Сетевая файловая система (NFS)

Евгений Мисник

Сетевая файловая система (NFS) позволяет обращаться к файлам на компьютерах в сети, как если бы они располагались локально.

NFS применяется для разделения каталогов или файлов между несколькими пользователями одной сети.

Вот некоторые из наиболее заметных преимуществ, которые даёт использование NFS:

- Отдельно взятые рабочие станции используют меньше собственного дискового пространства, так как совместно используемые данные могут храниться на одной отдельной машине и быть доступными для других машин в сети.
- Пользователям не нужно иметь домашние каталоги, отдельные для каждой машины в вашей сети. Домашние каталоги могут располагаться на сервере NFS и их можно сделать доступными отовсюду в сети.
- Устройства хранения информации, такие, как дискеты, приводы CD-ROM и устройства Zip(R), могут использоваться другими машинами в сети. Это может привести к уменьшению переносимых устройств хранения информации в сети.

Содержание

1 Как работает NFS. **3**

2 Процедуры RPC NFSv2 и NFSv3. **4**

3 Аутентификация и авторизация **6**

4 Протокол монтирования **6**

5 Сервер монтирования **7**

6 Протокол блокировки **10**

7 Сервер блокировки **12**

8 Протокол мониторинга состояния **12**

9 Сервер ограничения использования ресурсов **12**

10 Сервер преобразования идентификаторов **13**

11 Сервер NFS **13**

12 Клиент монтирования **18**

13 Настройка производительности **20**
13.1 Portmapper (для NFSv2 и NFSv3). 21
13.2 Блокировка файлов. 22

14 Подключение файловых систем NFS. **22**
14.1 Подключение файловых систем NFS с помощью /etc/fstab. 22
14.2 Подключение файловых систем NFS с помощью autofs. 22

15 Экспорт файловых систем NFS. **23**

16 Запуск и остановка сервера. **24**

17 NFS и firewall. **24**

18 NFSv4 **25**
18.1 Настройка сервера NFSv4 26
18.2 Настройка клиента NFSv4 26
18.3 Конфигурация idmapd (общая для сервера и клиента) 26
18.4 NFSv4 и Kerberos 27

19 NFSv4.1 **27**
19.1 Параллельное расширение NFSv4.1 27

1 Как работает NFS.

NFS строится по крайней мере из двух основных частей: сервера и одного или большего количества клиентов. Клиент обращается к данным, находящимся на сервере, в режиме удалённого доступа. Для того, чтобы это нормально функционировало, нужно настроить и запустить несколько процессов.

На сервере работают следующие демоны:

Демон	Описание
nfsd	Демон NFS, обслуживающий запросы от клиентов NFS
mountd	Демон монтирования NFS, который выполняет запросы, передаваемые ему от nfsd
rpcbind	Этот демон позволяет клиентам NFS определить порт, используемый сервером NFS

Таблица 1:

NFSv2 и NFSv3 сильно привязаны к семантике файловых операций в Unix (файлы ".", "..", права, uid/gid, имена в символьных ссылках и т.д.). Предполагается, что файловая система имеет иерархическую структуру, каждый файл или каталог имеет имя. Разделитель имён не важен, т.к. все процедуры NFS имеют дело только с простыми именами. Максимальная длина в NFSv2 для простого имени файла - 255 байт, составного - 1024 байта. Протокол NFSv3 не накладывает ограничения на длины простого и составного имён, но клиент и сервер могут иметь их. Максимальный размер блока данных - 8192 байта в NFSv2, определяется процедурой FSINFO в NFSv3. В NFSv3 добавлены типы файлов: блочное устройство, символьное устройство, сокет, канал. Максимальный размер файла увеличен в NFSv3 с 2^31-1 до 2^64-1 байта. В NFSv3 добавлены средства слабой синхронизации (т.е. отсутствует гарантия) кеша клиента (weak cache consistency).

Построен над RPC, ранние версии используют UDP, поздние - TCP, обычно используется порт 2049 (т.е. rpcbind использовать необязательно). При работе в режиме TCP позволяет установить постоянное соединение для конкретной файловой системы (одно для всех клиентских процессов), а опция TCP keepalive позволяет определить сбой в работе сервера или перезагрузку и автоматически осуществлять повторное соединение и повтор "зависших" запросов.

Сервер NFS версий 2 и 3 не хранит информацию о состоянии клиентов и их предыдущих запросах (stateless), это упрощает для клиента обработку сбоев сервера (перезагрузка сервера воспринимается как простая задержка). Процедуры open и close отсутствуют. Все операции в NFSv2 синхронные, в NFSv3 предусмотрено кеширование записи (асинхронный режим WRITE). Клиент может кешировать запись самостоятельно. Большинство клиентов NFSv3 кешируют атрибуты файлов. Невозможно реализовать удаление открытого файла так, чтобы его можно было продолжать читать до закрытия (нет команд open и close), но некоторые клиенты имитируют это свойство создавая временные файлы вида ".nfs...". Аналогичная проблема с динамическим изменением прав доступа к файлу (из-за этого в реализациях считается, что владелец файла всегда имеет доступ к нему). Кстати, права доступа к файлу необходимо проверять при каждом обращении, т.к. сервер не способен определить, что файл уже был открыт. Сервер не способен отличить запрос на чтение файла от запроса на выполнение, поэтому разрешает чтение, если у пользователя есть права на чтение или исполнение.

Для работы NFS, кроме собственно NFS (RPC программа 100003) требуется программа

удалённого монтирования (RPC программа 100005), блокировки файлов (RPC программа 100021), проверки статуса (RPC программа 100024), квотирования (RPC программа 100011). При работе с файлами клиент использует указатели на файл (file handles, фиксированная длина 32 байта в NFS2, переменная длина до 64 байт в NFS3), т.е. при открытии файла сервер возвращает указатель, которым клиент пользуется в дальнейшем при работе с файлом. Содержимое указателя не должно интерпретироваться клиентом, но в Unix реализации там хранится major/minor устройства, на котором смонтирована файловая система, inode и версия inode (требуется при повторном использовании одного и того же inode для различных файлов). Если указатели совпадают, то они указывают на один и тот же файл. Если они различны, то это не означает, что файлы различны. Указатель может "зачервстветь" (stale), если между обращениями клиента к серверу он, например, перезагрузился и major/minor блочного устройства изменились.

2 Процедуры RPC NFSv2 и NFSv3.

