dCache NFSv4.1

Tigran Mkrtchyan

Zeuthen, 13.04.12

Outline

- NFSv41 basics
  - NFSv4.1 concepts
  - PNFS
  - Id mapping
  - Industry standard
- dCache implementation

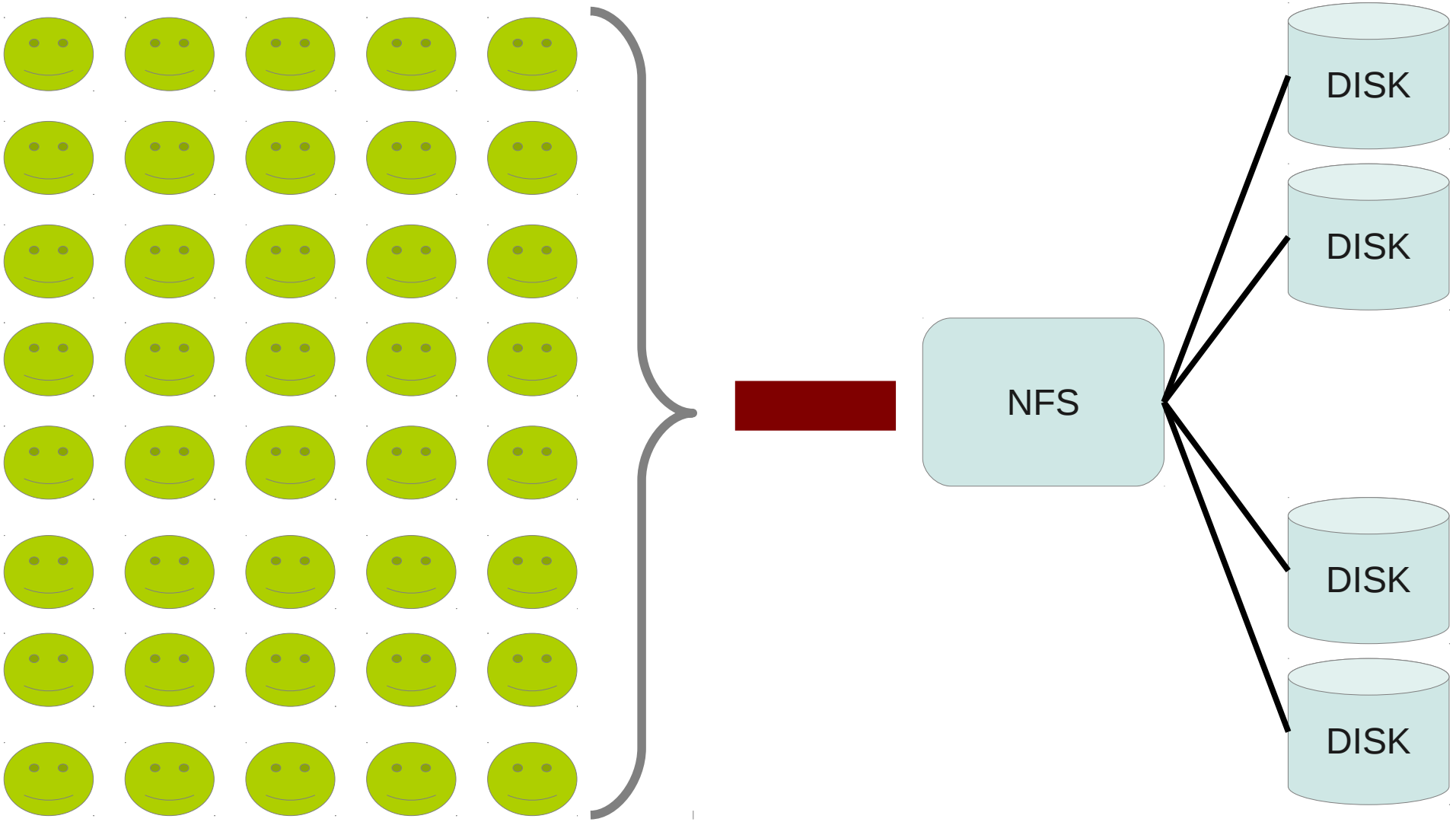
