04 2000  imprt
```

In diesem Beispiel sind Verzeichnisse mit dem Buchstaben „**d**“ gekennzeichnet. Die erste Dreiergruppe der Markierungen kennzeichnet die Lese- Schreib- und Ausführungsrechte für den Eigentümer der Datei oder des Verzeichnisses. Die zweite Gruppierung zeigt die Berechtigungen der zugewiesenen Benutzergruppe, die dritte Gruppierung zeigt die für alle Benutzer vergebenen Berechtigungen. Der erste Begriff nach den Berechtigungsmarkierungen weist den Eigentümer aus, der zweite Begriff kennzeichnet die Gruppen-ID, der die Datei oder das Verzeichnis zugeordnet ist.

Auch hier gibt es Übereinstimmungen. Die Benutzer-ID ***root*** ist äquivalent zu der Benutzer-ID QSECOFR unter OS/400 zu sehen. Beide verfügen über sämtliche Berechtigungen an allen Objekten. Aber es gibt einen entscheidenden Unterschied zwischen Linux ***root*** und OS/400 QSECOFR: Unter Linux existiert – im Gegensatz zu OS/400 – annähernd kein Schutz gegen versehentliches Löschen von Daten. Ist man an einem Linux System als ***root*****–**user angemeldet, führt das System jede eingegebene Anweisung ohne weitere Rückfrage aus, gleichgültig wie verheerend die Auswirkungen auch sein mögen. So sollte man unbedingt darauf achten, z.B. die folgende, nur neun Zeichen umfassende Anweisung, **niemals** auszuführen. Diese Anweisung löscht tatsächlich das gesamte Betriebssystem sowie sämtliche Daten und Benutzerobjekte von einem laufenden Linux-System:

```
rm -rf /*
```

Um unbeabsichtigte Katastrophen zu vermeiden, sollte man möglichst wenige Dinge tun, so lange man mit der Benutzer-ID ***root*** angemeldet ist.

In der obigen beispielhaften Verzeichnisauflistung hat ausschließlich die Benutzer-ID ***root*** als Eigentümer Schreibberechtigung für das Verzeichnis **bin**, alle anderen Benutzer haben nur Lese- und Ausführungsberechtigung. Bei Anwendung auf ein Verzeichnis erlaubt die Ausführungsberechtigung die Navigation innerhalb des Verzeichnisses. Für das Verzeichnis **cusdb**, dessen Eigentümer der Benutzer ***tom*** ist, haben sowohl ***tom*** selbst als auch die Benutzer der Benutzergruppe „**staff**“ Schreibberechtigung. Alle anderen Benutzer verfügen nur über Lese- und Ausführungsberechtigung. Für die Datei **imprt** hat nur der Benutzer ***tom*** Lese- und Schreibberechtigung, alle anderen können diese Datei nur lesen. Da für diese Datei keine

Ausführungsberechtigung vorliegt, kann sie nicht als Programm ausgeführt werden. Die Datei **imprt** enthält nur Anwendungsdaten.

Das Verzeichnis /etc ist angefüllt mit wichtigen Dateien, viele davon beziehen sich auf die System-Security. Eine der interessanteren Dateien in diesem Verzeichnis ist die Datei **passwd** (*password*). Lässt man sich diese Datei anzeigen, erhält man eine Auflistung, die der durch die OS/400 Anweisung WRKUSRPFR (Work with User Profile) mit der Option *ALL erzeugten Auflistung ähnlich ist. Die Datei **passwd** lässt sich mit der Anweisung **cat** (*concatenation*) anzeigen. Nachfolgend eine gekürzte Darstellung der erzeugten Auflistung:

