3.1 Lektion 1

Zertifikat:	Linux Essentials
Version:	1.6
Thema:	3 Die Macht der Befehlszeile
Lernziel:	3.1 Dateien mithilfe der Befehlszeile archivieren
Lektion:	1 von 1

Einführung

Komprimierung wird eingesetzt, um den Platzbedarf für einen bestimmten Datensatz zu reduzieren. Üblicherweise **dient Komprimierung der Einsparung von Speicherplatz** für eine Datei oder der Menge der über eine Netzwerkverbindung gesendeten Daten.

Kompression funktioniert durch das Ersetzen sich wiederholender Muster durch Daten. Angenommen in einem Roman kommen einige Wörter aus mehreren Zeichen extrem häufig vor, etwa das Wort “das”. Sie könnten die Größe des Romans deutlich reduzieren, indem Sie diese mehrstelligen Wörter und Muster durch Einzelzeichen ersetzen, zum Beispiel das “das” durch einen griechischen Buchstaben, der sonst im Text nicht vorkommt.

Kompression gibt es in **zwei Varianten**: ***verlustfrei* (*lossless*)** und ***verlustbehaftet* (*lossy*).** Dinge, die mit einem verlustfreien Algorithmus komprimiert wurden, lassen sich wieder in ihre ursprüngliche Form überführen. Daten, die mit einem verlustbehafteten Algorithmus komprimiert wurden, können nicht wiederhergestellt werden. **Verlustbehaftete Algorithmen werden oft für Bilder, Videos und Audiodateien verwendet**, bei denen der Qualitätsverlust für den Menschen unmerklich, für den Kontext irrelevant oder der Verlust den eingesparten Platz oder Netzwerkdurchsatz wert ist.

Archivierungswerkzeuge dienen dazu, Dateien und Verzeichnisse in einer einzigen Datei zusammenzufassen, wie etwa **Backups, Source Code und Langzeitarchive**.

Archive und Komprimierung gehen meist einher. Einige Archivierungswerkzeuge komprimieren sogar standardmäßig ihren Inhalt, andere komprimieren ihren Inhalt optional. Einige Archivierungswerkzeuge müssen mit eigenständigen Komprimierungswerkzeugen kombiniert werden, wenn Sie den Inhalt komprimieren möchten.

Das **gebräuchlichste Werkzeug** zur Archivierung von Dateien **auf Linux-Systemen** ist `tar`. Die meisten Linux-Distributionen werden mit der GNU-Version von `tar` ausgeliefert, darum behandeln wir es auch in dieser Lektion. **`tar` verwaltet** lediglich die **Archivierung von Dateien, komprimiert diese aber nicht**.

Es gibt viele Komprimierungswerkzeuge unter Linux, einige gängige verlustfreie sind **`bzip2`, `gzip` und `xz`**. Alle drei finden Sie auf den meisten Systemen, können aber auch auf ein altes oder reduziertes System stoßen, auf dem `xz` oder `bzip` nicht installiert sind. Mit hoher Wahrscheinlichkeit werden Sie aber auf Dateien treffen, die mit allen drei Werkzeugen komprimiert wurden. Alle drei verwenden unterschiedliche Algorithmen, so dass eine **mit einem Tool komprimierte Datei nicht von einem anderen dekomprimiert werden kann.** Bei allen Komprimierungswerkzeugen muss

man Kompromisse eingehen: Wenn Sie eine **hohe Kompressionsrate** wünschen, **dauert es länger,** die Datei zu komprimieren und zu dekomprimieren. Das liegt daran, dass eine höhere Kompression mehr Aufwand erfordert, komplexere Muster zu finden. Die genannten Tools komprimieren Daten, können aber keine Archive mit mehreren Dateien erstellen.

Selbständige Komprimierungswerkzeuge sind in der Regel auf Windows-Systemen nicht verfügbar. Windows-Archivierungs- und Komprimierungswerkzeuge sind meist miteinander kombiniert. Bedenken Sie dies, wenn Sie Linux- und Windows-Systeme haben, die Dateien gemeinsam nutzen müssen.

Linux-Systeme haben auch Werkzeuge für die Verwaltung von `.zip`-Dateien, die auf Windows-Systemen üblich sind: Sie heißen `zip` und `unzip`, werden aber nicht auf allen Systemen standardmäßig installiert, so dass Sie sie gegebenenfalls installieren müssen. Glücklicherweise finden sie sich aber in den Paketsammlungen der meisten Distributionen.

