Kapitel 3

Kommandos und Interaktion mit dem OS

Übersicht

Bisher haben wir uns nur mit dem OS beschäftigt und dessen Interaktion mit der Hardware im Kontext des Betriebs eines Rechners. Ein wesentlicher Teil der Arbeit eines bzw. mit einem Rechner ist aber die Interaktion. Hierfür stellt der Betriebssystemkern Basisoperationen zur Verfügung indem es die hierfür notwendigen Hardwareressourcen wie Bildschirm und Tastatur verwaltet und überwacht und entsprechende Eingabe- und Ausgabeoperationen verwaltet und an das Gerät weiterleitet. Für eine richtige Interaktion sind aber dann Systemprogramme erforderlich, die über eine Shell als Kommandozeileninterpreter (ein weiteres Systemprogramm) bedient werden.

Lernziele
Nach Abschluss dieses Kapitels

- können Sie erste Befehle auf einer Shell in Linux anwenden
- kennen Sie das Grundkonzept einer Shell
- können Sie sich mit dem Linux Hilfesystemen behelfen

3.1 Die Shell

Das Betriebssystem im Sinne des Kernels selbst erlaubt noch keine direkte Nutzerinteraktion. Es verwaltet zwar Bildschirm und Tastatur, die Eingabe von Text im Sinne von Befehlen und Anweisungen würde aber noch keine Wirkung zeigen.

Wie im vorherigen Kapitel dargestellt werden über dem Betriebssystemkern Systemprogramme zur Verfügung gestellt, um bestimmte Aufgaben zu bewältigen. Um dem Nutzer über Tastatur

einen Zugriff auf diese Systemprogamme zu gewähren und deren Ergebnisse wieder auf dem Bildschirm anzuzeigen bedarf es aber einer Software, die eine Befehl interpretiert, darauf basierend das Systemprogrammaufruf und dessen Rückgabewerte auf dem Bildschirm ausgibt. Diese Software bzw. dieser Kommandozeileninterpreter wird als Shell bezeichnet. Der Name sagt dabei bereits, dass es sich um einen Interpreter handelt (vergleiche Konzept des Interpreters in den Programmiersprachen). Im Gegensatz zu Programmiersprachen ist aber die Hauptaufgabe eine interaktive Anwendung, also nicht durch einen in einer Programmdatei hinterlegten Programmcode, sondern durch interaktive Eingabe dieses Programmcodes in die Kommandozeile und direkte Ausführung und anschließende Anzeige von Rückgabewerten der Kommandos.

Eine Shell erfüllt dabei folgende Aufgaben:

- Wird nach dem Login als erstes Programm gestartet
- Bietet dem Nutzer einen Prompt zur Befehlseingabe
- Erhält vom OS die Eingaben der Tastatur zur Interpretation
- Interpretiert Eingaben und reicht diese als Systemaufrufe an das OS weiter (Start eines Systemprogramms/Programms)
- Übernahme der Rückgabewerte des Programmes vom Betriebssystem und Ausgabe am Bildschirm oder einem anderen Medium.

Moderne Shells gehen aber in ihrem Funktionsumfang weiter darüber hinaus:

- Funktionen zur Autovervollständigung von Kommandos (und Dateinamen)
- Bereitstellung von Variablenkonzepten, Schleifen, Verzweigungen und damit fast dem vollen Umfang einer Programmiersprache
- Möglichkeit der Speicherung von Kommandos in Verbindung mit Programmstrukturen in einer Datei und Aufruf als Skript (Shell-Skripte) - damit Nutzung des klassischen Konzeptes einer interpretierten Programmiersprache
- Nutzung von Wildcards (z . B. *)
- Kommandohistorie ($history)
- Eingebaute Kommandos für Rechenoperationen (Bsp.: $expr 13 + 7)

Auf Unix Betriebssystemen stehen dem Benutzer heutzutage eine Vielzahl von Shells zur Verfügung. Einige Beispiele dafür sind:

- Thompson-Shell (osh) 1971 - 1979
- Bourne-Shell (sh) 1977/1978 - Heute bzw. kompatible Erweiterungen
- Bourne-again-shell (bash) 1987 - Heute (Standard in den meißten unixoiden OS)
- uvm. z. B. Korn-Shell, Z-Shell, C-Shell, ...