```
cat /etc/passwd
root:x:0:0:root:/root:/bin/bash
ftp:x:14:50:FTP User:/home/ftp:
don:x:51:51:Denoncourt:/home/don:/bin/bash
```

Obwohl die Datei den Namen **passwd** trägt, speichert Linux die eigentlichen Kennwörter an anderer Stelle in einem verschlüsselten Format.

Wie bereits zuvor erwähnt, hat die Benutzer-ID **root** die gleiche Mächtigkeit wie die OS/400 Benutzer-ID QSECOFR. Der Benutzer **root** wird auch als Linux Super-User bezeichnet. Ist man als Benutzer mit niedrigeren Rechten angemeldet und möchte eine Anweisung ausführen, die Super-User Rechte erfordert, kann man über die Anweisung **su**, gefolgt vom **root** Kennwort die Sitzung für die Ausführung der Anweisung temporär mit der **root** Berechtigung ausstatten. Zum Hinzufügen von Benutzern dient die Anweisung **addusr** (*add user*), vergleichbar mit der OS/400 Anweisung CRTUSRPRF (*Create User Profile*).

Bei der Eingabe von OS/400 Anweisungen kann über die Funktionstaste F1 ein Online-Hilfetext angefordert werden. Unter Linux dient die Anweisung **man** (manual) zum Zugriff auf Hilfetexte. Zur Anforderung eines Hilfetextes zu der Anweisung **addusr** würde zum Beispiel die Eingabe von **man addusr** dienen.

Daraufhin erscheint eine Hilfe-Anzeige im Unix- Standardformat.

Wie OS/400, so verfügt auch Linux über Gruppenprofile. Diese Gruppenprofile sind in einer Datei namens **group** im Verzeichnis /etc gespeichert. Mit der Anweisung **cat** lassen sich die enthaltenen Gruppenprofile auflisten. Die Eintragungen für jedes Gruppenprofil sind – unterteilt durch Doppelpunkte – der Gruppen-Name, ein Platzhalter für das Kennwort, die Gruppen-ID und eine durch Kommata unterteilte Aufzählung der Benutzer-IDs, die dieser Gruppe zugeordnet sind. Die Definition von Benutzergruppen für Star Office und Programmiererzugriff könnten beispielshalber folgendermaßen aussehen:

```
Office:x:501:don,root
qpgmr:x:502:don,root,qpgmr
```

Die Datei-Zugriffsberechtigungen lassen sich für Benutzer mit der Anweisung **chmod** (*change file permissions*), für den Eigentümer einer Datei oder eines Verzeichnisses mit der Anweisung **chown** (*change owner*), für eine Gruppe, der eine Datei oder ein Verzeichnis zugeordnet ist, mit der Anweisung **chgrp** (*change group*) ändern.

OS/400 sieht die Möglichkeit vor, die Berechtigungen eines Benutzerprofils zu adoptieren. Linux verfügt über die gleiche Funktionalität über die Option **-s** in der Anweisung **chmod**. Mit der nachfolgenden Anweisung wird allen Benutzern die Berechtigung erteilt, ein bestimmtes Programm

ausführen zu dürfen:

```
chmod u-s programname
```

Aber Vorsicht: Die Anweisung wird mit dem gesamten Berechtigungsumfang des Eigentümers der Programmdatei ausgeführt. Dies kann unter Umständen nicht ungefährlich sein, speziell wenn **root** der Eigentümer ist.

Ensprechungen im Job Management

WRKACTJOB (*Work with active Jobs*) ist zweifelsfrei die „berüchtigtste“ OS/400 Anweisung. Ich bezeichne sie als „berüchtigt“, weil zu viele Programmierer sie zu häufig verwenden, obwohl hierbei reichlich wertvolle System-Ressourcen verbraucht werden. WRKACTJOB ist deshalb so populär, weil diese Anweisung eine wesentliche Funktionalität bietet: Sie listet alle aktiven OS/400 Jobs auf und weist dabei die von jedem einzelnen Job verbrauchten Ressourcen aus.

Die entsprechende Linux-Anweisung lautet **top** (show *process information in top-down order by CPU usage*). Abbildung 2 zeigt eine gekürzte Beispielauflistung, die von dieser Anweisung erzeugt wird. Die Anweisung **top** erzeugt eine nur einseitige Anzeige, die die Jobs enthält, die aktuell die meisten System-Ressourcen verbrauchen. Die Anzeige aller Jobs erreicht man mit der Anweisung **ps** (*process on current term*). Ohne weitere Zusätze zeigt die Anweisung **ps** nur die Jobs, die der jeweiligen Sitzung des Benutzers zugeordnet sind, zum Beispiel:

```
ps
PID  TTY   TIME     CMD
4449 pts/5 00:00:00 bash
4626 pts/5 00:00:00 ps
```

Die Anzeige aller Jobs lässt sich mit der Anweisung **ps** in Verbindung mit der Option **aux** aber auch in einer benutzerorientierten Form erreichen, wobei die Angabe des kontrollierenden Terminalnamens entfällt.