Komprimierungswerkzeuge

Wie viel Speicherplatz durch die Komprimierung von Dateien eingespart wird, hängt von einigen Faktoren ab: der Art der zu komprimierenden Daten, dem Algorithmus zur Komprimierung der Daten und der Komprimierungsstufe. Nicht alle Algorithmen unterstützen unterschiedliche Komprimierungsstufen.

Legen wir zunächst einige Testdateien zur Komprimierung an:

```
$ mkdir ~/linux_essentials-3.1
$ cd ~/linux_essentials-3.1
$ mkdir compression archiving
$ cd compression
$ cat /etc/* > bigfile 2> /dev/null
```

Jetzt erstellen wir drei Kopien dieser Datei:

```
$ cp bigfile bigfile2
$ cp bigfile bigfile3
$ cp bigfile bigfile4
$ ls -lh
total 2.8M
-rw-r--r-- 1 emma emma 712K Jun 23 08:08 bigfile
-rw-r--r-- 1 emma emma 712K Jun 23 08:08 bigfile2
-rw-r--r-- 1 emma emma 712K Jun 23 08:08 bigfile3
-rw-r--r-- 1 emma emma 712K Jun 23 08:08 bigfile4
```

Nun komprimieren wir die Dateien mit jedem der oben genannten Kompressionstools:

```
$ bzip2 bigfile2
$ gzip bigfile3
$ xz bigfile4
$ ls -lh
total 1.2M
-rw-r--r-- 1 emma emma 712K Jun 23 08:08 bigfile
-rw-r--r-- 1 emma emma 170K Jun 23 08:08 bigfile2.bz2
-rw-r--r-- 1 emma emma 179K Jun 23 08:08 bigfile3.gz
-rw-r--r-- 1 emma emma 144K Jun 23 08:08 bigfile4.xz
```

Vergleichen Sie die Größen der komprimierten Dateien mit der unkomprimierten Datei namens `bigfile` und beachten Sie, dass die Komprimierungswerkzeuge Erweiterungen zu den Dateinamen hinzugefügt und die unkomprimierten Dateien entfernt haben.

Verwenden Sie `bunzip2`, `gunzip` oder `unxz`, um die Dateien zu dekomprimieren:

```
$ bunzip2 bigfile2.bz2
$ gunzip bigfile3.gz
$ unxz bigfile4.xz
$ ls -lh
total 2.8M
-rw-r--r-- 1 emma emma 712K Jun 23 08:20 bigfile
-rw-r--r-- 1 emma emma 712K Jun 23 08:20 bigfile2
-rw-r--r-- 1 emma emma 712K Jun 23 08:20 bigfile3
-rw-r--r-- 1 emma emma 712K Jun 23 08:20 bigfile4
```

Beachten Sie, dass nun die komprimierte Datei gelöscht wird, sobald sie dekomprimiert wurde.

Einige Komprimierungswerkzeuge unterstützen unterschiedliche Komprimierungsstufen. Eine **höhere Komprimierungsstufe erfordert in der Regel mehr Speicher und CPU-Zyklen**, führt aber zu einer kleineren komprimierten Datei. Für niedrigere Stufen gilt entsprechend das Gegenteil. Hier ein Beispiel mit `xz` und `gzip`:

```
$ cp bigfile bigfile-gz1
$ cp bigfile bigfile-gz9
$ gzip -1 bigfile-gz1
$ gzip -9 bigfile-gz9
$ cp bigfile bigfile-xz1
$ cp bigfile bigfile-xz9
$ xz -1 bigfile bigfile-xz1
$ xz -9 bigfile bigfile-xz9
$ ls -lh bigfile bigfile-* *
total 3.5M
-rw-r--r-- 1 emma emma 712K Jun 23 08:08 bigfile
-rw-r--r-- 1 emma emma 205K Jun 23 13:14 bigfile-gz1.gz
-rw-r--r-- 1 emma emma 178K Jun 23 13:14 bigfile-gz9.gz
-rw-r--r-- 1 emma emma 156K Jun 23 08:08 bigfile-xz1.xz
-rw-r--r-- 1 emma emma 143K Jun 23 08:08 bigfile-xz9.xz
```

