

Reiko Kaps

Daten-Sprinter

NFSv4 unter Linux

Die aktuellen Linux-Distributionen haben das Network File System Version 4 an Bord und verteilen damit flott ganze Dateisysteme.

Die Neuerungen von NFSv4 machen Lust auf einen ersten Blick, zumal die meisten Linuxe alle nötigen Pakete für das verteilte Dateisystem schon mitbringen.

Das eigene Netz benötigt für die Experimente einige Voraussetzungen. In der einfachsten Einrichtung authentifiziert NFSv4 Clients über ihre IP-Adresse, die der Server gegen das Domain Name System (DNS) überprüft. Die meisten Netze besitzen dank des Heim-Routers zwar einen lokalen DNS-Server. Doch verwaltet dieser meist nicht die Rechner des lokale Netzes, sodass man entweder einen zweiten Server aufsetzen oder die Adress-Namens-Zuordnung in der Datei /etc/hosts (auf allen LAN-Rechnern) eintragen muss.

Bei nur einem NFS-Server und einem Client reichen dort die beiden Zeilen

```
192.0.2.10    nfs-server.example.net
192.0.2.11    nfs-client.example.net
```

Alle Adressangaben entstammen dem Bereich 192.0.2.0/24, der für die Dokumentation reserviert ist. Auch die Domainnamen folgen den Empfehlungen aus RFC 2606 und müssen für eigene Experimente angepasst werden.

Netzwerkfreigaben, klassisch

Der NFSv4-Server steckt in den meisten Linux-Distributionen im Paket nfs-kernel-server, das der Paketmanager nachlädt und installiert. OpenSuse benötigt zusätzlich den Wert NFS4_SUPPORT="yes" in der Datei /etc/sysconfig/nfs, der NFSv4 nach einem Neustart des NFS-Servers einschaltet.

Anschließend überprüft man, ob alle nötigen Dienste laufen.

NFS-Freigaben bindet OpenSuse über ein Yast-Interface ein, das mittels eines Browsers die Server im Netz finden kann.

Der Aufruf `ps ax | egrep 'rpc|nfs'` sollte eine Ausgabe liefern, die dem folgenden Beispiel ähnelt:

```
2036 ?    S<    0:00 [nfsiod]
2521 ?    S<    0:00 [nfsd4]
2522 ?    S     0:00 [nfsd]
...
1663 ?    S<    0:00 [rpciod/0]
2029 ?    Ss    0:00 /sbin/rpc.statd
2045 ?    Ss    0:00 ↗
           /usr/sbin/rpc.idmapd
2533 ?    Ss    0:00 ↗
           /usr/sbin/rpc.mountd
```

Die Prozesse `nfsd`, `rpc.statd` und `rpc.mountd` gehören zu NFSv3, `nfsd4` und `rpc.idmapd` zu NFSv4.

Anders als seine Vorgänger hängt NFSv4 die Freigaben in ein Basisverzeichnis ein. Der Befehl `mkdir /srv/nfsv4` legt es an und die Zeile

```
/srv/nfsv4 192.0.2.0/24(rw,fsid=0, ↗
           insecure,no_subtree_check,async)
```

in der Datei /etc/exports teilt es dem NFS-Server mit. Das NFSv4-Basisverzeichnis benötigt zwingend die Option `fsid=0`, die es als Wurzelverzeichnis markiert und über das ein Client alle Freigaben des Servers in einem Rutsch einhängen kann. Diese Freigabe authentifiziert Clients wie die Vorgängerversion über die IP-Adresse oder den Hostnamen – mit der Angabe `192.168.1.0/24` können alle Rechner aus einem privaten

Netz zugreifen. Die weiteren Optionen zwischen den Klammern sind optional, sie erlauben NFS-Zugriffe von Ports überhalb 1024 (`insecure`) und Schreibzugriffe (`rw`). Der Parameter `async` schaltet asynchrones Schreiben ein und `no_subtree_check` eine Sicherheitsfunktion ab, die überprüft, ob sich eine vom Client angesprochene Datei im freigegebenen Verzeichnisbaum befindet.