Процедуры NFS (в NFS3 каждый вызов, изменяющий атрибуты файла, возвращает их новое значение):

- STATFS (получить информацию о файловой системе: размер блоков для чтения и записи, размер блоков, число блоков файловой системы, число свободных блоков, число доступных для непривилегированного пользователя блоков; удалена в NFSv3);
- FSINFO (получить статическую информацию о файловой системе: максимальный и оптимальный размер блока для чтения и записи, максимальный размер файла, аккуратность учёта времени в файловой системе и возможность установки времён, наличие поддержки жёстких и символьных ссылок, различные значения атрибутов POSIX.1 для различных файлов; нет в NFSv2);
- FSSTAT (получить динамическую информацию о файловой системе: размер файловой системы, остаток свободного места, количество файлов, остаток свободных inode-ов, намёк на частоту изменений; нет в NFSv2);
- READDIR (выдать список файлов в каталоге по его указателю; задаётся максимально-ожидаемое число байт; возвращается список имен файлов, номеров (?) файлов и ссылка на следующую страницу (cookie); cookie позволяет продолжить получение списка файлов с прерванного места; в NFSv2 ссылка прилагалась к каждому файлу; в NFSv3 только одна ссылка на запрос с дополнительным верификатором для проверки);
- READDIRPLUS (выдать список файлов с их атрибутами и указателями; нет в NFS2);
- CREATE (создать файл; в NFSv2 используется также для создания специальных файлов; в NFSv3 появился флажок режима создания: без проверки файла с таким же именем, с проверкой, эксклюзивного создания - используется идентификатор клиента, сетевой адрес и идентификатор запроса, которые сервер запоминает в постоянной памяти);
- MKDIR (создать каталог);
- REMOVE (удалить файл);
- RMDIR (удалить каталог; сервер может отказать в удалении непустого каталога);

- RENAME (переименовать файл или каталог в пределах одной файловой системы; может повлечь удаление существующего файла с таким же именем);
- LINK (создать жёсткую ссылку на файл в пределах одной файловой системы);
- SYMLINK (создать символьную ссылку; сервер не интерпретирует текст ссылки; можно также задать атрибуты ссылки, хотя в Unix они всегда '0777');
- READLINK (читать имя файла по символьной ссылке (ASCII); формат определяется сервером и клиент д.б. к этому готов);
- MKNOD (создание специальных файлов - блочные и символьные устройства, сокеты, каналы; нет в NFSv2);
- GETATTR (получить атрибуты файла по его указателю: тип, mode, размер (4 байта в NFSv2 и 8 байт в NFSv3), размер используемого места (в NFSv3), nlink, uid, gid, размер блока (удалён в NFSv3), rdev, идентификатор файловой системы, inode, atime, mtime, ctime);
- PATHCONF (получить POSIX.1 атрибуты файла по его указателю: максимальное количество жёстких ссылок, максимальная длина простого имени, слишком длинные имена молчаливо обрезаются, ограничения на смену владельца и группы, прописные и строчные буквы в именах файлов различаются и сохраняются; нет в NFSv2 - реализовно в протоколе монтирования);
- SETATTR (установить атрибуты файла по его указателю: mode, размер (если указать 0, то файл будет урезан), uid, gid, atime, mtime; в NFSv3 введена возможность предварительной проверки - по ctime - изменяемого файла);
- ACCESS (проверить права доступа; нет в NFSv2; клиент указывает список действий, права на которые он хочет проверить; это только предварительная проверка, но помогает решить проблему с преобразованием идентификаторов пользователей между клиентом и сервером или с авторизацией, отличной от AUTH_UNIX);
- LOOKUP (возвращает указатель на файл по его имени и указателю на каталог; один уровень иерархии за обращение; клиент ответственен за монтирование промежуточных точек, иначе он увидит содержимое подлежащего каталога, и за обработку символьных ссылок);
- READ (клиент передаёт указатель на файл, смещение от начала файла (32 бита для NFSv2 и 64 бита для NFSv3), количество байт - для NFSv2 до 8192 байт, для NFSv3 максимальное и оптимальное значение возвращается FSINFO; в NFSv3 появился флажок конца файла; возвращается количество считанных байт и сами данные);
- WRITE (клиент передаёт указатель на файл, смещение от начала файла, количество байт, содержимое; количество байт - для NFSv2 до 8192 байт, для NFSv3 максимальное и оптимальное значение возвращается FSINFO; в NFSv2 запись только синхронная, в NFSv3 появился флажок требования синхронности записи данных и метаданных, а также подсказки для слабой синхронизации кеширования; в NFSv3 возвращается количество реально записанных байт);
- COMMIT (сбросить накопленный при асинхронной записи кеш для указанного файла на диск с проверкой перезагрузки сервера в промежутке; нет в NFSv2);

- WRITECACHE (не используется в NFSv2; удалена в NFSv3);
- ROOT (получить указатель корня - не используется в NFSv2, удалена в NFSv3).

3 Аутентификация и авторизация

Для авторизации запросов при использовании RPC аутентификации AUTH_UNIX используются uid и gid передаваемые клиентом, т.е. сервер полностью доверяет клиенту, а единственной возможностью проверить клиента является его IP адрес и таблицы преобразования идентификаторов (обычно производится на сервере по статической таблице или таблица создаётся в момент монтирования).

Обычно uid=0 клиента преобразуется в UID_NOBODY (nfsnobody, 65534, -2) на сервере, кроме случаев использования NFS в качестве корневой файловой системы.

Предполагается возможность использовать аутентификацию AUTH_DES или AUTH_KERB.

4 Протокол монтирования

Прежде чем обратиться к файлу на удалённой файловой системе клиент (ОС клиента) должен смонтировать её и получить от сервера указатель на неё с помощью явной команды mount или с помощью одного из расплодившихся автоматических монтировщиков (amd, autofs, automount, supermount, superpupermount). Для этого клиентская программа монтирования (mount, amd) с помощью rpcbind определяет номер порта, на котором на сервере слушает программа RPC mount нужной версии (1, 2, 3), обращается к ней за указателем на корень требуемой файловой системы, далее этот указатель хранится на клиентской машине до размонтирования и используется при открытии файлов на удалённой файловой системе. Протокол RPC mount 1 используется совместно с NFSv2, аутентификация AUTH_UNIX или AUTH_NONE (RFC 1094). Состояния клиентов хранятся. Может использовать как UDP, так и TCP. Сервер проверяет права клиента, основываясь на его IP адресе и "привилегированности" исходящего порта. Совместно с NFSv3 используется RPC mount 3 (RFC 1813), максимальный размер простого имени файла - 255, составного - 1024.

Процедуры протокола RPC mount 1 (RFC 1094):

- NULL
- MNT (смонтировать; принимает имя каталога; возвращает указатель на корень файловой системы, который можно использовать в NFvS; сервер пополняет таблицу смонтированных файловых систем с указанием клиента: IP, имя хоста?);
- UMNT (размонтировать; принимает имя каталога);
- UMNTALL (размонтировать все файловые системы, смонтированные для этого клиента; а кто клиент-то: IP, имя хоста?);
- EXPORT (выдать список всех экспортируемых файловых систем и имён групп, которые смогут импортировать данную файловую систему (?));
- DUMP (выдать список смонтированных файловых систем с указанием хостов-клиентов).

Процедуры протокола RPC mount 3 (RFC 1813):

- NULL

- MNT (смонтировать; принимает имя каталога (ASCII) и список допустимых методов авторизации; возвращает указатель на корень файловой системы, который можно использовать в NFvS; сервер пополняет таблицу смонтированных файловых систем с указанием имени хоста клиента);
- DUMP (выдать список смонтированных файловых систем с указанием имён хостов клиентов в виде пар: имя каталога, имя хоста; клиент останется в списке, если не выдаст явно UMNT/UMNTALL);
- UMNT (размонтировать; принимает имя каталога; иногда список неразмонтированных файловых систем используется для извещения клиентов о предстоящем выключении сервера);
- UMNTALL (размонтировать все файловые системы, смонтированные для этого клиента; а кто клиент-то: IP, имя хоста?);
- EXPORT (выдать список всех экспортируемых файловых систем и имена, которые смогут импортировать данную файловую систему; имена могут представлять имена хостов или групп хостов, не могут быть интерпретированы клиентами).