Es ist **nicht notwendig**, eine Datei **bei jeder Verwendung zu dekomprimieren**. Komprimierungswerkzeuge verfügen in der Regel über spezielle Versionen gängiger Werkzeuge zum Lesen von Textdateien, wie z.B. `gzip` mit `cat`, `grep`, `diff`, `less`, `more` und einigen anderen. Bei `gzip` ist den Werkzeugen ein `z` vorangestellt, bei `bzip2` das `bz` und bei `xz` das Präfix `xz`:

```
$ cp /etc/hosts ./
$ gzip hosts
$ zcat hosts.gz
127.0.0.1       localhost

# The following lines are desirable for IPv6 capable hosts
::1     localhost ip6-localhost ip6-loopback
ff02::1 ip6-allnodes
ff02::2 ip6-allrouters
```

Archivierungswerkzeuge

Das Programm `tar` ist wohl das am weitesten verbreitete Archivierungswerkzeug auf Linux-Systemen. **Der Name ist übrigens eine Abkürzung für "Tape Archive".** Dateien, die mit `tar` erstellt wurden, werden oft als *tar balls* bezeichnet. Es ist sehr verbreitet, den Quellcode von Anwendungen in tar balls zur Verfügung zu stellen.

Die GNU-Version von `tar`, mit der Linux-Distributionen ausgeliefert werden, hat viele Optionen. In dieser Lektion stellen wir die am häufigsten verwendeten vor.

Beginnen wir mit der Erstellung eines Archivs der zur Komprimierung vorgesehenen Dateien:

```
$ cd ~/linux_essentials-3.1
$ tar cf archiving/3.1.tar compression
```

Die Option `c` weist `tar` an, eine neue Archivdatei zu erstellen, und die Option `f` ist der Name der zu erstellenden Datei: Das Argument unmittelbar nach den Optionen ist immer der Name der zu bearbeitenden Datei. Die übrigen Argumente sind die Pfade zu allen Dateien oder Verzeichnissen, die Sie der Datei hinzufügen, auflisten oder extrahieren möchten. In dem Beispiel fügen wir das Verzeichnis `compression` samt Inhalt dem Archiv hinzu.

Um den Inhalt eines tar balls zu sehen, verwenden Sie die Option `t` von `tar`:

```
$ tar -tf 3.1.tar
compression/
compression/bigfile-xz1.xz
compression/bigfile-gz9.gz
compression/hosts.gz
compression/bigfile2
compression/bigfile
compression/bigfile-gz1.gz
compression/bigfile-xz9.xz
compression/bigfile3
compression/bigfile4
```

Beachten Sie, wie den Optionen `-` vorangestellt wird. Im Gegensatz zu den meisten Programmen ist bei `tar` das `-` bei der Angabe von Optionen nicht erforderlich, hat aber auch keine negativen Folgen.

Note Nutzen Sie die Option `-v`, um `tar` die Namen der Dateien ausgeben zu lassen, mit denen es beim Erstellen oder Extrahieren eines Archivs arbeitet.

Lassen Sie uns nun die Datei entpacken:

```
$ cd ~/linux_essentials-3.1/archiving
$ ls
3.1.tar
$ tar xf 3.1.tar
$ ls
3.1.tar  compression
```

Angenommen, Sie benötigen nur eine Datei aus dem Archiv — in diesem Fall geben Sie diese hinter dem Dateinamen des Archivs an. Bei Bedarf können Sie auch mehrere Dateien angeben:

```
$ cd ~/linux_essentials-3.1/archiving
$ rm -rf compression
$ ls
3.1.tar
$ tar xvf 3.1.tar compression/hosts.gz
compression/
compression/bigfile-xz1.xz
compression/bigfile-gz9.gz
compression/hosts.gz
```

```
compression/bigfile2
compression/bigfile
compression/bigfile-gz1.gz
compression/bigfile-xz9.xz
compression/bigfile3
compression/bigfile4
$ ls
3.1.tar  compression
$ ls compression
hosts.gz
```

Mit Ausnahme von absoluten Pfaden (also solchen, die mit `/` beginnen), behalten `tar`-Dateien den gesamten Pfad zu Dateien bei, wenn sie erstellt werden. Da die Datei `3.1.tar` mit einem einzigen Verzeichnis erstellt wurde, wird dieses Verzeichnis beim Extrahieren relativ zu Ihrem aktuellen Arbeitsverzeichnis erstellt. Ein weiteres Beispiel soll dies verdeutlichen:

```
$ cd ~/linux_essentials-3.1/archiving
$ rm -rf compression
$ cd ../compression
$ tar cf ../tar/3.1-nodir.tar *
$ cd ../archiving
$ mkdir untar
$ cd untar
$ tar -xf ../3.1-nodir.tar
$ ls
bigfile   bigfile3  bigfile-gz1.gz  bigfile-xz1.xz  hosts.gz
bigfile2  bigfile4  bigfile-gz9.gz  bigfile-xz9.xz
```

Tip Wenn Sie den absoluten Pfad in einer `tar` Datei verwenden möchten, nutzen Sie die Option `P`. Beachten Sie, dass dies wichtige Dateien überschreiben und Fehler auf Ihrem System verursachen kann.