Classic NFS

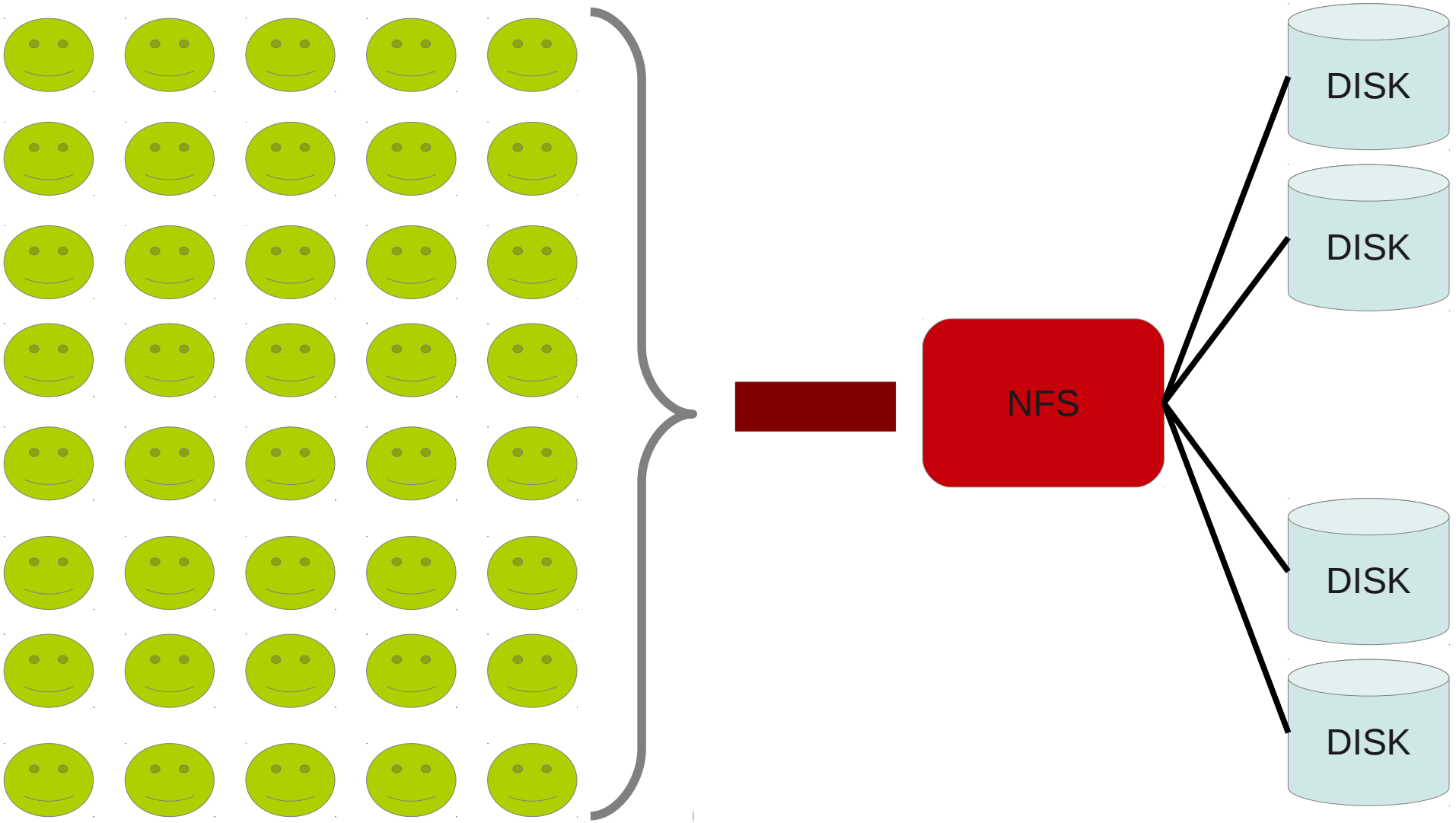
- BigData never fits into a single server.
- Big administrative overhead to keep data on multiple servers
- Single NFS server becomes bottleneck