5 Сервер монтирования

Список экспортируемых каталогов задаётся в файле `/etc/exports` в виде отдельных строк для каждого каталога с указанием допустимых клиентов и опций монтирования для них (строки комментариев начинаются с '#', после редактирования файла необходимо перезагрузить (reload) сервис nfs): имя-каталога описание-клиента(опции,через,запятую) [описание-клиента(опции,через,запятую) ...]

Описание клиента:

- пробел (любой клиент);
- имя хоста или IP адрес;
- шаблон (типа glob, но точка не включается в '*') имени хоста;
- IP-адрес-сети/маска-сети или IP-адрес-сети/длина-маски-сети;
- @имя-NIS-группы;
- gss/krb5 (требуется аутентификация RPCSEC_GSS).

Опции (sync, ro, root_squash, no_all_squash, wdelay, secure, hide, subtree_check, secure_locks, nocrossmnt)[1]:

- insecure (по умолчанию - secure; разрешать запросы с портов клиента с номерами более 1024);
- rw;
- sync | async;
- no_wdelay;

[1] В зависимости от операционной системы опции могут отличаться.

- hide | nohide (по умолчанию - hide: если сервер экспортирует 2 файловые системы, одна из которых смонтирована внутри второй, то смонтировавший только вторую систему клиент увидит пустой каталог в точке монтирования первой файловой системы, для того чтобы добраться до первой файловой системы клиенту необходимо смонтировать её явно куда-то ещё; nohide для "внутренней" файловой системы позволяет клиенту "прозрачно" переходить от одной файловой системы к другой; можно задавать только для одиночных хостов, могут быть проблемы - совпадающие inode);
- crossmnt (аналогично hide, но устанавливается для "внешней" файловой системы);
- mountpoint[=путь] (aka mp=; экспортировать только, если точка экспорта является точкой монтирования);
- noaccess (экспортируемый каталог и ниже невидим для клиента);
- no_subtree_check (при экспортировании только части файловой системы сервер должен проверить запрос на нахождение требуемого файла внутри экспортируемой части и отсутствие запрета доступа несуперпользователей к каталогу; для этой проверки сервер должен добавить информацию о расположении файла в указатель файла, что приводит к проблемам при переименовании файлов; отключение проверки уменьшает безопасность, но увеличивает надёжность работы);
- fsid=число (по умолчанию - major (12 бит, до ядра 2.6 - 8 бит) и minor (20 бит, до ядра 2.6 - 8 бит) блочного устройства; позволяет вручную задавать идентификатор файловой системы; обязательно, если экспортируемый каталог лежит в файловой системе не на блочном устройстве, при использовании кластеров и если имеется шанс смены major/minor при перезагрузке сервера; в протоколе NFSv3 под fsid зарезервировано 64 бита (uint64), но find показывая (-printf "%F %D %i %p\n") в формате st_dev (__u32) выводит что-то странное (аналогично "mountpoint -d путь") (несколько файловых систем могут иметь одинаковый fsid на стороне клиента - необходимо использовать inode верхнего каталога); требуется уникальность (в пределах сервера?), т.к. клиент получается совсем другой fsid; для NFSv4 fsid=0 обозначает корень экспортируемых систем);
- root_squash (по умолчанию; отключение - no_root_squash) - запросы от uid=0 интерпретируются как будто они приходят от анонимного uid (см. anonuid) и анонимного gid ()см. anongid;
- all_squash - отображать все uid и gid в анонимные;
- squash_uids=список (отображать uid из списка в nobody);
- squash_gids=список (отображать gid из списка в nobody);
- anonuid=число (использовать число в качестве анонимного uid; по умолчанию, nfsnobody (65534));
- anongid=число (использовать число в качестве анонимного gid; по умолчанию, nfsnobody (65534));
- map_daemon (использовать ugidd(8) на клиентской стороне для отображения uid и gid между клиентом и сервером - протокол RPC UGID; по умолчанию, преобразование не производится - map_identity);

- map_static=имя-файла (отображение uid/gid между клиентом и сервером производится с помощью указанного файла; каждая строка состоит из 3 полей (комментарии начинаются с '#'): "uid" или "gid", uid/gid на стороне клиента, uid/gid на стороне сервера);
- map_nis=домен-NIS (отображение uid/gid между клиентом и сервером производится с помощью NIS сервера);
- mapping=identity (?);
- no_auth_nlm (aka insecure_locks; не требовать аутентификации для запросов блокировки; многие клиенты не умеют аутентифицировать такие запросы, при этом блокировка работает только для открытых на запись всем файлов);
- no_acl (по умолчании - acl; клиенты NFSv2 и старые клиенты NFSv3 проверяют права доступа самостоятельно; новые клиенты NFSv3 делают запрос ACCESS к серверу; no_acl позволяет не выдавать клиенту ACL файлов для файловых систем с поддержкой ACL);
- link_absolute (не трогать символьные ссылки на абсолютные адреса);
- link_relative (символьные ссылки на абсолютные адреса преобразуются в относительные).

Утилита `exportfs` управляет списком экспортируемых с сервера файловых систем. Список хранится в файле `/var/lib/nfs/etab`, список смонтированных клиентами файловых систем хранится в `/var/lib/nfs/xtab`, используется `mountd` при запросе на монтирование с удалённого компьютера.

Параметры exportfs:

- -v;
- -r (синхронизовать /etc/exports и /var/lib/nfs/xtab и ядро 2.4);
- [клиент:имя-каталога] (добавить или удалить указанную файловую систему для указанного клиента);
- -u (удаление из списка экспортируемых вместо добавления);
- -a (добавить или удалить все файловые системы);
- -o опции,через,запятую (перечень опций аналогичен `/etc/exports`; позволяет изменять опции уже смонтированных файловых систем);
- -i (не использовать /etc/exports при добавлении, только запросы из командной строки);
- -f (сбросить список экспортируемых систем в ядре 2.6; для активных клиентов добавятся новые записи при следующем обращении).

Утилита showmount запрашивает mountd на удалённом хосте о смонтированных файловых системах. По умолчанию выдаётся отсортированный список клиентов.

Ключи:

- --all (выдаётся список клиентов и точек монтирования с указанием для кого);

- --directories (выдаётся список точек монтирования с указанием для кого);
- --exports (выдаётся список экспортируемых файловых систем с точки зрения nfsd).

Сервер rpc.mountd запускается из сервиса /etc/init.d/nfs:

- --debug {all | auth | call | general | parse};
- --foreground;
- --exports-file имя-файла (/etc/exports);
- --descriptors максимальное-число-дескрипторов-файлов;
- --no-nfs-version номер-версии;
- --nfs-version номер-версии;
- --no-tcp;
- --port номер-порта (явное задание номера прослушиваемого порта вместо случайного от portmapper).

Параметры можно задавать в `/etc/sysconfig/nfs`:

- MOUNTD_NFS_V2={no|default};
- MOUNTD_NFS_V3={no|default};
- MOUNTD_PORT;
- RPCMOUNTDOPTS.