`tar` kann auch Komprimierung und Dekomprimierung von Archiven on the fly durchführen, indem es eines der oben beschriebenen Komprimierungswerkzeuge aufruft. Dazu ist lediglich die dem Komprimierungsalgorithmus entsprechende Option hinzuzufügen. Die am häufigsten verwendeten sind `j`, `J` und `z` für `bzip2`, `xz` und `gzip`. Im Folgenden einige Beispiele:

```
$ cd ~/linux_essentials-3.1/compression
$ ls
bigfile   bigfile3  bigfile-gz1.gz  bigfile-xz1.xz  hosts.gz
bigfile2  bigfile4  bigfile-gz9.gz  bigfile-xz9.xz
$ tar -czf gzip.tar.gz bigfile bigfile2 bigfile3
$ tar -cjf bzip2.tar.bz2 bigfile bigfile2 bigfile3
$ tar -cJf xz.tar.xz bigfile bigfile2 bigfile3
$ ls -l | grep tar
-rw-r--r-- 1 emma emma  450202 Jun 27 05:56 bzip2.tar.bz2
-rw-r--r-- 1 emma emma  548656 Jun 27 05:55 gzip.tar.gz
-rw-r--r-- 1 emma emma  147068 Jun 27 05:56 xz.tar.xz
```

Sie sehen, dass im Beispiel die `.tar`-Dateien unterschiedlich groß sind. Das zeigt, dass sie erfolgreich komprimiert wurden. Wenn Sie komprimierte `.tar`-Archive erstellen, sollten Sie immer eine zweite Dateierweiterung hinzufügen, die den von Ihnen verwendeten Algorithmus angibt: `.xz`, `.bz` und `.gz` für `xz`, `bzip2` bzw. `gzip`. Manchmal werden verkürzte Erweiterungen wie `.tgz` verwendet.

Es ist möglich, Dateien zu bereits vorhandenen unkomprimierten `tar`-Archiven hinzuzufügen, und zwar über die Option `u`. Wenn Sie versuchen, sie zu einem komprimierten Archiv hinzuzufügen, erhalten Sie eine Fehlermeldung.

```
$ cd ~/linux_essentials-3.1/compression
$ ls
bigfile   bigfile3  bigfile-gz1.gz  bigfile-xz1.xz  bzip2.tar.bz2  hosts.gz
bigfile2  bigfile4  bigfile-gz9.gz  bigfile-xz9.xz  gzip.tar.gz
xz.tar.xz
$ tar cf plain.tar bigfile bigfile2 bigfile3
$ tar tf plain.tar
bigfile
bigfile2
bigfile3
$ tar uf plain.tar bigfile4
$ tar tf plain.tar
bigfile
bigfile2
bigfile3
bigfile4
$ tar uzf gzip.tar.gz bigfile4
tar: Cannot update compressed archives
Try 'tar --help' or 'tar --usage' for more information.
```

Verwalten von ZIP-Dateien

Windows-Maschinen verfügen oft nicht über Anwendungen zur Verarbeitung von tar balls und andere auf Linux-Systemen übliche Kompressionstools. Wenn Sie mit Windows-Systemen interagieren müssen, können Sie ZIP-Dateien verwenden. Eine ZIP-Datei ist eine Archivdatei, die einer komprimierten `tar`-Datei ähnelt.

Sie können die Programme `zip` und `unzip` für die Arbeit mit ZIP-Dateien auf Linux-Systemen nutzen. Das folgende Beispiel zeigt, was Sie dafür benötigen:

```
$ cd ~/linux_essentials-3.1
$ mkdir zip
$ cd zip/
$ mkdir dir
$ touch dir/file1 dir/file2
```

Jetzt verwenden wir `zip`, um diese Dateien in eine ZIP-Datei zu packen:

```
$ zip -r zipfile.zip dir
  adding: dir/ (stored 0%)
  adding: dir/file1 (stored 0%)
  adding: dir/file2 (stored 0%)
$ rm -rf dir
```

Anschließend entpacken wir die ZIP-Datei wieder:

```
$ ls
zipfile.zip
$ unzip zipfile.zip
Archive:  zipfile.zip
   creating: dir/
 extracting: dir/file1
 extracting: dir/file2
$ find
.
./zipfile.zip
./dir
./dir/file1
./dir/file2
```

Wenn Sie Verzeichnisse zu ZIP-Dateien hinzufügen, bewirkt die Option `-r`, dass `zip` auch den Inhalt eines Verzeichnisses enthält. Ohne die Option bliebe das Verzeichnis in der ZIP-Datei leer.