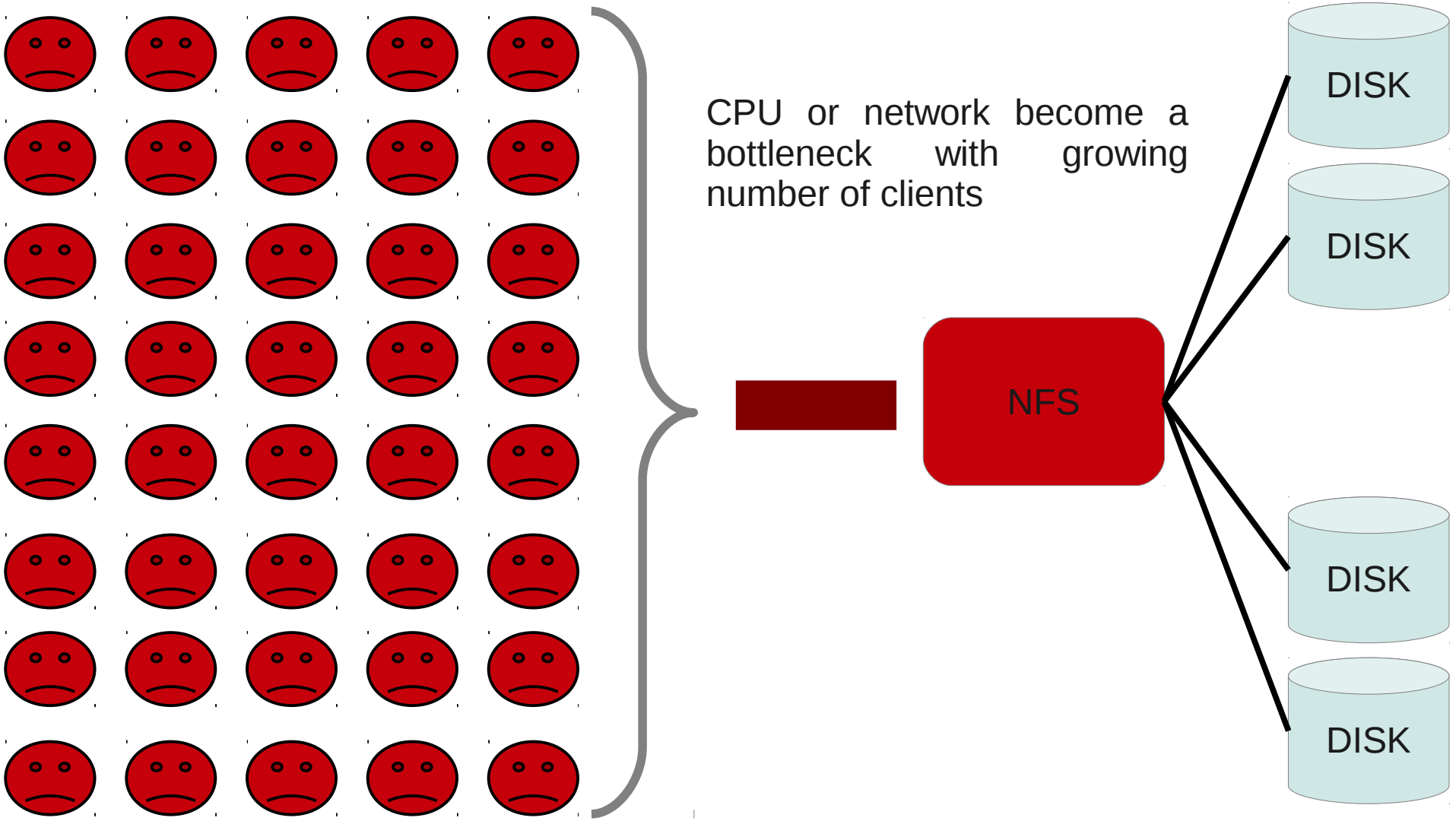
NFSv4.1

- Provides access to files and directories
- Stateful
  - Keeps track of OPEN/CLOSE (LOCK/UNLOCK)
  - Detects client/server reboot
  - Client controlled reply cache and EOS
- Aware of multihomed servers
- Detects retransmits
- Recovery from network disconnect