Монтирование в NFSv2 и NFSv3 осуществляется от имени хоста клиента, а не от конкретного пользователя, что требует изрядного доверия к администратору данного хоста. Причём хост определяется по имени (IP адресу). NFSv4 аутентифицирует конечного пользователя (RPCSEC_GSS, Kerberos 5). Имеется поддержка tcp_wrapper (имя программы - mountd).

6 Протокол блокировки

Протокол блокировки файлов Network Lock Manager (NLM, RFC 1813, X/OpenNFS) версии 3 (для NFSv2) и 4 (для NFSv3) обеспечивает возможность блокировки файлов и записей. Клиент NFS должен быть готов к его отсутствию на сервере. После запуска сервера некоторое время (grace period, 45 секунд) принимаются только восстановления блокировок, новые блокировки не принимаются. Блокировки делятся на отслеживаемые и неотслеживаемые. Отслеживаемые блокировки должны восстанавливаться после перезапуска сервера без участия клиента и сбрасываться при сбое клиента. Для реализации отслеживаемых блокировок требуется наличие серверов мониторинга состояния (NSM) на сервере и клиенте. Перед запросом блокировки клиент NLM выдаёт запрос к своему NSM на мониторинг сервера. Сервер перед выполнением запроса на блокировку выдаёт запрос к своему NSM на мониторинг клиента. По снятию блокировки или отказу от неё мониторинг клиента снимается. При перезагрузке сервера его NSM извещает все клиентские NSM об изменении состояния. Те, в свою очередь, извещают заинтересованные локальные NLM клиенты, которые восстанавливают блокировки во время "льготного" периода. При перезагрузке клиента его

NSM извещает все серверные NSM об изменении состояния. Те, в свою очередь, извещают заинтересованный локальный NLM сервер, который снимает все блокировки от данного клиента.

Неотслеживаемые блокировки (SHARE, UNSHARE, NM_LOCK, FREE_ALL) предназначены для совместимости с однозадачными ОС типа DOS. Клиент должен самостоятельно заботиться о восстановлении блокировок после перезапуска сервера и освобождать старые блокировки при своей перезагрузке.

В основном, NLM4 отличается от NLM3 увеличение длины и смещения в файле с 32 до 64 бит.

NLM4 может использовать как UDP, так и TCP протокол (RPC 100021, версия 4).

Процедуры:

- NULL;
- TEST (проверить возможность установить блокировку);
- LOCK (установить блокировку; режимы немедленного ответа и постановки в очередь; флажок восстановления блокировок в период после перезапуска сервера; пр постановке в очередь клиент передаёт серверу процедуру GRANTED);
- CANCEL (отменить задержанную в очереди блокировку);
- UNLOCK (снять блокировку);
- GRANTED (вызывается сервером после успешного завершения блокировки, предоставляется клиентом при вызове LOCK);
- SHARE (для совместимости с DOS, предполагается однозадачность клиентского хоста);
- UNSHARE (для совместимости с DOS);
- NM_LOCK (для совместимости с DOS);
- FREE_ALL (для совместимости с DOS, освободить все блокировки).

Асинхронные запросы:

- TEST_MSG;
- LOCK_MSG;
- CANCEL_MSG;
- UNLOCK_MSG;
- GRANTED_MSG.

Асинхронные ответы:

- TEST_RES;
- LOCK_RES;
- CANCEL_RES;
- UNLOCK_RES;
- GRANTED_RES.

7 Сервер блокировки

Сервер блокировки rpc.lockd (/etc/init.d/nfslock, /var/lock/subsys/nfslock, параметры из /etc/sysconfig/nfs). Запускается в пространстве пользователя только для старых ядер (до 2.2.18). Для новых ядер запускается как модуль ядра (lockd).

Параметры в /etc/sysconfig/nfs:

- LOCKD_TCPPORT
- LOCKD_UDPPORT

Параметры модуля ядра:

- nlm_tcpport
- nlm_udpport

Или задать те же параметры с помощью sysctl:
/sbin/sysctl -w fs.nfs.nlm_tcpport=номер-порта
/sbin/sysctl -w fs.nfs.nlm_udpport=номер-порта

8 Протокол мониторинга состояния

Сервер мониторинга состояния rpc.statd. Запускается из сервиса nfslock (/etc/init.d/nfslock). Список отслеживаемых хостов хранится в /var/lib/nfs/statd/sm/.

Параметры:

- -F (не уходить в фоновый режим)
- -d (отладочная печать на stderr)
- --name имя-локального-хоста
- --outgoing-port порт
- --port порт
- --state-directory-path имя-каталога (где хранить информацию; /var/lib/nfs)

Параметры могут задаваться в /etc/sysconfig/nfs:

- STATD_PORT
- STATD_OUTGOING_PORT

Имеется поддержка tcp_wrapper (имя программы - statd).

9 Сервер ограничения использования ресурсов

Сервер ограничения использования ресурсов rpc.rquotad. Запускается из /etc/init.d/nfs.

Параметры можно задавать в /etc/sysconfig/nfs:

- RQUOTAD_PORT
- RPCRQUOTADOPTS

Параметры:

- --no-setquota (по умолчанию; не разрешать установку ограничений)
- --setquota (разрешать)
- --foreground
- --autofs (отслеживать точки монтирования autofs)
- --port номер-порта

Имеется поддержка tcp_wrapper (имя программы - rquotad).

10 Сервер преобразования идентификаторов

Сервер rpc.idmapd (сервис rpcidmapd) для NFSv4 на сервере преобразует локальные uid/gid пользователей в формат вида имя@домен, а сервис rpcidmapd на клиенте преобразует имена пользователей/групп вида имя@домен в локальные идентификаторы пользователя и группы (/etc/idmapd.conf, idmapd.conf(5)):

```
[General]
Domain = имя-домена # должно совпадать на клиенте и сервере
[Mapping]
Nobody-User = nobody
Nobody-Group = nobody
[Translation]
Method = nsswitch # umich_ldap и static
```

Запуск:

```
[mkdir /var/lib/nfs/rpc_pipefs]
[mount -t rpc_pipefs sunrpc /var/lib/nfs/rpc_pipefs]
service rpcidmapd start
```

Все незнакомые клиенту uid и gid будут преобразованы в nobody.

11 Сервер NFS

Перед запуском NFS сервера необходимо убедиться в наличии сервиса portmap. Основная функциональность в модуле ядра nfsd.

Сервис nfs запускает серверы rpcsvcgssd, rpc.nfsd, rpc.mountd и rpc.rquotad (скрипт - /etc/init.d/nfs, параметры - /etc/sysconfig/nfs). Сервис nfslock запускает серверы rpc.lockd и rpc.statd (скрипт - /etc/init.d/nfslock). Сервис nfs-rdma (/etc/rdma/rdma.conf) обеспечивает NFS через RDMA (загружает модули svcrdma и xprtrdma).

