B

Access Control Lists unter Linux

Dieses Kapitel gibt einen kurzen Einblick in die Hintergründe und Funktionsweise von POSIX ACLs für Linux-Dateisysteme. Sie erfahren, wie das traditionelle Rechtekonzept für Dateisystemobjekte mit Hilfe von ACLs (*Access Control Lists*) erweitert wird und welche Vorteile dieses Konzept bietet.

Warum ACLs?

Hinweis

POSIX ACLs

Der Ausdruck „POSIX ACL" suggeriert, dass es sich um einen echten Standard aus der POSIX (*Portable Operating System Interface*) Familie handelt. Aus verschiedenen Gründen wurden die betreffenden Standardentwürfe POSIX 1003.1e und POSIX 1003.2c zurückgezogen. ACLs unter vielen UNIX-artigen Betriebssystemen basieren allerdings auf diesen Dokumenten. Die in diesem Kapitel beschriebene Implementierung von Dateisystem ACLs folgt den Inhalten dieser beiden Dokumente, die Sie unter folgender URL einsehen können:

`http://wt.xpilot.org/publications/posix.1e/`

Hinweis

Traditionell ist ein Dateiobjekt unter Linux mit drei Sets von Berechtigungen assoziiert. Diese Sets geben die Lese- (`r`), Schreib- (`w`) und Ausführungsrechte (`x`) für die drei Benutzerklassen Eigentümer der Datei (engl. *owner*), Gruppe (engl. *group*) und „Rest der Welt" (engl. *other*) wieder. Zusätzlich können noch die *set user id*, *set group id* und *sticky* Bits gesetzt werden. Mehr zu diesem Thema finden Sie im *Benutzerhandbuch* im Abschnitt *Benutzer und Zugriffsrechte*.

Für die meisten in der Praxis auftretenden Fälle reicht dieses schlanke Konzept völlig aus. Für komplexere Szenarien oder fortgeschrittenere Anwendungen mussten Systemadministratoren bisher eine Reihe von Tricks anwenden, um die Einschränkungen des traditionellen Rechtekonzepts zu umgehen.

In Situationen, in denen das traditionelle Dateirechte-Konzept nicht ausreicht, helfen ACLs. Sie erlauben es, einzelnen Benutzern oder Gruppen Rechte zuzuweisen, auch wenn diese nicht mit dem Eigentümer oder der Gruppe einer Datei übereinstimmen.

Access Control Lists sind ein Feature des Linux-Kernels und werden zur Zeit von ReiserFS, Ext2, Ext3, JFS und XFS unterstützt. Mit ihrer Hilfe können komplexe Szenarien umgesetzt werden, ohne dass auf Applikationsebene komplexe Rechtemodelle implementiert werden müssten.

Ein prominentes Beispiel für die Vorzüge von Access Control Lists ist der Austausch eines Windows-Servers gegen einen Linux-Server. Manche der angeschlossenen Workstations werden auch nach dem Umstieg weiter unter Windows betrieben werden. Das Linux-System bietet den Windows-Clients via Samba Datei- und Druckserver-Dienste an.

Da Samba Access Control Lists unterstützt, können Benutzerrechte sowohl auf dem Linux-Server als auch über eine grafische Benutzeroberfläche unter Windows (nur Windows NT und höher) eingerichtet werden. Über den winbindd ist es sogar möglich, Benutzern Rechte einzuräumen, die nur in der Windows-Domain existieren und über keinen Account auf dem Linux-Server verfügen. Auf der Serverseite können Sie die Access Control Lists mithilfe von getfacl und setfacl bearbeiten.

Definitionen

Benutzerklassen Das herkömmliche POSIX Rechtekonzept kennt drei *Klassen* von Benutzern für die Rechtevergabe im Dateisystem: Eigentümer (engl. *owner*), Gruppe (engl. *group*) und andere Benutzer oder den „Rest der Welt" (engl. *other*). Pro Benutzerklasse lassen sich jeweils die drei Berechtigungsbits (engl. *permission bits*) für Lesezugriff (`r`), für Schreibzugriff (`w`) und für Ausführbarkeit (`x`) vergeben. Eine Einführung in das Benutzerkonzept unter Linux finden Sie im *Benutzerhandbuch* im Abschnitt *Benutzer und Zugriffsrechte*.

Access ACL Die Zugriffsrechte für Benutzer und Gruppen auf beliebige Dateisystemobjekte (Dateien und Verzeichnisse) werden über Access ACLs (dt. *Zugriffs-ACLs*) festgelegt.