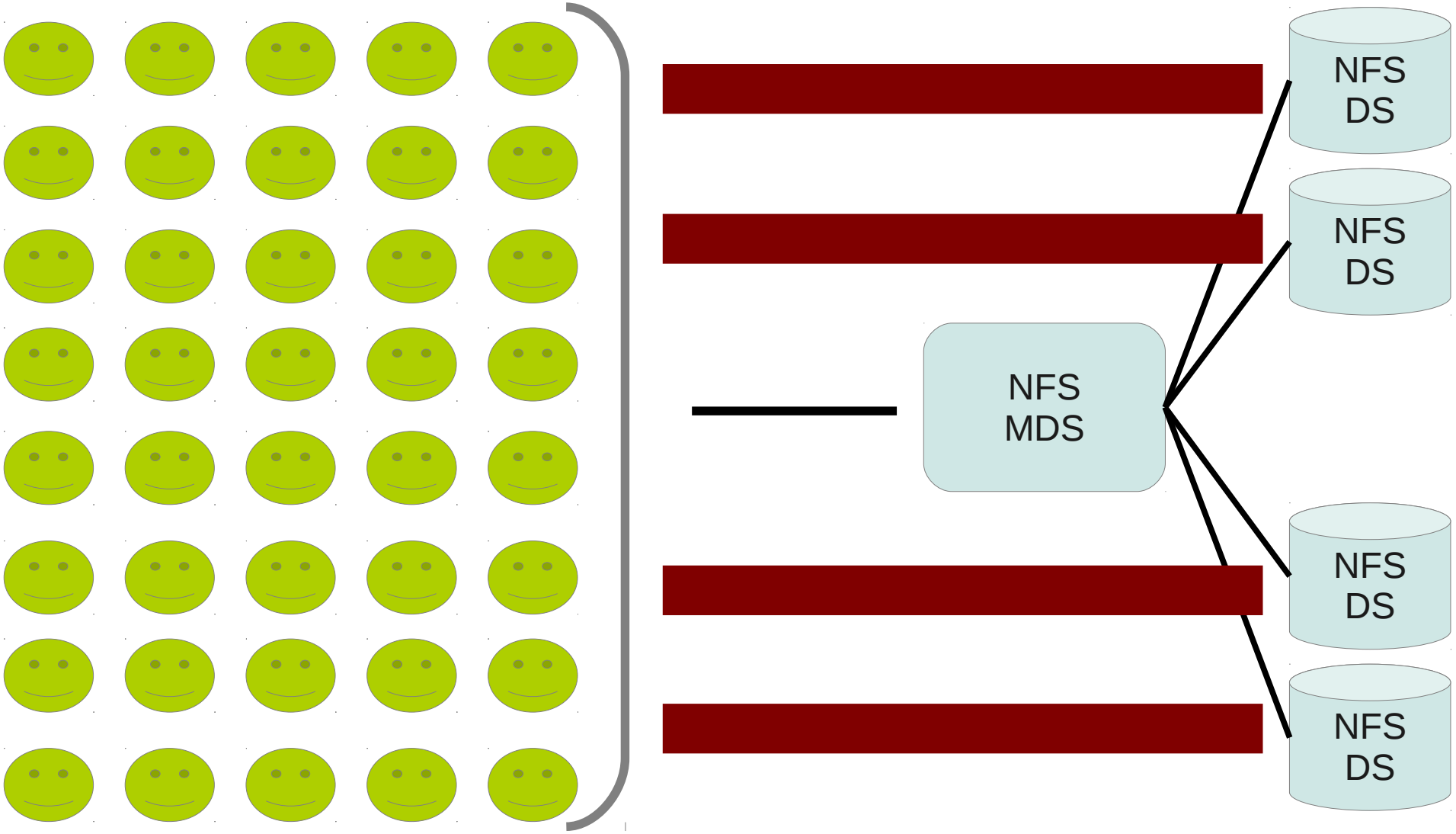
Classic NFS



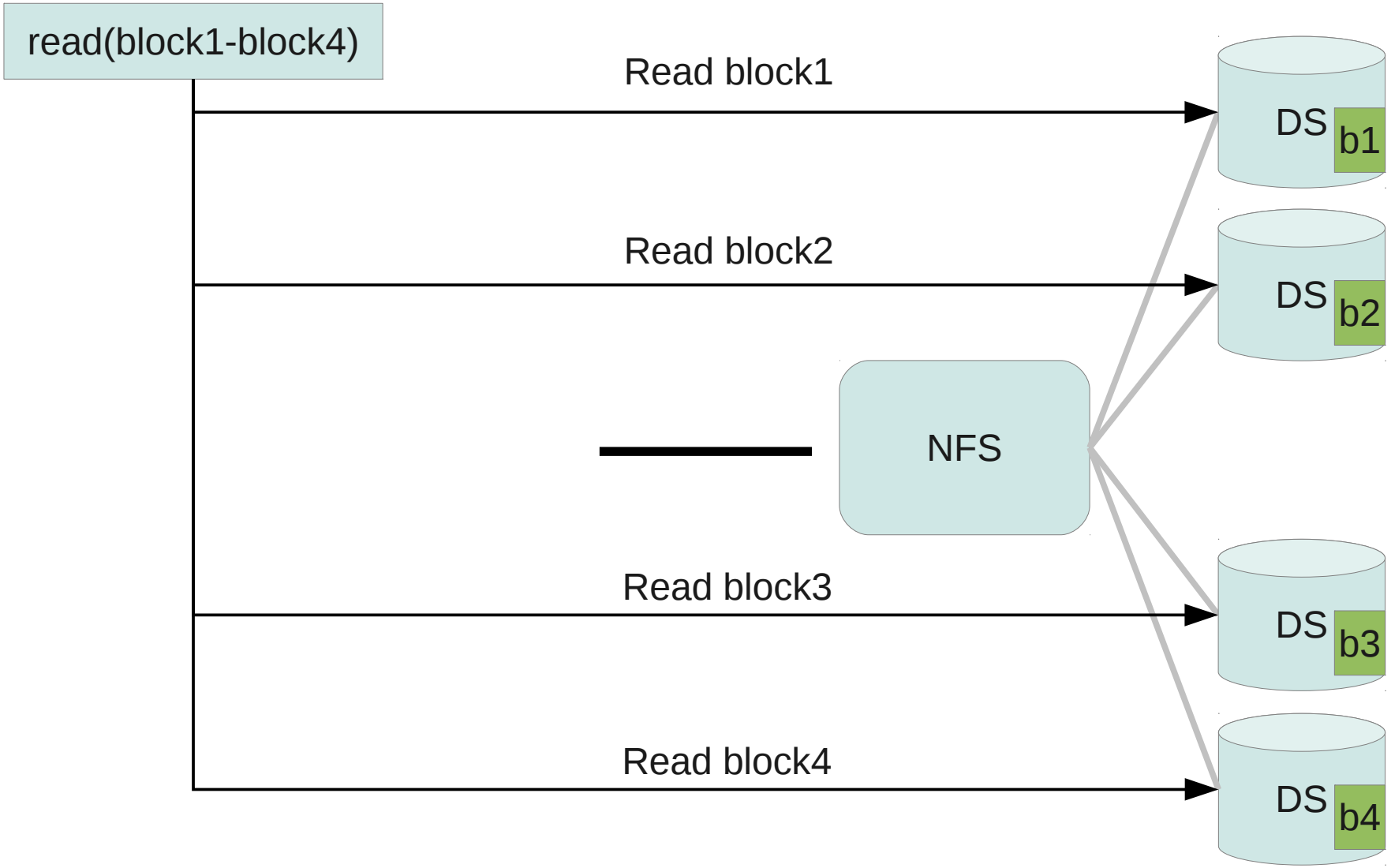

DISK
DISK
NFS
DISK
DISK


CPU or network become a bottleneck with growing number of clients
NFS
DISK
DISK
DISK
DISK