Параметры для rpc.nfsd в /etc/sysconfig/nfs:

- RPCNFSDCOUNT (количество обслуживающих процессов; 8)
- RPCNFSDARGS

Ключи rpc.nfsd:

- --port номер-порта (2049)
- --no-nfs-version номер-версии
- --no-tcp --no-udp число-потоков

Запуск сервера NFS, монтирования, блокировки, квотирования и статуса с "правильными" портами (для сетевого экрана)

- желательно предварительно размонтировать все ресурсы на клиентах;
- остановить и запретить запуск rpcidmapd, если не планируется использование NFSv4:

```
chkconfig --level 345 rpcidmapd off service rpcidmapd stop
```

- если нужно, то разрешить запуск сервисов portmap, nfs и nfslock:

```
chkconfig --levels 345 portmap/rpcbind on
chkconfig --levels 345 nfs on
chkconfig --levels 345 nfslock on
```

- если нужно, то остановить сервисы nfslock и nfs, запустить portmap/rpcbind, выгрузить модули

```
service nfslock stop
service nfs stop
umount /proc/fs/nfsd # /var/lib/nfs/rpc_pipefs
rmmod nfsd # nfs
service rpcidmapd stop
service autofs stop # где-то потом его надо запустить, если он реально
нужен
rmmod nfs
rmmod nfs_acl
rmmod lockd
service portmap restart # service rpcbind restart
```

- открыть порты в iptables

для	RPC: UDP/111, TCP/111
для	NFS: UDP/2049, TCP/2049
для	rpc.statd: UDP/4000, TCP/4000
для	lockd: UDP/4001, TCP/4001
для	mountd: UDP/4002, TCP/4002
для	rpc.rquota: UDP/4003, TCP/4003

- для сервера rpc.nfsd добавить в /etc/sysconfig/nfs строку

  RPCNFSDARGS="--port 2049"

- для сервера монтирования добавить в /etc/sysconfig/nfs строку

  MOUNTD_PORT=4002

- для настройки rpc.rquota для новых версий необходимо добавить в /etc/sysconfig/nfs строку

  RQUOTAD_PORT=4003

- для настройки rpc.rquota необходимо для старых версий (тем не менее, надо иметь пакет quota 3.08 или свежее) добавить в /etc/services:

  rquotad 4003/tcp
  rquotad 4003/udp

- проверить адекватность /etc/exports

- запустить сервисы rpc.nfsd, mountd и rpc.rquota (заодно запускаются rpcsvcgssd и rpc.idmapd, если не забыли их удалить)

  service nfsd start

или в новых версиях

  service nfs start

- для сервера блокировки для новых систем добавить в /etc/sysconfig/nfs строки:

  LOCKD_TCPPORT=4001
  LOCKD_UDPPORT=4001

- для сервера блокировки для старых систем добавить непосредственно в /etc/modprobe[.conf]:

options lockd nlm_udpport=4001 nlm_tcpport=4001

- привязать сервер статуса rpc.statd к порту 4000 (для старых систем в /etc/init.d/nfslock запускать rpc.statd с ключом -p 4000)

  STATD_PORT=4000

- запустить сервисы lockd и rpc.statd

  service nfslock start

- убедиться, что все порты привязались номально с помощью "lsof -i -n -P" и "netstat -a -n" (часть портов используется модулями ядра, которые lsof не видит)

- если перед "перестройкой" сервером пользовались клиенты и их не удалось размонтировать, то придётся перезапустить на клиентах сервисы автоматического монтирования (am-utils, autofs)

Версии серверов проще всего посмотреть с помощью команды "rpcinfo -p".
nfsstat позволяет посмотреть статистику RPC и NFS:

- --all

- --server

- --client

- --nfs (без RPC)
- --rpc (без NFS)
- -2 (ограничиться версией 2)
- -3 (ограничиться версией 3)
- -4 (ограничиться версией 4)
- -o {nfs | rpc | net | fh | rc} (fh - статистика кеша указателей файлов)
- -z (сброс статистики)

/proc/fs/nfsd - специальная файловая система (ядро 2.6) для управления NFS сервером (nfsd(7)). Монтировать командой

```
mount -t nfsd nfsd /proc/fs/nfsd
```

Файлы в /proc/fs/nfsd

- exports содержит список экспортируемых файловых систем и клиентов, которым их экспортировали, а также параметры;
- threads содержит число потоков (также можно изменять);
- с помощью filehandle можно получить указатель на файл.

Каталог /proc/net/rpc содержит "сырую" статистику, которую можно получить с помощью nfsstat, а также различные кеши.

Каталог /proc/sys/sunrpc содержит "ручки управления" отладочной печатью (сюда надо записывать битовые маски классов отладочной печати: RPCDBG_, NFSDBG_, NFSDDBG_, NLMDBG_ в include/linux/sunrpc/debug.h, include/linux/nfs_fs.h, include/linux/nfsd/debug.h и include/linux/lockd/debug.h; вывод идёт в syslog в раздел kernel):

- nfs_debug (клиент)
- nfsd_debug (сервер)
- nlm_debug (сервер блокировки)
- rpc_debug

rpc_debug (44):

- RPCDBG_XPRT - 0x0001 (работа с сетью)
- RPCDBG_CALL 0x0002 (трассировка вызовов подпрограмм)
- RPCDBG_DEBUG 0x0004
- RPCDBG_NFS - 0x0008
- RPCDBG_AUTH - 0x0010 (обработка авторизации)
- RPCDBG_PMAP - 0x0020
- RPCDBG_SCHED - 0x0040 (внутренняя кухня обработки запроса: очереди, задержки и т.д.)

- RPCDBG_SVCSOCK - 0x0100 (приходящие запросы)
- RPCDBG_SVCDSP - 0x0200 (?)
- RPCDBG_MISC - 0x0400 (?)

nfs_debug (241):

- NFSDBG_VFS - 0x0001 (работа с указателями файлов)
- NFSDBG_DIRCACHE - 0x0002
- NFSDBG_LOOKUPCACHE - 0x0004
- NFSDBG_PAGECACHE - 0x0008
- NFSDBG_PROC - 0x0010 (вызов RPC процедур)
- NFSDBG_XDR - 0x0020
- NFSDBG_FILE - 0x0040 (операции чтения/записи)
- NFSDBG_ROOT - 0x0080

nfsd_debug (759):

- NFSDDBG_SOCK - 0x0001 (?)
- NFSDDBG_FH - 0x0002 (операции с указателями файлов)
- NFSDDBG_EXPORT - 0x0004
- NFSDDBG_SVC - 0x0008 (номер версии, номер процедуры)
- NFSDDBG_PROC - 0x0010
- NFSDDBG_FILEOP - 0x0020
- NFSDDBG_AUTH - 0x0040
- NFSDDBG_REPCACHE - 0x0080
- NFSDDBG_XDR - 0x0100 (ressize_check: проверка допустимости параметров)
- NFSDDBG_LOCKD - 0x0200
- NFSDDBG_ALL - 0x7FFF
- NFSDDBG_NOCHANGE - 0xFFFF

nlm_debug (511):

- NLMDBG_SVC 0x0001
- NLMDBG_CLIENT 0x0002
- NLMDBG_CLNTLOCK 0x0004
- NLMDBG_SVCLOCK 0x0008

- NLMDBG_MONITOR 0x0010
- NLMDBG_CLNTSUBS 0x0020
- NLMDBG_SVCSUBS 0x0040
- NLMDBG_HOSTCACHE 0x0080
- NLMDBG_XDR 0x0100

12 Клиент монтирования

Чтобы смонтировать файловую систему NFS командой mount необходимо указать тип файловой системы (-t nfs[4]), вместо блочного устройства указать имя-хоста:каталог и указать точку монтирования. Размонтировать можно с помощью обычной команды umount (она не извещает сервер). Если необходимо размонтировать не дожидаясь завершения процессов, то можно использовать ключ "-f" в Solaris (процессы получат сообщения об ошибках ввода-вывода) или ключ "-l" в Linux (реальное размонтирование произойдёт по завершению процесса). Опции монтирования можно указать в /etc/fstab, в командной строке после ключа -o или в настройках автомонтировщика:

- стандартные: default (rw,suid,dev,exec,auto,nouser,async)
- async, sync, dirsync,
- [no]atime, [no]diratime, relatime
- [no]auto
- [no]dev
- [no]exec (запретить запуск выполняемых файлов с этой файловой системы)
- [no]suid (не интерпретировать suid бит)
- [no]user
- [no]users
- [no]group
- [no]owner
- remount
- ro
- rw
- [no]mand
- _netdev
- {fs|def}context

- rsize=размер-блока-чтения (максимальный размер определён в nfsd/const.h: 1024 или 8192 для NFSv2; 32768 для NFSv3/UDP; 32768 (RHEL5) или 1MB (F10) для NFSv3/TCP; перед увеличением задумайтесь о значениях /proc/sys/net/core/?mem_max и /proc/sys/net/core/?mem_d
- wsize=размер-блока-записи
- timeo=время (для UDP; 7 секунд; сколько времени ожидать ответа от сервера до первого повторения запроса; при очередном повторе время ожидания удваивается до достижения 60 секунд)
- retrans=раз (для UDP; 3; число повторных запросов в "малом" цикле)
- acregmin=секунд (3; минимальное время кеширования атрибутов файлов)
- acregmax=секунд (60; максимальное время кеширования атрибутов файлов
- acdirmin=секунд (30; минимальное время кеширования атрибутов каталогов)
- acdirmax=секунд (60; максимальное время кеширования атрибутов каталогов)
- actimeo=секунд (установить все ac* в одинаковое значение)
- port=номер-порта-NFS-сервера (0; если 0, то запрашивать portmapper)
- nfsprog=номер-RPC-программы-NFS (100003)
- nfsvers=номер-версии-NFS (2; не существует для nfs4)
- vers=номер-версии-NFS (2; не существует для nfs4; в FC3)
- mountport=номер-порта-mountd
- mounthost=имя-хоста-mountd
- mountprog=номер-RPC-программы-mountd (100005)
- mountvers=номер-версии-mountd (1)
- sec=метод-аутентификации (sys; sys - обычный AUTH_SYSTEM; krb5 - Kerberos V5; krb5i - Kerberos V5 с защитой от изменения сообщений; krb5p - Kerberos V5 с шифрованием и защитой от изменения сообщений)
- namlen=длина (255; максимальная длина имени файла; требуется для серверов с RPC mount v1)
- [no]bg (nobg; если первая попытка монтирования исчерпала время ожидания, то продолжать попытки в фоновом режиме)
- retry=минут (10000; сколько времени пытаться смонтировать файловую систему)
- [no]fg (fg; если первая попытка монтирования исчерпала время ожидания, то продолжать)
- [no]soft (hard; прекращать запросы к серверу по завершении цикла повторных запросов и выдать ошибку ввода/вывода; не рекомендуется при использовании UDP)

- [no]hard (hard; повторять запросы к серверу до победного конца; в конце цикла выдавать сообщение об ошибке; для возможности прерывания доступа к файлу необходима опция intr)
- [no]intr (nointr; позволять прерывать сигналом INTR запросы к серверу в режиме hard) [no]posix (полностью поддерживать семантику POSIX, требует RPC mount v2 на сервере)
- [no]cto (не извлекать новые атрибуты при создании файла)
- [no]ac (ac; кешировать или не кешировать атрибуты файлов; без кеширования работать нереально)
- [no]tcp (udp; использовать протокол TCP)
- [no]udp (udp; использовать протокол UDP)
- proto=udp|tcp (udp; только для nfs4)
- [no]lock (lock; не использовать блокировку файлов)
- [no]acl (отключить обработку ACL, обработка ACL в новых клиентах (NT ACL) несовместима со старыми серверами (POSIX), в FC5)
- lookupcache={all|positive|none} (кешировать позитивные (как в ядре до 2.6.28) или все результаты поиска в каталоге)
- local_lock={none|flock|posix} (какие механизмы блокировки файлов обрабатывать без обращения к серверу)
- [no]rdirplus (rdirplus; использовать ли запросы READDIRPLUS)
- [no]sharecache (sharecache после 2.6.18; общий кеш для всех монтирований одной точки экспорта)

13 Настройка производительности

Рекомендации для максимальной потоковой скорости (nfs-tuning-secrets-130-lca2008-d7.odp):

- настройка аппаратуры дисковой подсистемы сервера; размер полосы (stripe) RAID должен вмещаться в буфер NFS; разделение пропускной способности шины; DMA
- настройка драйвера дисковой подсистемы сервера (Command Tag Queue для SCSI)
- настройка блочного уровня дисковой подсистемы сервера (в т.ч. md, lvm с его снимками), планировщик (/sys/block/*/queue/scheduler), /sys/block/*/queue/max_sectors_kb, /sys/block/*/queue/read_ahead_kb
- настройка файловой системы сервера (data=writeback, журнал на отдельный диск); bonnie настройка VFS и VM сервера (vm.dirty_writeback_centisecs, vm.dirty_background_ratio)
- настройка NFS сервера (число потоков) и экспорта (async), no_subtree_check
- настройка сетевой подсистемы сервера (netperf)

  - использовать максимально возможное MTU
  - net.ipv4.tcp_{,r,w}mem='8192 262144 524288'

- настройка сетевого драйвера сервера (ethtool) и оборудования; разделение пропускной способности шины; DMA; разгрузка CPU от подсчёта контрольных сумм и "лишних" прерываний; привязка прерываний к конкретному процессору; респределение обработки прерываний по процессорам

- настройка сетевых коммутаторов настройка сетевого драйвера клиента (ethtool) и оборудования; разделение пропускной способности шины; DMA; разгрузка CPU от подсчёта контрольных сумм и "лишних" прерываний; привязка прерываний к конкретному процессору

- настройка сетевой подсистемы клиента (netperf)
  - использовать максимально возможное MTU
  - net.ipv4.tcp_{,r,w}mem='8192 262144 524288'

- монтирование на стороне клиента (/etc/auto.master, /etc/auto.net, /proc/mounts, "nfsstat -m")
  - NFSv3
  - TCP
  - максимально возможное значение rsize/wsize с обеих сторон (32768?), должен вмещать полосу (stripe) RAID
  - монтирование в режиме hard и intr (не soft)
  - async
  - не noac
  - параллелизм на сторон клиента (16, как?)

- настройка VFS клиента (число предварительно читаемых блоков для nfs д.б. кратно 4)

- настройка приложения

http://www.bog.pp.ru/work/NFS.html

13.1 Portmapper (для NFSv2 и NFSv3).

portmapper - это преобразователь номеров портов DARPA в вызовы соответствующих программ RPC. Для обеспечения безопасности надо редактировать файлы `/etc/hosts.allow` и `/etc/hosts.deny`.