Default ACL Default ACLs (dt. *Vorgabe-ACLs*) können nur auf Verzeichnisse angewandt werden und legen fest, welche Rechte ein Dateisystemobjekt von seinem übergeordneten Verzeichnis beim Anlegen erbt.

ACL-Eintrag Jede ACL besteht aus einem Satz von ACL-Einträgen (engl. *ACL entries*). Ein ACL-Eintrag hat einen Typ (siehe Tabelle B.1 auf der nächsten Seite), einen Bezeichner für den Benutzer oder die Gruppe, auf die sich dieser Eintrag bezieht, und Berechtigungen. Der Bezeichner für Gruppe oder Benutzer bleibt für einige Typen von Einträgen leer.

Umgang mit ACLs

Im folgenden Abschnitt lernen Sie den Grundaufbau einer ACL und deren verschiedene Ausprägungen kennen. Der Zusammenhang zwischen ACLs und dem traditionellen Rechtekonzept im Linux-Dateisystem wird anhand mehrerer Grafiken kurz erläutert. An zwei Beispielen lernen Sie, selbst ACLs zu erstellen und deren korrekte Syntax zu beachten. Zuletzt erfahren Sie, nach welchem Muster ACLs vom Betriebssystem ausgewertet werden.

Aufbau von ACL-Einträgen

ACLs werden grundsätzlich in zwei Klassen eingeteilt. Eine *minimale* ACL besteht ausschließlich aus den Einträgen vom Typ *owner* (Besitzer), *owning group* (Besitzergruppe) und *other* (Andere), und entspricht den herkömmlichen Berechtigungsbits für Dateien und Verzeichnisse. Eine *erweiterte* (engl. *extended*) ACL geht über dieses Konzept hinaus. Sie muss einen *mask* (Maske) Eintrag enthalten und darf mehrere Einträge des Typs *named user* (namentlich gekennzeichneter Benutzer) und *named group* (namentlich gekennzeichnete Gruppe) enthalten. Tabelle B.1 fasst die verschiedenen verfügbaren Typen von ACL-Einträgen zusammen.

Typ	Textform
owner	`user::rwx`
named user	`user:name:rwx`
owning group	`group::rwx`
named group	`group:name:rwx`
mask	`mask::rwx`
other	`other::rwx`

Tabelle B.1: *Überblick: Typen von ACL-Einträgen*

In den Einträgen *owner* und *other* festgelegte Rechte sind immer wirksam. Vom *mask* Eintrag abgesehen, können alle übrigen Einträge (*named user*, *owning group* und *named group*) entweder wirksam oder maskiert werden. Sind Rechte sowohl in einem der oben genannten Einträge als auch in der Maske vorhanden, werden sie wirksam. Rechte, die nur in der Maske oder nur im eigentlichen Eintrag vorhanden sind, sind nicht wirksam. Das nachfolgende Beispiel verdeutlicht diesen Mechanismus (siehe Tabelle B.2):

Typ	Textform	Rechte
named user	`user:jane:r-x`	`r-x`
mask	`mask::rw-`	`rw-`
	Wirksame Berechtigungen:	`r--`

Tabelle B.2: *Maskierung von Zugriffsrechten*

ACL-Einträge und Berechtigungsbits

Die beiden Abbildungen illustrieren die beiden auftretenden Fälle einer minimalen und einer erweiterten ACL (siehe Abb. B.1 und B.2). Die Abbildungen gliedern sich in drei Blöcke. Links die Typangaben der ACL-Einträge, in der Mitte eine beispielhafte ACL und rechts die entsprechenden Berechtigungsbits, wie sie auch `ls -l` anzeigt.

In beiden Fällen werden die *owner class* Berechtigungen dem *owner* ACL-Eintrag zugeordnet. Die Zuordnung der *other class* Berechtigungen zum entsprechenden ACL-Eintrag ist ebenfalls konstant. Die Zuordnung der *group class* Berechtigungen variiert:

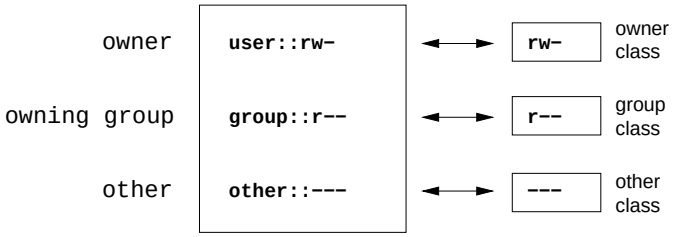

Abbildung B.1: *Minimale ACL: ACL-Einträge vs. Berechtigungsbits*

- Im Fall einer minimalen ACL — ohne *mask* Eintrag — werden die *group class* Berechtigungen dem *owning group* ACL-Eintrag zugeordnet (siehe Abb. B.1).
- Im Fall einer erweiterten ACL — mit *mask* Eintrag — werden die *group class* Berechtigungen dem *mask* Eintrag zugeordnet (siehe Abb. B.2).

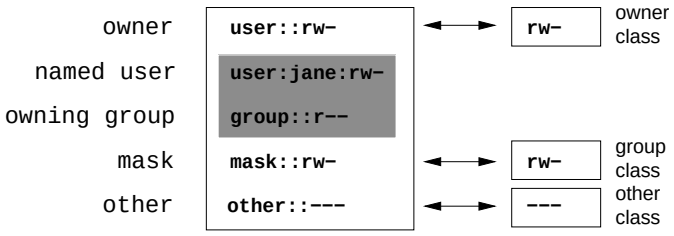

Abbildung B.2: *Erweiterte ACL: ACL-Einträge vs. Berechtigungsbits*

Durch diese Art der Zuordnung ist die reibungslose Interaktion von Anwendungen mit und solchen ohne ACL-Unterstützung gewährleistet. Die Zugriffsrechte, die mittels der Berechtigungsbits festgelegt wurden, sind die Obergrenze für alle anderen „Feineinstellungen", die per ACL gemacht werden. Alle Rechte, die hier nicht widergespiegelt sind, wurden entweder in der ACL nicht gesetzt oder sind nicht effektiv. Werden Berechtigungsbits geändert, spiegelt sich dies in der entsprechenden ACL wider und umgekehrt.

Ein Verzeichnis mit Access ACL

In drei Schritten werden Sie anhand folgendem Beispiel den Umgang mit einer Access ACL lernen:

- Anlegen eines Dateisystemobjekts (hier eines Verzeichnisses)
- Änderungen an der ACL
- Einsatz von Masken

1. Bevor Sie das Verzeichnis anlegen, können Sie mittels des `umask` Befehls festlegen, welche Zugriffsrechte von vorneherein maskiert werden sollen:

   ```
   umask 027
   ```

   `umask 027` beschränkt die Rechte der einzelnen Benutzergruppen folgendermaßen: der Besitzer der Datei behält sämtliche Rechte (`0`), die Besitzergruppe darf nicht lesend auf die Datei zugreifen (`2`) und alle anderen Benutzer erhalten keinerlei Zugriff (`7`). Die Zahlen sind als Bitmaske zu lesen. Details zu `umask` entnehmen Sie der entsprechenden Manpage (`man umask`).

   ```
   mkdir mydir
   ```

   Das Verzeichnis `mydir` ist angelegt und hat die durch die `umask` festgelegten Rechte erhalten. Mit

   ```
   ls -dl mydir drwxr-x--- ... tux projekt3 ... mydir
   ```

   überprüfen Sie, ob alle Rechte korrekt vergeben wurden.

2. Nachdem Sie sich über den Ausgangszustand der ACL informiert haben, fügen Sie ihr jeweils einen neuen Benutzer- und Gruppen-Eintrag hinzu.

```
getfacl mydir

# file: mydir
# owner: tux
# group: projekt3
user::rwx
group::r-x
other::---
```

Die Ausgabe von `getfacl` spiegelt exakt die unter Abschnitt *ACL-Einträge und Berechtigungsbits* auf Seite 561 beschriebene Zuordnung von Berechtigungsbits und ACL-Einträgen wider. Die ersten drei Zeilen der Ausgabe nennen Namen, Eigentümer und zugehörige Gruppe des Verzeichnisses. Die drei nächsten Zeilen enthalten die drei ACL-Einträge *owner*, *owning group* und *other*. Insgesamt liefert Ihnen der `getfacl` Befehl im Fall dieser einfachsten („minimalen") ACL keine Information, die Sie mittels `ls` nicht auch erhalten hätten.