Parallel NFS



Parallel + Striping



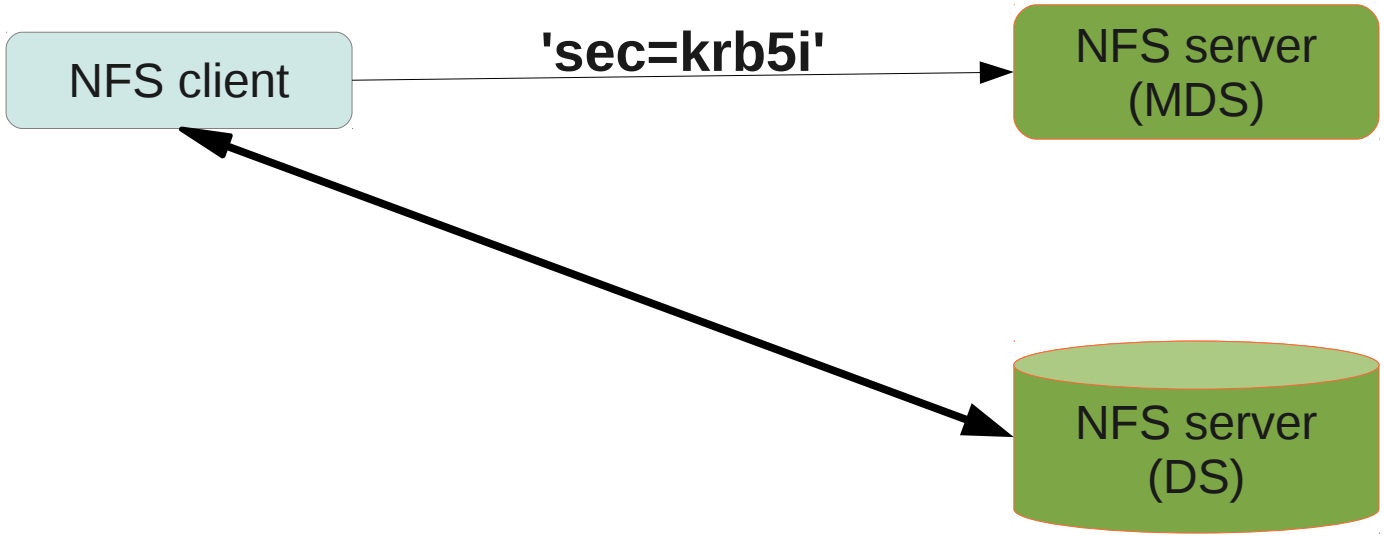
Parallel NFS

- Single Namespace, distributed data
- Client talks to Meta Data Server for metadata only
- Bandwidth and performance grow with number of Data Server nodes
- File striping (like raid0)
- Enforces the same security on DS (pools) as on MDS

Security

- To verify client credentials RPCSEC_GSS is used
  - Krb5 implementation supported by all clients/server
- Three type of Quality of Protection (QOP):
  - NONE – Auth only. Checksum protection of RPC header
  - INTEGRITY – Checksum protection of RPC messages
  - PRIVACY – full encryption of RPC messages
- Security flavor used on mount enforced to IO traffic as well