В файл hosts.allow записываются адреса, для которых монтирование разрешено и в файл hosts.deny записывается ключ “ALL” для запрета всем остальным. Напр. `hosts.allow`:

```
portmap mountd nfsd statd lockd rquotad : 192.168.xxx.0/255.255.255.0
```

и hosts.deny:

```
portmap mountd nfsd statd lockd rquotad : ALL
```

13.2 Блокировка файлов.

Некоторым приложениям (например, mutt) для корректной работы необходима возможность блокировки файлов (file locking). При работе по NFS блокировка файлов может осуществляться при помощи демона lockd.

14 Подключение файловых систем NFS.

Используйте команду mount для подключения файловой системы NFS с другого компьютера:

```
mount shadowman:/mnt/export /mnt/local
```

Каталог - точка подключения на локальном компьютере (`/mnt/local` в приведённом выше примере) должен существовать.

В этой команде, `shadowman` - это имя файлового сервера NFS, `/mnt/export` - файловая система, экспортированная сервером shadowman, а `/mnt/local` - каталог на локальном компьютере, в который будет подключена файловая система. После выполнения команды `mount` (и если вам даны соответствующие разрешения на сервере `shadowman`) вы сможете ввести `ls /mnt/local` и получить список файлов в каталоге `/mnt/export` компьютера shadowman.

14.1 Подключение файловых систем NFS с помощью /etc/fstab.

Также можно подключить общий каталог NFS другого компьютера, добавив строку в файл `/etc/fstab`. В этой строке задаётся имя сервера NFS, экспортируемый каталог сервера и каталог локального компьютера, в который будет подключена файловая система. Вы должны быть пользователем root чтобы изменить файл /etc/fstab.

Типичная запись в файле /etc/fstab выглядит следующим образом:

```
server:/usr/local/pub /pub nfs rsize=8192,wsize=8192,timeo=14,intr
```

Точка подключения `/pub` должна существовать на вашем компьютере. Добавив эту строку в файл `/etc/fstab`, вы можете ввести в приглашении оболочки команду `mount /pub`, и каталог `/pub` будет подключен к серверу.

14.2 Подключение файловых систем NFS с помощью autofs.

Еще одним способом подключить разделяемый ресурс NFS можно с помощью autofs. Autofs использует демона automount для управления своими точками подключения, подключая их только по мере обращения.

Autofs анализирует главный файл соответствий `/etc/auto.master` для определения точек подключения. Затем запускается процесс автоматического подключения каждой точки с указанными параметрами. Каждая строка в главном файле соответствий определяет точку подключения, а отдельный файл связей определяет, какие файловые системы должны быть подключены в эту точку. Например, в файле `/etc/auto.mnt` определяются точки подключения в каталог `/mnt`; это соответствие определяется в файле `/etc/auto.master`.

Каждая запись в `auto.master` имеет три поля. В первом поле указывается точка подключения. Во втором поле определяется размещение файла связей, а третье поле является необязательным. Третье поле может содержать дополнительную информацию, например значение таймаута.

Например, чтобы подключить каталог /project52 удалённого компьютера penguin.host.net в точку подключения /mnt/myproject на вашем компьютере, добавьте в файл auto.master следующую строку:

```
/mnt /etc/auto.mnt --timeout 60
```

Добавьте следующую строку в файл /etc/auto.mnt:

```
myproject -rw,soft,intr,rsize=8192,wsize=8192 penguin.host.net:/project52
```

Первое поле в файле /etc/auto.mnt содержит имя каталога /mnt. Этот каталог динамически создаётся демоном automount. Он не должен существовать на клиентском компьютере. Второе поле содержит параметры подключения, например rw - определение доступа на чтение и запись. В третьем поле определяется размещение NFS-экспорта, включая имя компьютера и каталога.

Каталог /mnt должен существовать в локальной файловой системе. В каталоге /mnt локальной файловой системы не должно быть подкаталогов.

Autofs представляет собой службу. Чтобы запустить службу, введите в приглашении оболочки следующую команду:

```
service autofs restart
```

Чтобы просмотреть активные точки подключения, введите в приглашении оболочки следующую команду:

```
service autofs status
```

Если вы измените файл конфигурации /etc/auto.master во время работы autofs, вы должны указать демону automount перегрузить его, выполнив следующую команду в приглашении оболочки:

```
service autofs reload
```

15 Экспорт файловых систем NFS.

В файле /etc/exports определяются экспортируемые файловые системы. Он имеет следующий формат:

directory hostname(options)

Параметры (options) не являются обязательными. Например:

```
/mnt/export speedy.redhat.com
```

позволит пользователям компьютера speedy.redhat.com подключить /mnt/export со стандартным разрешением только-чтение, а:

```
/mnt/export speedy.redhat.com(rw)
```

позволит пользователям компьютера speedy.redhat.com подключить /mnt/export со правами на чтение и запись.

Будьте осторожны, используя пробелы в файле /etc/exports. Если между именем компьютера и параметрами в скобках пробелы отсутствует, эти параметры будут относиться только к компьютеру. Если между именем компьютера и параметрами присутствует пробел, эти параметры относятся ко всем остальным компьютерам. Например, изучите следующие строки:

```
/mnt/export speedy.redhat.com(rw)
/mnt/export speedy.redhat.com (rw)
```

В первой строке пользователи компьютера speedy.redhat.com получают доступ на чтение-запись, всем остальным доступ запрещён. Во второй строке пользователи компьютера speedy.redhat.com получают доступ только на чтение (по умолчанию), а все остальные получают доступ на чтение и запись.

После каждого изменения /etc/exports, укажите демонам NFS прочитать новую информацию или перезагрузить файл конфигурации:

```
/sbin/service nfs reload
```

16 Запуск и остановка сервера.

На сервере, экспортирующем файловые системы NFS, должна работать служба nfs.

Просмотрите состояние демона NFS, выполнив команду

```
/sbin/service nfs status
```

Запустите демона NFS, выполнив команду

```
/sbin/service nfs start
```

Остановите демона NFS, выполнив команду

```
/sbin/service nfs stop
```

Чтобы служба nfs запускалась при загрузке, выполните следующую команду:

```
/sbin/chkconfig --level 345 nfs on
```

17 NFS и firewall.

Для правильной работы с NFS-сервером на стороне сервера для lockd, mountd, rquotad и statd должны быть определены свободные фиксированные порты, напр.:

```
# vi /etc/sysconfig/nfs
# TCP port rpc.lockd should listen on.
LOCKD_TCPPORT=lockd-port-number
# UDP port rpc.lockd should listen on.
LOCKD_UDPPORT=lockd-port-number
# Port rpc.mountd should listen on.
MOUNTD_PORT=mountd-port-number
# Port rquotad should listen on.
RQUOTAD_PORT=rquotad-port-number
# Port rpc.statd should listen on.
STATD_PORT=statd-port-number
# Outgoing port statd should used. The default is port is random
STATD_OUTGOING_PORT=statd-outgoing-port-number
```

где

```
LOCKD_TCPPORT=32803
LOCKD_UDPPORT=32769
MOUNTD_PORT=892
RQUOTAD_PORT=875
STATD_PORT=662
STATD_OUTGOING_PORT=2020
```