3.2 Arbeiten auf der Konsole - Zugang zur Shell in den Terminals

Ein Linux/Unix System bietet neben der graphischen Benutzeroberfläche in der Regel auf mehrere Text-Konsolen an. Diese werden im System meist als tty1-tty6 referenziert. Um auf bzw. zwischen den Textkonsolen zu wechseln drücken Sie zusammen die Tastenkombination

<Strg> <Alt> <Funktionstaste>

Die Funktionstasten sind dabei in der Regel wie folgt belegt:

F1 - F6: Textkonsolen (tty1-6)
F7 : grafische Benutzeroberfläche unter Ubuntu

Hintergrund: Früher wurde Unix als Betriebssystem für Großrechner verwendet. User meldeten sich an Textterminals (Monitor und Tastatur) an. Diese Terminals waren über eine serielle Schnittstelle - im Unix System als tty bezeichnet - mit dem Großrechner verbunden. Die Textterminals und Großrechner sind verschwunden, die tty sind geblieben.

Sie können auf einem Unix-System an mehreren Konsolen/Terminals mit dem gleichen Benutzer oder unterschiedlichen Nutzern angemeldet sein.

Nach dem Login zeigt Ihnen das System ggf. Statusmeldungen zu Updates etc. an. Darauf werden wir ggf. später nochmals kommen.

Das System zeigt Ihnen nun eine Prompt. Dort können Sie Befehle absetzen, die von der Shell, einem kleinen Programm aufgenommen werden und zur Verarbeitung an die entsprechenden Programme weitergereicht werden.

Vor dem Prompt finden Sie übrigens wichtige Informationen zum aktuellen Nutzer (Nutzername) sowie dem Verzeichnis in dem Sie sich befinden. Mit dem Zeichen wird angezeigt, dass sich der Nutzer in seinem Home-Verzeichnis, also dem Verzeichnis für seine persönlichen Dateien befindet. Mehr dazu später.

3.3 Aufbau von Kommandos

Befehle oder Kommandos werden in Unix auf der Kommandozeile eingegeben. Die Kommandozeile ist eigentlich selbst ein Programm, das diese Eingabe entgegennimmt, interpretiert, das Programm zum Kommando aufruft und auch die Rückgabewerte wieder als Text auf dem Bildschirm in der Kommandozeile ausgibt.

Befehle bzw. Kommandos sind nach einem standardisierten Schema aufgebaut. Neben dem Befehlsnamen als Schlüsselwort können zusätzlich Optionen angegeben und Argumente übergeben werden, die der Befehl bei seinem Aufruf aufnimmt. Die Befehlsausführung wird dementsprechend verändert. Die Übergabe von Optionen und Argumenten haben wir schon kennengelernt:

```
1 # ls -la
2 cd  /home
```

Mit den Optionen l und a haben wir den Befehl ls dazu gebracht ausführlichere Informationen zur Verfügung zu stellen (l Option) sowie alle, also auch versteckte Dateien anzuzeigen. Dem cd Befehl haben wir das Argument /home übergeben, so dass er den Verzeichniswechsel in das Home-Verzeichnis vorgenommen hat.

Der Standardaufbau eines Unix-Befehls sieht damit wie folgt aus:

```
Befehlsnamen [-Optionen] [Argument1] [Argument2] ...
```

Optionen sind in der Regel Buchstaben, die mit einem Minuszeichen vorangestellt eine bestimmte Befehlsausführung herbeiführen. Jeder Buchstabe hat dabei stets seine eigene Bedeutung. Mehrere Buchstaben als Option bedeutet damit auch, dass der Befehl mit mehreren Optionen gleichzeitig aufgerufen wird. Argumente sind Werte, die an das auszuführende Programm übergeben werden. Für jeden Befehl ist dabei fest definiert, welche Parameter in welcher Form übergeben werden müssen.

Je nach Art des Befehls liefert der Befehl einen Rückgabewert (z.B. Auflistung des Verzeichnisinhalts beim ls-Befehl) oder schließt die Ausführung ohne Rückgabewert ab (z.B. cd-Befehl). In der Regel gibt der Befehl aber sofort nach Ausführung die Kontrolle wieder an die Shell ab und es erscheint wieder die Eingabeaufforderung und der nächste Befehl kann eingegeben werden.