Security

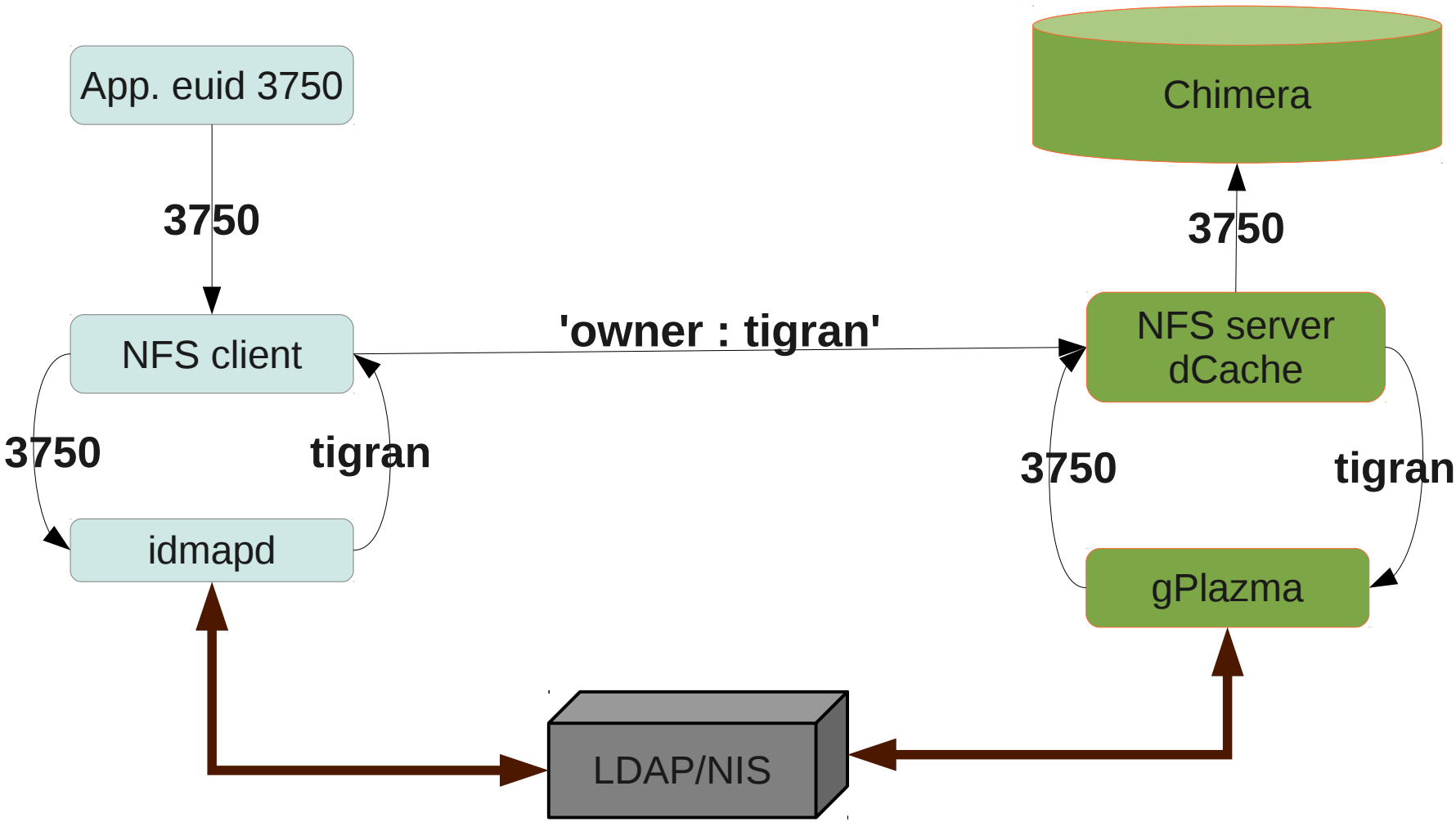


Client always will use the same security flavor and Quality of Protection for all RPC traffic to MDS and DS.