Ihr erster Eingriff in die ACL besteht darin, einem zusätzlichen Benutzer `jane` und einer zusätzlichen Gruppen `djungle` Lese-, Schreib- und Ausführrechte zu gewähren.

```
setfacl -m user:jane:rwx,group:djungle:rwx mydir
```

Die Option `-m` weist `setfacl` an, die bestehende ACL zu modifizieren. Das nachfolgende Argument gibt an, welche ACL-Einträge geändert werden (mehrere werden durch Kommata voneinander getrennt). Abschließend geben Sie den Namen des Verzeichnisses an, für das diese Änderungen gelten sollen.

Die resultierende ACL geben Sie wieder mit `getfacl` aus.

```
getfacl mydir

# file: mydir
# owner: tux
# group: projekt3
user::rwx
user:jane:rwx
group::r-x
group:djungle:rwx
mask::rwx
other::---
```

Zusätzlich zu den von Ihnen initiierten Einträgen für den Benutzer `jane` und die Gruppe `djungle` wurde ein *mask* Eintrag erzeugt. Dieser *mask* Eintrag wird automatisch gesetzt, um alle Einträge in der *group class* auf einen gemeinsamen Nenner zu bringen. Außerdem passt `setfacl` bestehende *mask* Einträge an von Ihnen geänderte Einstellungen automatisch an, so Sie dieses Verhalten nicht mit `-n` deaktivieren. *mask* legt die maximal wirksamen Zugriffsrechte für alle Einträge innerhalb der *group class* fest. Dies beinhaltet: *named user*, *named group* und *owning group*. Die *group class* Berechtigungsbits, die ein `ls -dl mydir` ausgeben würde, entsprechen jetzt dem `mask`-Eintrag.

```
ls -dl mydir drwxrwx---+ ... tux projekt3 ... mydir
```

Es erscheint in der ersten Spalte der Ausgabe ein `+`, das auf eine *erweiterte* ACL hinweist.

3. Gemäß der Ausgabe des `ls` Kommandos beinhalten die Rechte für den *mask* Eintrag auch Schreibzugriff. Traditionell würden diese Berechtigungsbits auch darauf hinweisen, dass die *owning group* (hier: `projekt3`) ebenfalls Schreibzugriff auf das Verzeichnis `mydir` hätte. Allerdings sind die wirklich wirksamen Zugriffsrechte für die *owning group* als die Schnittmenge aus den für *owning group* und *mask* gesetzten Rechten definiert; also in unserem Beispiel `r-x` (siehe Tabelle B.2 auf Seite 560). Es hat sich auch nach Hinzufügen der ACL-Einträge nichts an den Rechten der *owning group* geändert.

   Verändern können Sie den *mask* Eintrag mittels `setfacl` oder `chmod`.

```
chmod g-w mydir
ls -dl mydir

drwxr-x---+ ... tux projekt3 ... mydir

getfacl mydir

# file: mydir
# owner: tux
# group: projekt3
user::rwx
user:jane:rwx          # effective: r-x
group::r-x
group:djungle:rwx      # effective: r-x
mask::r-x
other::---
```

Nachdem Sie per `chmod` die *group class* Bits um den Schreibzugriff verringert haben, liefert Ihnen schon die Ausgabe des `ls` Kommandos den Hinweis darauf, dass die *mask* Bits über `chmod` entsprechend angepasst wurden. Man erkennt, dass nur der Besitzer Schreibberechtigung im Verzeichnis `mydir` hat. Noch deutlicher wird dies an der Ausgabe von `getfacl`. `getfacl` fügt für alle Einträge Kommentare hinzu, deren tatsächlich wirksame Berechtigungsbits nicht mit den ursprünglich gesetzten übereinstimmen, weil sie vom *mask* Eintrag herausgefiltert werden. Sie können den Ausgangszustand mit dem entsprechenden `chmod` Kommando wiederherstellen:

```
chmod g+w mydir
ls -dl mydir

drwxrwx---+ ... tux projekt3 ... mydir

getfacl mydir

# file: mydir
# owner: tux
# group: projekt3
user::rwx
user:jane:rwx
group::r-x
group:djungle:rwx
mask::rwx
other::---
```

Ein Verzeichnis mit Default ACL

Verzeichnisse können mit einer besonderen Art von ACLs versehen werden; einer Default ACL. Diese Default ACL legt fest, welche Zugriffsrechte sämtliche Unterobjekte dieses Verzeichnisses zum Zeitpunkt der Erstellung erben. Eine Default ACL wirkt sich auf Unterverzeichnisse ebenso wie auf Dateien aus.