Das Basisverzeichnis nimmt weitere Freigaben als normale Verzeichnisse auf. Der Befehl `mount -bind Quellverzeichnis Freigabeverzeichnis` verbindet anschließend ein Datenverzeichnis mit der Freigabe unterhalb von /srv/nfsv4:

```
mkdir /srv/nfsv4/daten
mount --bind /home /srv/nfsv4/daten
```

Für eine dauerhafte Freigabe benötigt die Datei /etc/exports den Eintrag

```
/srv/nfsv4/daten 192.0.2.0/24(rw,nohide, ↗
           insecure,async,no_subtree_check)
```

Der zusätzliche Parameter `nohide` weist den Server an, die Freigabe nicht zu verstecken. Mit der Zeile

```
/home /srv/nfsv4/daten none bind 0 0
```

in /etc/fstab verknüpft Linux die Freigabe mit dem Datenverzeichnis gleich beim Start. Änderungen an den Freigaben in /etc/exports teilt der Befehl `exportfs -ra` dem NFS-Server mit. Temporäre NFS-Freigaben benötigen keinen Eintrag in /etc/exports, sie lassen sich mittels `exportfs` anlegen:

```
exportfs -o rw,nohide,insecure ↗
           192.0.2.0/24:/srv/nfsv4/daten
```

Unter OpenSuse 11 steht für die Grundeinrichtung des NFSv4-Servers ein Yast-Frontend bereit. Will man allerdings Freigaben in das Basisverzeichnis einbinden,

erleichtert das GUI die Arbeit nicht: Die Hilfe gibt sich kryptisch und schlägt vor, die Option `bind=/target/path` einzufügen. Wo diese hingehört, offenbart Yast hingegen nicht. Die oben beschriebene Methode per Kommandozeile führt aber auch dort zum Ziel.

Für das Auflösen von Benutzernamen benötigt NFSv4 sowohl auf dem Client als auch auf dem Server den Dienst `idmapd`, der per Vorgabe Benutzer- und Gruppennamen via /etc/passwd respektive /etc/group in ihre Kennungen umsetzt. `idmapd` liest seine Einstellungen aus der Datei /etc/idmapd.conf. Dort setzt man den Wert hinter `Domain =` auf den Wert der eigenen, lokalen Domain und startet Client- und Server-Software neu.

Kann `idmapd` die Benutzer- und Gruppenkennung nicht zuordnen, bildet es sie auf einen Benutzer mit minimalen Rechten ab (`nobody`, `nogroup`). Die Zuordnung lässt sich unter [Mapping] anpassen.

In dieser Einrichtung kümmert sich `idmapd` nicht um die korrekte Übersetzung zwischen gleichen Benutzernamen, die aber unterschiedliche User-IDs besitzen: Ein Client-Benutzer namens `gast` mit der User-ID 5001 kann auf dem Server nicht schreiben, wenn der dortige Gast-Account eine abweichende ID besitzt.

Der Befehl `mount -t nfs4 nfs-server:/Einhängepunkt` verknüpft die Server-Freigaben mit dem Dateisystem des Clients.

NFS getunnelt

In der Kombination mit SSH lässt sich NFSv4 recht leicht verschlüsseln und über Netzwerk-grenzen tunneln. Auf dem Server benötigt dies eine Freigabe für localhost respektive für die IP-Adresse 127.0.0.1.

Der Client verbindet sich mittels des Kommandos `ssh -L 8888:localhost:2049 nfs-server.example.net` zum NFS-Server. Der Parameter `-L` tunnelt über die SSH-Verbindung den NFS-Port 2049 auf dem Server zum Port 8888 auf dem Client-Rechner, der die Freigaben durch

```
mount -t nfs4 -o port=8888,proto=tcp
localhost:/remote /mnt/remote
```

in sein eigenes Dateisystem einhängt. (rek)