Кроме того, надо создать правила iptables, напр. для сети 192.168.1.0/24 и вышеуказанных портов:

```
   -A RH-Firewall-1-INPUT -s 192.168.1.0/24 -m state --state NEW -p udp
--dport 111 -j ACCEPT
   -A RH-Firewall-1-INPUT -s 192.168.1.0/24 -m state --state NEW -p tcp
--dport 111 -j ACCEPT
   -A RH-Firewall-1-INPUT -s 192.168.1.0/24 -m state --state NEW -p tcp
--dport 2049 -j ACCEPT
   -A RH-Firewall-1-INPUT -s 192.168.1.0/24 -m state --state NEW -p tcp
--dport 32803 -j ACCEPT
   -A RH-Firewall-1-INPUT -s 192.168.1.0/24 -m state --state NEW -p udp
--dport 32769 -j ACCEPT
   -A RH-Firewall-1-INPUT -s 192.168.1.0/24 -m state --state NEW -p tcp
--dport 892 -j ACCEPT
   -A RH-Firewall-1-INPUT -s 192.168.1.0/24 -m state --state NEW -p udp
--dport 892 -j ACCEPT
   -A RH-Firewall-1-INPUT -s 192.168.1.0/24 -m state --state NEW -p tcp
--dport 875 -j ACCEPT
   -A RH-Firewall-1-INPUT -s 192.168.1.0/24 -m state --state NEW -p udp
--dport 875 -j ACCEPT
   -A RH-Firewall-1-INPUT -s 192.168.1.0/24 -m state --state NEW -p tcp
--dport 662 -j ACCEPT
   -A RH-Firewall-1-INPUT -s 192.168.1.0/24 -m state --state NEW -p udp
--dport 662 -j ACCEPT
```

18 NFSv4

NFSv4 (RFC 3010 - устарел, RFC 3530 - устарел, RFC 7530) не использует rpcbind, поддерживает NT ACL и сохраняет информацию о состоянии клиентов. Только режим TCP. Протоколы монтирования, блокировки и статуса встроены непосредственно в NFSv4. При монтировании аутентифицируется конечный пользователь, а не хост (причём по IP) как в NFSv3.

Описывается в RFC 7530 (RFC 3530). Не требуется portmap (хотя RPC используется), rpc.lockd и rpc.statd, rpc.mountd имеется, но он не обменивается данными с клиентом, необходим rpc.idmapd (т.е. соответствие имён пользователей на сервере и клиенте текстовое, а не по uid/gid), имеется поддержка ACL (Windows NT, а не POSIX!), сервер хранит информацию о клиентах (stateful), что позволяет клиентам безопасно кешировать данные и метаданные (используется механизм делегирования - delegations, leases; клиенты NFSv3 кешируют "на удачу" - в надежде, что в ближайшие секунды этот файл никому не будет нужен).

NFSv4.1 (RFC 5661, RFC 5662, RFC 5663, RFC 5664, RFC 6688) вводит расширения: pNFS (параллельное распределение доступа к файлам), сессии, многосерверное пространство имён, ACL, делегирование каталогов и извещения,

По умолчанию, сервис portmap запущен только на loopback (lo) интерфейсе в целях безопасности. Этого достаточно для раздачи сетевых ресурсов через nfs4.

18.1 Настройка сервера NFSv4

Файл /etc/exports должен выглядеть приблизительно так:

```
/export <client_ip|hostname|wildcard>(rw,fsid=0,sync,no_root_squash)
```

где

fsid=0 | root (zero or root): определяет, что данная файловая система является корнем для множественных экспортируемых директорий (только для NFSv4), напр.:

```
/NFS-SHARE 192.168.0.17(fsid=0,no_subtree_check,rw,root_squash,sync,anonuid=1000,anongid=
/NFS-SHARE/mydir 192.168.0.17(ro,sync,no_subtree_check)
```

/export - любая необходимая точка монтирования (export_path);
остальные опции по необходимости.

Необходимо настроить файл /etc/idmapd.conf, указав для “Domain” DNS доменное имя.
Загрузить “idmad”.
Загрузить “nfs” сервер.

18.2 Настройка клиента NFSv4

Необходимо настроить файл /etc/idmapd.conf, указав для “Domain” DNS доменное имя.
Загрузить “idmad”.
Произвести монтирование:

```
# mount -t nfs4 <servername>:/<export_path> </mount_path>
```

18.3 Конфигурация idmapd (общая для сервера и клиента)

Конфигурационный файл “idmapd.conf”:

```
[General]
Verbosity = 0
Pipefs-Directory = /var/lib/nfs/rpc_pipefs
Domain = mydomain.com
[Mapping]
Nobody-User = nobody
Nobody-Group = nobody
[Translation]
Method = nsswitch
```

10

18.4 NFSv4 и Kerberos

Требования:

- Key Distribution Center (KDC) должен быть установлен для сети;
- krb5, krb5-client должен быть инсталлирован и на NFS сервере, и на NFS клиенте;
- NFS сервер, клиент и сервер KDC должны иметь синхронизацию времени;

Файл /etc/exports при использовании Kerberos:

```
/export gss/krb5(rw,fsid=0,insecure,no_subtree_check,sync,no_root_squash)
/export gss/krb5i(rw,fsid=0,insecure,no_subtree_check,sync,no_root_squash)
```

Монтирование с Kerberos:

```
#mount -t <nfs4 or nfs> -osec=<secmode> nfsserver:/ /mntpoint
```

<secmode> может быть krb5(Autentication) или krb5i (Integrity).

19 NFSv4.1

Протокол NFSv4.1 является не расширением NFSv4, а самостоятельным протоколом. Описывается в RFC 5661-5667.

19.1 Параллельное расширение NFSv4.1

Сервер pNFS и клиент обмениваются только метаданными (разметка, layout, pNFS протокол). Разметка хранится в каталоге на сервере и может кешироваться клиентами. Сервер хранит информацию о клиентах, т.е. не является stateless, есть понятие "open" - команды LOOKUP+OPEN, LAYOUTGET, LAYOUTRETURN, LAYOUTCOMMIT. Сервер асинхронно извещает клиентов об изменениях разметки (CB_LAYOUTRECALL). Содержимое файлов хранится на отдельных серверах системы хранения данных. Протокол управления (не стандартизован) позволяет серверу управлять распределением файлов по серверам хранения. Протокол доступа к системе хранения данных позволяет клиентам осуществлять параллельный доступ к содержимому файлов. Возможны системы хранения различного типа:

- файловые (NFS; например, Network Appliance 8.1 C-mode) - в разметке передаётся, какая часть файла хранится на каком сервере (смещение, длина, идентификатор сервера хранения, NFS-дескриптор); в RHEL6.2 доступен клиент только этого типа;
- блочные (SBC; например, EMC Highroad, LSI) - используется протокол SCSI;
- объектные (OSD2; например, Panasas DirectFLOW) - вместо слова файл используется слово объект.

Головным разработчиком pNFS для Linux является CITI (University of Michigan Center for Information Technology Integration).

Сервер pNFS для Linux в виде прототипа поддерживает подлежащие файловые системы GFS2/OCFS2 (файловый тип хранения), EXOFS (объектный тип хранения), ext4/btrfs/xfs (блочный тип хранения).