Die wichtigsten Befehle unter Unix mit ihren wichtigsten Optionen und Parametern wird man nach kurzer Zeit und etwas Übung auf der Konsole auswendig wissen. Da aber viele der grundlegenden Befehle über 20 oder mehr Optionen verfügen und es in Summe einige dutzend Unix-Kommandos gibt, ist man nicht in der Lage sich alles zu merken. Hierfür bringt jedes Unix-System mehrere Hilfesysteme bereits als Bordmittel mit.

3.4 Hilfesystem

3.4.1 Hilfeoption der Befehle

Jeder Unix-Befehl ist mit einer eigenen Hilfe-Option aufrufbar. In der Regel liefert einer der folgenden Aufrufe eine knappe Zusammenfassung der Funktion eines Befehls, der möglichen Optionen und Parameter:

```
Befehlsname --help
```

oder

```
Befehlsname -h
```

oder

```
Befehlsname -?
```

Auf die Schnell lassen sich hier die wichtigsten Informationen finden.

3.4.2 Manpages

Eine ausführlichere Informationsquelle sind die sogenannten Manpages, die ein Hilfesystem für das gesamte System zur Verfügung stellen.Der Name leitet sich von Man wie manual (Handbuch) ab.

Um zu einem bestimmten Thema eine Manpage aufzurufen, wird das Programm man als Befehl aufgerufen und das entsprechende Theme, Dateiname oder Befehl als Argument an man übergeben:

```
man Befehlsname
```

Folgende Aufrufe als Beispiele:

```
1 $man ls
```

Liefert die Manpage für den ls-Befehl. Die Ausführungen sind hier sehr ausführlich, manchmal für den Anfänger sogar zu umfangreich und verwirrend. Die Bedienung des Programmes ist aber verhältnismäßig einfach. Mit den Bildauf und Bildab-Tasten bzw. den Pfeiltasten kann man in

der aufgerufenen Manpage navigieren. Mit der Taste q (quit) lässt sich das Programm man beenden.

Um mehr über die Bedienung und Nutzung der Manpages zu erfahren, gibt es auch hierfür eine eigene Manpage:

```
1 man man
```

Ruft man den man-Befehl mit der Option -k auf und hängt daran ein Schlüsselwort oder ein Wortfragment, werden alle Manpages aufgelistet, in denen sich das Suchwort findet.

Folgender Aufruf als Beispiel:

```
1 man -k rename
2
3
4
```

Die Manpages sind nach einem standardisierten Schema gegliedert:

1. Einfache Kommandos der Anwendungsebene
2. Systemaufrufe für Programmierer
3. Bibliotheksaufrufe für Programmierer
4. Spezialdateien für den Zugriff auf Hardware in /dev
5. Dateiformate und -aufbauten, beispielsweise printcap
6. Spiele
7. Makropakete und Konventionen. Hier findet sich beispielsweise eine Beschreibung von X oder die Definition der verschiedenen Zeichensätze.
8. Systemadministrationsbefehle (in der Regel nur für root)

Die in den Manpages verwendete Schreibweise für Optionen und Argumente liest sich bei genauerem Hinsehen sehr einfach:

[optional] optional, kann also, muss aber nicht angegeben werden

{dies | das} Hier muss entweder dies oder das stehen.

<variablen> Hier steht eine zu benennende Variable, also nicht das Wort variable, sondern ein Argument.

3.4.3 Weitere Hilfesysteme und Dokumente

Für viele Arbeiten an einem Unix-System gibt es Anleiten, sogenannte HowTos. Auf vielen System sind diese auch vorinstalliert und befinden sich in dem Ordner

/usr/doc oder unter /usr/share/doc

Auch im Internet kann man die Howtos nachschlagen:

http://www.linuxdoc.org

Eine Zusammenfassung wichtiger Befehle findet man häufig in sogenannten "Cheat Sheets". Ein Beispiel:

Google-Suche: linux bash cheat sheet

Übung

- Rufen Sie die Hilfeoption für die Befehle ls und rm auf und versuchen Sie die Information zu finden, wie man sich mit ls alle Dateierweiterungen, Rechte etc. anzeigen lassen kann und wie mit rm auch rekursiv gelöscht werden kann.
- Sehen Sie sich die manpages für den Befehl man an.
- Werfen Sie einen Blick in die Manualpages für den Befehl ls.
- Versuchen Sie zum Stichwort zur Nutzung von config-Dateien mehr Infos in den Manpages zu finden. Für zwei Befehle/Anwendungen wird in den manpages die Bedeutung einer config beschrieben. Erproben Sie hierzu den Unterschied zwischen der f und k Option für man.