Auswirkungen einer Default ACL

Die Zugriffsrechte in einer Default ACL werden an Dateien und Unterverzeichnisse unterschiedlich vererbt:

- Ein Unterverzeichnis erbt die Default ACL des übergeordneten Verzeichnisses sowohl als seine eigene Default ACL als auch als Access ACL.
- Eine Datei erbt die Default ACL als ihre eigene Access ACL.

Alle Systemaufrufe (engl. *system calls*), die Dateisystemobjekte anlegen, verwenden einen `mode` Parameter. Der `mode` Parameter legt die Zugriffsrechte auf das neu anzulegende Dateisystemobjekt fest:

- Hat das übergeordnete Verzeichnis keine Default ACL, ergeben sich die Berechtigungen aus den im `mode`-Parameter angegebenen Berechtigungen, von denen die in der `umask` gesetzten Rechte abgezogen werden.
- Existiert eine Default ACL für das übergeordnete Verzeichnis, werden die Berechtigungsbits entsprechend der Schnittmenge aus dem Wert des `mode` Parameters und den in der Default ACL festgelegten Berechtigungen zusammengesetzt und dem Objekt zugewiesen. Die `umask` wird dabei nicht beachtet.

Default ACLs in der Praxis

Die drei folgenden Beispiele führen Sie an die wichtigsten Operationen an Verzeichnissen und Default ACLs heran:

- Anlegen einer Default ACL für ein bestehendes Verzeichnis
- Anlegen eines Unterverzeichnisses in einem Verzeichnis mit Default ACL
- Anlegen einer Datei in einem Verzeichnis mit Default ACL

1. Sie fügen dem schon existierenden Verzeichnis `mydir` eine Default ACL hinzu:

   ```
   setfacl -d -m group:djungle:r-x mydir
   ```

   Die `-d` Option des `setfacl` Kommandos weist `setfacl` an, die folgenden Modifikationen (Option `-m`) auf der Default ACL vorzunehmen.

   Sehen Sie sich das Ergebnis dieses Befehls genauer an:

   ```
   getfacl mydir
   ```

```
# file: mydir
# owner: tux
# group: projekt3
user::rwx
user:jane:rwx
group::r-x
group:djungle:rwx
mask::rwx
other::---
default:user::rwx
default:group::r-x
default:group:djungle:r-x
default:mask::r-x
default:other::---
```

`getfacl` liefert sowohl die Access ACL als auch die Default ACL zurück. Alle Zeilen, die mit `default` beginnen, bilden zusammen die Default ACL. Obwohl Sie dem `setfacl` Befehl lediglich einen Eintrag für die Gruppe `djungle` in die Default ACL mitgegeben hatten, hat `setfacl` automatisch alle anderen Einträge aus der Access ACL kopiert, um so eine gültige Default ACL zu bilden. Default ACLs haben keinen direkten Einfluss auf die Zugriffsberechtigungen und wirken sich nur beim Erzeugen von Dateisystemobjekten aus. Beim Vererben wird nur die Default ACL des übergeordneten Verzeichnisses beachtet.

2. Legen Sie im nächsten Beispiel mit `mkdir` ein Unterverzeichnis in `mydir` an, welches die Default ACL „erben“ wird.

```
mkdir mydir/mysubdir
getfacl mydir/mysubdir

# file: mydir/mysubdir
# owner: tux
# group: projekt3
user::rwx
group::r-x
group:djungle:r-x
mask::r-x
other::---
default:user::rwx
default:group::r-x
default:group:djungle:r-x
default:mask::r-x
default:other::---
```

Wie erwartet hat das neu angelegte Unterverzeichnis `mysubdir` die Rechte aus der Default ACL des übergeordneten Verzeichnisses. Die Access ACL von `mysubdir` ist ein exaktes Abbild der Default ACL von `mydir`, ebenso die Default ACL, die dieses Verzeichnis wiederum an seine Unterobjekte weitervererben wird.

3. Legen Sie im `mydir` Verzeichnis mit `touch` eine Datei an:

```
touch mydir/myfile
ls -l mydir/myfile

-rw-r-----+ ... tux projekt3 ... mydir/myfile

getfacl mydir/myfile

# file: mydir/myfile
# owner: tux
# group: projekt3
user::rw-
group::r-x          # effective:r--
group:djungle:r-x   # effective:r--
mask::r--
other::---
```

Wichtig an diesem Beispiel: `touch` übergibt `mode` mit dem Wert von `0666`, das bedeutet, dass neue Dateien mit Lese- und Schreibrechten für alle Benutzerklassen angelegt werden, so nicht entweder per `umask` oder Default ACL andere Beschränkungen existieren (siehe Abschnitt *Auswirkungen einer Default ACL* auf Seite 565).

