Wiki / Konfiguration

Dieser Artikel wurde für die folgenden Ubuntu-Versionen getestet:

- **Ubuntu 12.04** Precise Pangolin

Zum Verständnis dieses Artikels sind folgende Seiten hilfreich:

1. **Ein Terminal öffnen**
2. **Umgang mit dem Terminal-Editor "Vim(vi)"**

Inhaltsverzeichnis

Mit der Konfiguration zum Befehl **sudo** [http://wiki.ubuntuusers.de/sudo] bestimmt der System-Administrator, welche Benutzer (oder Benutzergruppen) welche Befehle wann an welchem Ort als Benutzer root (oder einem anderen Benutzer) ausführen dürfen.

Einstellungen

Um einem Benutzer den Befehl **sudo** zu ermöglichen, reicht es aus, ihn in die Benutzergruppe sudo aufzunehmen. Die Einstellungen des **sudo**-Befehls und damit verbundener Rechte werden in der Datei **/etc/sudoers** eingestellt.

Achtung!

Die Datei **/etc/sudoers** sollte immer mit dem Befehl visudo bearbeitet werden, da so eine Syntaxüberprüfung gewährleistet ist. Bei der direkten Bearbeitung ohne Prüfung kann der kleinste Tippfehler dazu führen, dass man sich aus dem System aussperrt und nur über den **Recovery Modus** wieder Zugang erhalten kann.

Ebenfalls ist es möglich, eine 2. Konsole vor der Änderung mit Root-Rechten zu öffnen (Konsole auf machen und sudo -i eingeben). Stellt man dann fest, dass man sich aus dem System ausgesperrt hat, kann man über die 2. Konsole die Änderungen wieder rückgängig machen.

Die letzte Zeile der Sudoers-Datei muss zudem immer leer sein!

Es ist wichtig die **sudoers.tmp** zu speichern, denn visudo überprüft nur dann die Syntax und man hat die Möglichkeit Fehler zu korrigieren.

Man öffnet also ein Terminal[1] und gibt dort

```
sudo visudo
```

ein. Oder

```
EDITOR=vim sudo -E visudo
```

um statt des Editors nano den Editor vim zu benutzen.

Man kann aber stattdessen auch generell den Standardeditor global in Ubuntu ändern (siehe hier: **Standard-Editor-festlegen** [http://wiki.ubuntuusers.de/Editor#Standard-Editor-festlegen]).

Ein Beispieleintrag für **/etc/sudoers** sieht so aus:

```
root            ALL = (ALL) ALL
```

Das heißt root darf alle Befehle mit **sudo** ausführen.

```
%administrator          ALL = (ALL) ALL
```

Die Gruppe administrator darf alle Befehle mittels **sudo** als root ausführen.

```
admin       ALL = NOPASSWD: ALL
%users      ALL = NOPASSWD: /usr/sbin/IRGENDEINSCRIPT
```

Der Benutzer admin darf ohne Passwortabfrage alle Programme ausführen. Die Gruppe users darf **ohne** Passwortabfrage den Befehl **/usr/sbin/IRGENDEINSCRIPT** ausführen.

Hinweis:

Die untersten Einträge haben die höchste Priorität. D.h., gilt für die Gruppe admin: "ALL = (ALL) ALL", so werden NOPASSWD-Einträge für die Benutzer aus admin überschrieben. Es ist also ratsam, "%admin ALL=(ALL) ALL" über entsprechenden NOPASSWD-Einträgen zu definieren.

Achtung!

Man sollte die Passwortabfrage nur für Scripte oder Programme deaktivieren, die in einem Systemverzeichnis (/bin, /sbin, /usr/bin, /usr/sbin, ...) liegen und root gehören. Grund: Ist die Passwortabfrage beispielsweise für das Script ~/bin/mein-script deaktiviert, dann kann es ein Angreifer mit Benutzerrechten einfach löschen und durch ein beliebiges Script oder Programm ersetzen und dann mit root-Rechten ausführen. Auf diese Weise wäre das gesamte System kompromittiert. Erst wenn das Script root gehört und nur von root geändert werden kann und auch in einem Verzeichnis liegt, in dem nur root Schreibrechte hat, ist dieser Angriff nicht mehr möglich.

Alias

Es gibt 4 verschieden Aliastypen:

- User_Alias
- Runas_Alias
- Host_Alias
- Cmnd_Alias

Einen Alias definiert man so:

```
Alias_Type NAME = item1, item2, ...
```

Der *NAME* ist ein String mit Großbuchstaben und (optional) _ .

Beispielhaft werden hier nur die User-Aliase und die Befehls-Aliase beschrieben, weil sie am wichtigsten sind.