Geführte Übungen

1. Welches der folgenden Tools wurde entsprechend der jeweiligen Erweiterung zur Erstellung dieser Dateien verwendet?

Dateiname	**`tar`**	**`gzip`**	**`bzip2`**	**`xz`**
`archive.tar`				
`archive.tgz`				
`archive.tar.xz`				

2. Welche dieser Dateien sind entsprechend der jeweiligen Erweiterung Archive und welche sind komprimiert?

Dateiname	**Archiv**	**Komprimiert**
`file.tar`		
`file.tar.bz2`		
`file.zip`		
`file.xz`		

3. Wie würden Sie eine Datei zu einer mit `gzip` komprimierten `tar`-Datei hinzufügen?
4. Welche `tar`-Option weist `tar` an, den führenden `/` in absolute Pfade aufzunehmen?
5. Unterstützt `zip` verschiedene Kompressionsstufen?

Offene Übungen

1. Unterstützt `tar` beim Extrahieren von Dateien Globs in der Dateiliste?
2. Wie können Sie sicherstellen, dass eine dekomprimierte Datei mit der Datei identisch ist, bevor sie komprimiert wurde?
3. Was passiert, wenn Sie versuchen, eine Datei aus einem `tar`-Archiv zu extrahieren, die bereits auf Ihrem Dateisystem existiert?
4. Wie würden Sie die Datei `archive.tgz` extrahieren, ohne die `tar`-Option `z` zu verwenden?

Zusammenfassung

Linux-Systeme verfügen über mehrere Komprimierungs- und Archivierungswerkzeuge. In dieser Lektion wurden die gängigsten behandelt. Das meistgenutzte Archivierungswerkzeug ist `tar`. Wenn eine Interaktion mit Windows-Systemen erforderlich ist, können `zip` und `unzip` ZIP-Dateien erstellen und extrahieren.

Der Befehl `tar` hat einige Optionen, die man sich merken sollte: `x` steht für das Extrahieren, `c` für das Erstellen, `t` für das Anzeigen der Inhalte und `u` für das Hinzufügen oder Ersetzen von Dateien. `v` listet die Dateien auf, die von `tar` beim Erstellen oder Entpacken eines Archivs verarbeitet werden.

Das Repository der typischen Linux-Distribution verfügt über viele Komprimierungswerkzeuge, darunter `gzip`, `bzip2` und `xz`. Komprimierungsalgorithmen unterstützen oft verschiedene Ebenen, mit denen Sie die Geschwindigkeit oder Dateigröße optimieren. Dateien lassen sich mit `gunzip`, `bunzip2` und `unxz` dekomprimieren.

Komprimierungswerkzeuge umfassen häufig Programme, die sich wie gängige Textdateiwerkzeuge verhalten, mit dem Unterschied, dass sie mit komprimierten Dateien arbeiten. Einige von ihnen sind `zcat`, `bzcat` und `xzcat`. Komprimierungswerkzeuge werden normalerweise mit Programmen mit der Funktionalität von `grep`, `more`, `less`, `diff` und `cmp` ausgeliefert.

Befehle, die in den Übungen verwendet werden:

`bunzip2:` Dekomprimiert eine mit `bzip2` komprimierte Datei.

`Bzcat`: Gibt den Inhalt einer mit `bzip` komprimierten Datei aus.

`bzip2:` Komprimiert Dateien mit dem `bzip2`-Algorithmus und -Format.

`Gunzip:` Dekomprimiert eine mit `gzip` komprimierte Datei.

`Gzip:` Komprimiert Dateien mit dem `gzip`-Algorithmus und -Format.

`Tar:` Erstellt, aktualisiert, listet und extrahiert `tar`-Archive.

`Unxz:` Dekomprimiert eine mit `xz` komprimierte Datei.

`Unzip:` Dekomprimiert und extrahiert Inhalte aus einer ZIP-Datei.

`Xz:` Komprimiert Dateien mit dem `xz`-Algorithmus und -Format.

`Zcat:` Gibt den Inhalt einer mit `gzip` komprimierten Datei aus.

`Zip:` Erzeugt und komprimiert ZIP-Archive.