Am konkreten Beispiel heißt dies, dass alle Zugriffsrechte, die nicht im `mode` Wert enthalten sind, aus den entsprechenden ACL-Einträgen entfernt werden. Aus dem ACL-Eintrag der *group class* wurden keine Berechtigungen entfernt, allerdings wurde der *mask* Eintrag dahingehend angepasst, dass nicht per `mode` gesetzte Berechtigungsbits maskiert werden.

Auf diese Weise ist sichergestellt, dass zum Beispiel Compiler reibungslos mit ACLs interagieren können. Sie können Dateien mit beschränkten Zugriffsrechten anlegen und diese anschließend als ausführbar markieren. Über den `mask` Mechanismus ist gewährleistet, dass die richtigen Benutzer und Gruppen schließlich die Rechte erhalten, die ihnen in der Default ACL zugestanden werden.

Auswertung einer ACL

Nachdem Sie den Umgang mit den wichtigsten Tools zur ACL-Konfiguration bereits verstanden haben, werden Sie im Folgenden kurz an den Auswertungsalgorithmus herangeführt, den jeder Prozess oder jede Anwendung durchlaufen muss, bevor ihm Zugriff auf ein ACL-geschütztes Dateisystemobjekt gewährt werden kann.

Grundsätzlich werden die ACL-Einträge in folgender Reihenfolge untersucht: *owner*, *named user*, *owning group* oder *named group* und *other*. Über den Eintrag, der am besten auf den Prozess passt, wird schließlich der Zugang geregelt.

Komplizierter werden die Verhältnisse, wenn ein Prozess zu mehr als einer Gruppe gehört, also potentiell auch mehrere *group* Einträge passen könnten. Aus den passenden Einträgen mit den erforderlichen Rechten wird ein beliebiger ausgesucht. Für das Endresultat „Zugriff gewährt" ist es natürlich unerheblich, welcher dieser Einträge den Ausschlag gegeben hat. Enthält keiner der passenden *group* Einträge die korrekten Rechten, gibt wiederum ein beliebiger von ihnen den Ausschlag für das Endresultat „Zugriff verweigert".

Ausblick

Wie in den vorangehenden Abschnitten beschrieben, können mit ACLs sehr anspruchsvolle Rechteszenarien umgesetzt werden, die modernen Anwendungen gerecht werden. Das traditionelle Rechtekonzept und ACLs lassen sich geschickt miteinander vereinbaren.

Allerdings fehlt einigen wichtigen Anwendungen noch die Unterstützung für ACLs. Insbesondere auf dem Gebiet der Backup-Anwendungen gibt es mit Ausnahme des star Archivierers keine Programme, die den vollen Erhalt der ACLs sicherstellen.

Die grundlegenden Dateikommandos (`cp`, `mv`, `ls`, ...) unterstützen ACLs. Viele Editoren und Dateimanager (z.B. Konqueror) beinhalten jedoch keine ACL-Unterstützung. Beim Kopieren von Dateien mit Konqueror gehen zur Zeit noch die ACLs verloren. Wenn eine Datei mit einer Access ACL im Editor bearbeitet wird, hängt es vom Backup-Modus des verwendeten Editors ab, ob die Access ACL nach Abschluss der Bearbeitung weiterhin vorliegt:

- Schreibt der Editor die Änderungen in die Originaldatei, bleibt die Access ACL erhalten.
- Legt der Editor eine neue Datei an, die nach Abschluss der Änderungen in die alte umbenannt wird, gehen die ACLs möglicherweise verloren, es sein denn, der Editor unterstützt ACLs.

Je breiter die Unterstützung von ACLs durch Anwendungen, desto umfassender kann dieses Feature ausgenutzt werden.

Tipp

Weitere Informationen

Detailinformationen zu ACLs finden Sie unter den folgenden URLs

`http://sdb.suse.de/de/sdb/html/81_acl.html`

`http://acl.bestbits.at/`

und auf der Manual-Page von `getfacl` (`man 1 getfacl`), der Manual-Page von `acl` (`man 5 acl`) und der Manual-Page von `setfacl` (`man 1 setfacl`).

Tipp