User-Alias

Indem man Aliase definiert, kann man bestimmten Usern (ohne sie in einer Gruppe zusammenzulegen) gezielt Superuserrechte vergeben. Beispiel:

```
User_Alias     FULLTIMERS = millert, mikef, dowdy
User_Alias     PARTTIMERS = bostley, jwfox, crawl
User_Alias     WEBMASTERS = will, wendy, wim
```

In diesem Beispiel wird ein User-Alias namens *FULLTIMERS* definiert mit 3 Mitgliedern: *millert, mikef, dowdy*.

Die Rechte werden dann so vergeben:

```
FULLTIMERS        ALL = (ALL) ALL
```

Befehls-Alias

Befehls-Aliase werden genauso erstellt, sie haben entsprechend ein führendes Cmnd_Alias. In diesem Beispiel wird eine Gruppe mit Namen DOWN erstellt, die mehrere Befehle zum runterfahren, neu starten, usw. enthält:

```
Cmnd_Alias DOWN = /sbin/shutdown, /sbin/reboot, /usr/sbin/pm-suspend, /usr/sbin
/pm-hibernate
```

Werden zusätzlich Parameter für den Befehl benötigt, müssen die Zeichen ",", "\", ":" und "=" mit einem Backslash "\" escaped werden, da die Syntaxprüfung sonst Fehler meldet und die Parameter nicht an den Befehl durchreicht. Das folgende Beispiel erstellt eine Gruppe namens HELLIGKEIT, die nur einen langen Befehl inkl. Parametern enthält:

```
Cmnd_Alias HELLIGKEIT = /usr/bin/setpci -s 00\:02.0 f4.b\=ff
```

Zugewiesen werden diese Befehls-Aliase, in diesem Beispiel der oben definierten Gruppe der PARTTIMERS, so:

```
PARTTIMERS ALL = NOPASSWD: DOWN
```

Nun können die User der Gruppe PARTTIMERS die Befehle der Gruppe DOWN ohne Passwortabfrage ausführen.

Adminstrator auf Zeit

Der voreingestellte Zeitraum von 15 Minuten, in dem das Passwort bei **sudo**-Aktionen nicht abgefragt wird, kann verändert werden; beispielsweise auf den Wert 0. Dazu fügt man diese Zeile hinzu oder hängt timestamp_timeout = 0 , mit Komma getrennt vom bereits existerenden Parameter, an die bestehende Defaults-Zeile dran:

```
# Timeout auf Null setzen. Standardwert ist 15 Minuten.
Defaults timestamp_timeout = 0
```

Danach muss man bei jedem sudo-Aufruf das Passwort eingeben.

Visuelles Feedback bei Passworteingabe

Normalerweise gibt **sudo** nichts aus, wenn das Passwort eingegeben wird, was es einem Angreifer schwer machen soll, die Länge des Passwortes zu bestimmen. Das hat aber beispielsweise den Nachteil, dass unerfahrende Benutzer denken, dass die Passworteingabe nicht funktioniert oder, dass man das Passwort in einem anderen Fenster eintippt, weil dieses Fenster gerade den Focus bekommen hat. Durch Einfügen der Zeile

```
Defaults pwfeedback
```

wird für jedes eingegebene Zeichen ein Stern (*) ausgegeben und beim Drücken von  werden alle Sterne

wieder gelöscht.

Man kann außerdem auf folgende Weise ein externes Programm zum Einlesen des Passwortes benutzen:

```
Defaults askpass = /pfad/zum/externen/programm
```

Dieses Programm kann sich dann um das Feedback bei der Passworteingabe kümmern, und es sind auch grafische Programme möglich (beispielsweise **ssh-askpass**). Um das externe Programm aber auch zu benutzen, muss **sudo** mit der Option **-A** aufgerufen werden. Das externe Programm kann außerdem in der Variable **SUDO_ASKPASS** spezifiziert werden, was den Eintrag in der Datei **/etc/sudoers** überstimmt.

Für mehr Informationen sei auf die **Manpage** [http://wiki.ubuntuusers.de/man] von **sudo** und **sudoers** verwiesen.

Root-Passwort einrichten

Im Allgemeinen fährt man bei Ubuntu gut damit, den Root-Account deaktiviert zu lassen und das System ausschließlich über sudo zu administrieren. Es gibt allerdings einen Grund, den Root-Account u.U. zu aktivieren. Wenn nicht-vertrauenswürdige Benutzer direkten Zugang zum Rechner haben, können sie ihn über den Bootmanager Grub im **Recovery Modus** [http://wiki.ubuntuusers.de/Recovery_Modus] starten. Ist der Root-Account deaktiviert, erhält ein böswilliger Benutzer so ohne Passwortabfrage eine Root-Shell. Wird der Rechner in einer Multiuser-Umgebung eingesetzt, z.B. im Rechnerraum einer Schule oder Universität, sollte der Root-Account daher aktiviert werden. Allerdings gibt es in so einem Szenario noch eine Reihe weiterer möglicher Schlupflöcher, die relativ einfach auszunutzen sind. Die Beachtung des Artikels **Lokale Sicherheit** [http://wiki.ubuntuusers.de/Lokale_Sicherheit] ist daher das mindeste, was man unternehmen sollte.

Was man aber nach Möglichkeit unterlassen sollte, ist nur noch auf den Root-Account zu setzen und die sudo-Möglichkeit ganz zu deaktivieren. Die Konfigurationswerkzeuge von Ubuntu sind nämlich auf die Benutzung von sudo ausgelegt, und wenn diese Art der Authentifizierung nicht mehr zur Verfügung steht, kann dies durchaus zu Problemen insbesondere bei den grafischen Administrationswerkzeugen kommen.

Wer also den direkten Login als Root aus irgendeinem Grund aktivieren will, muss den Root-Benutzerzugang mit einem gültigen Passwort versehen. Bitte bei folgendem Befehl beachten, dass **sudo** zuerst nach dem eigenen Passwort fragt.

```
sudo passwd
```

Damit kann man sich mit **su** bereits als Benutzer root einloggen, allerdings ist es unter Verwendung von **sudo** nach wie vor möglich, Root-Rechte zu erlangen. Damit für alle administrativen Tätigkeiten das Root-Passwort anstelle des Benutzerpasswortes benötigt wird (auch für Sudo-Frontends), empfiehlt es sich, einen entsprechenden Eintrag in **/etc/sudoers** vorzunehmen. Dazu startet man den dazu vorgesehenen Editor **visudo** und fügt der ersten nicht auskommentierten Zeile die flags targetpw und timestamp_timeout = 0 hinzu.

```
Defaults        !lecture,tty_tickets,!fqdn,targetpw,timestamp_timeout = 0
```

Das timestamp_timeout = 0 führt dazu, dass bei *jeder* Benutzung von **sudo** nach dem Passwort gefragt wird. Was für einen Sinn so eine Maßnahme haben soll, ist allerdings fraglich. Schließlich kann man jeden Benutzer, der

kein sudo benutzen soll, auch einfach aus der sudo-Gruppe entfernen.

Root-Anmeldung untersagen

Hat man einmal dem root-Benutzer ein Passwort gegeben und möchte dies wieder rückgängig machen, so kann man mit dem Befehl

```
sudo passwd -d root
```

den Account wieder in den "deaktivierten" Zustand bringen.

Achtung!

Mit dem früher und auch heute noch oft im Netz zu findenden

```
sudo passwd -l root
```

ist danach kein Zugang zum root-Recovery-Modus mehr möglich! Deshalb wird von Benutzung der Option -l zum Deaktivieren des root-Passwortes abgeraten!

sudo-Eingaben umleiten

Soll die Ausgabe eines per sudo ausgeführten Befehls **umgeleitet** [http://wiki.ubuntuusers.de/Shell/Umleitungen] werden, so erfolgt dies im allgemeinen **nicht** mit Root-Rechten. Vor allem, wenn man mit dem Befehl **echo** etwas in eine Datei schreiben oder anhängen will (auch wenn es ein **Editor** [http://wiki.ubuntuusers.de/Editor] genauso tun würde), funktioniert dies nicht:

```
sudo echo mem > /sys/power/state            # Zugriff verweigert
```

Abhilfe schafft das Kapseln des Befehls in eine eigene Shell (bash):

```
sudo bash -c "echo mem > /sys/power/state"
```

oder einfach die Umsetzung mit **tee**:

```
echo mem | sudo tee /sys/power/state
```

Experten-Info:

Alternativ lässt sich auch in eine Root-Shell wechseln, indem sudo -i oder sudo /bin/bash ausgeführt wird.

Links

Für weitere Informationen siehe auch: **Manpage zu sudoers** [http://manpages.ubuntu.com/manpages/maverick/en/man5/sudoers.5.html]

- **Projektseite** [http://www.sudo.ws/sudo/sudoers.man.html] sudoers-Manual
- **Fix Broken Sudo** [http://www.psychocats.net/ubuntu/fixsudo]

Diese Revision [http://wiki.ubuntuusers.de/sudo/Konfiguration?rev=713422] wurde am 22. April 2014 21:19 von **noisefloor** erstellt.
Die folgenden Schlagworte wurden dem Artikel zugewiesen: **Shell** [http://wiki.ubuntuusers.de/Wiki/Tags?tag=Shell], **Server** [http://wiki.ubuntuusers.de/Wiki/Tags?tag=Server], **sudoers** [http://wiki.ubuntuusers.de/Wiki/Tags?tag=sudoers]