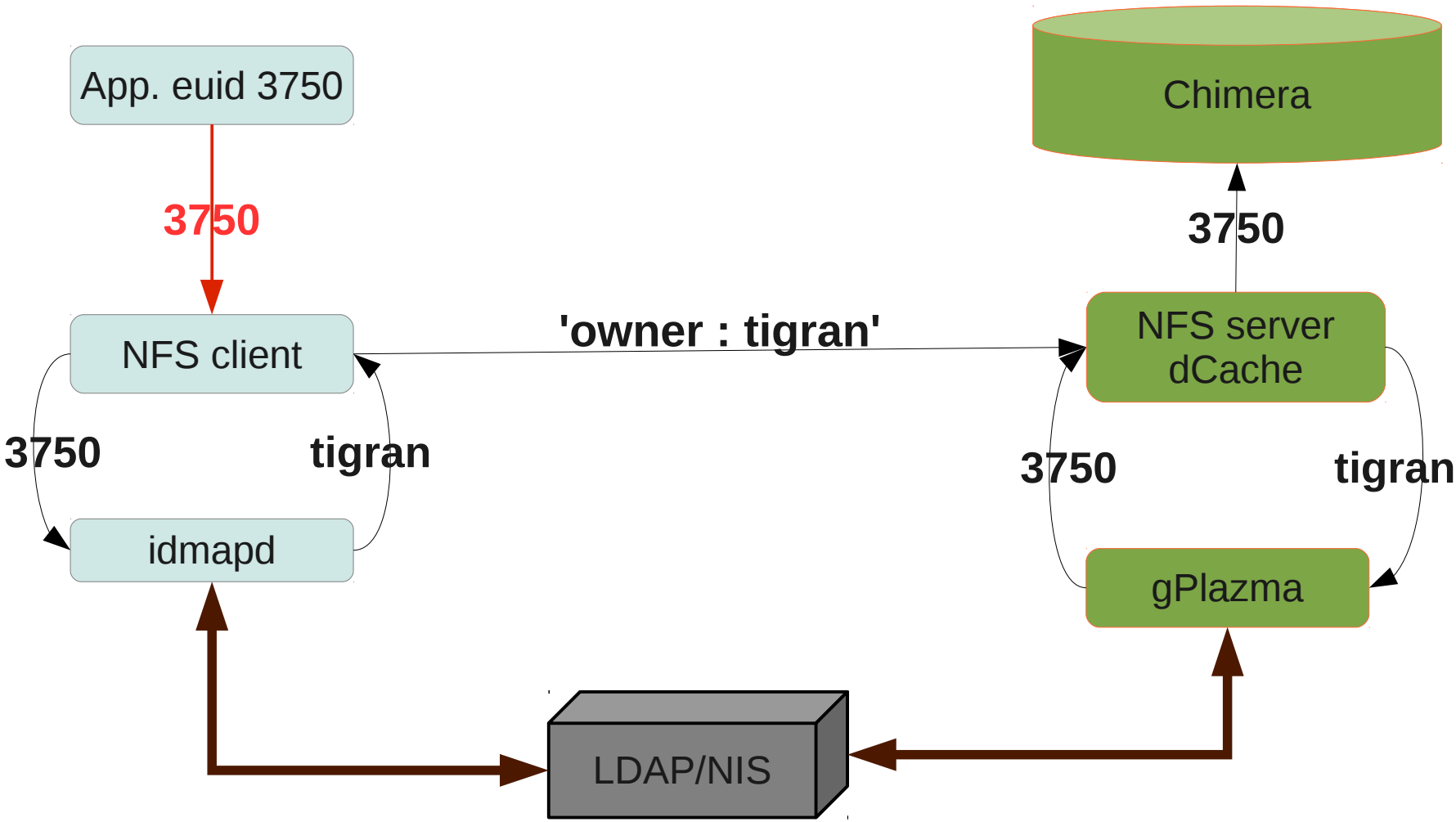
ID mapping

- All principals are utf-8 strings
  - No UID, no GID
- To resolve conflict NFSDOMAIN used:
  - "tigran@desy.de" vs. "tigran@cern.ch"
  - E.g. talk to corresponding mapping service
- Mapping delegated to client and server
  - Client and server may use different mapping services/sources
  - Windows client and server do not need numeric ID
- Best with ldap or nis
- Linux client can use numeric strings
  - "124" => 124

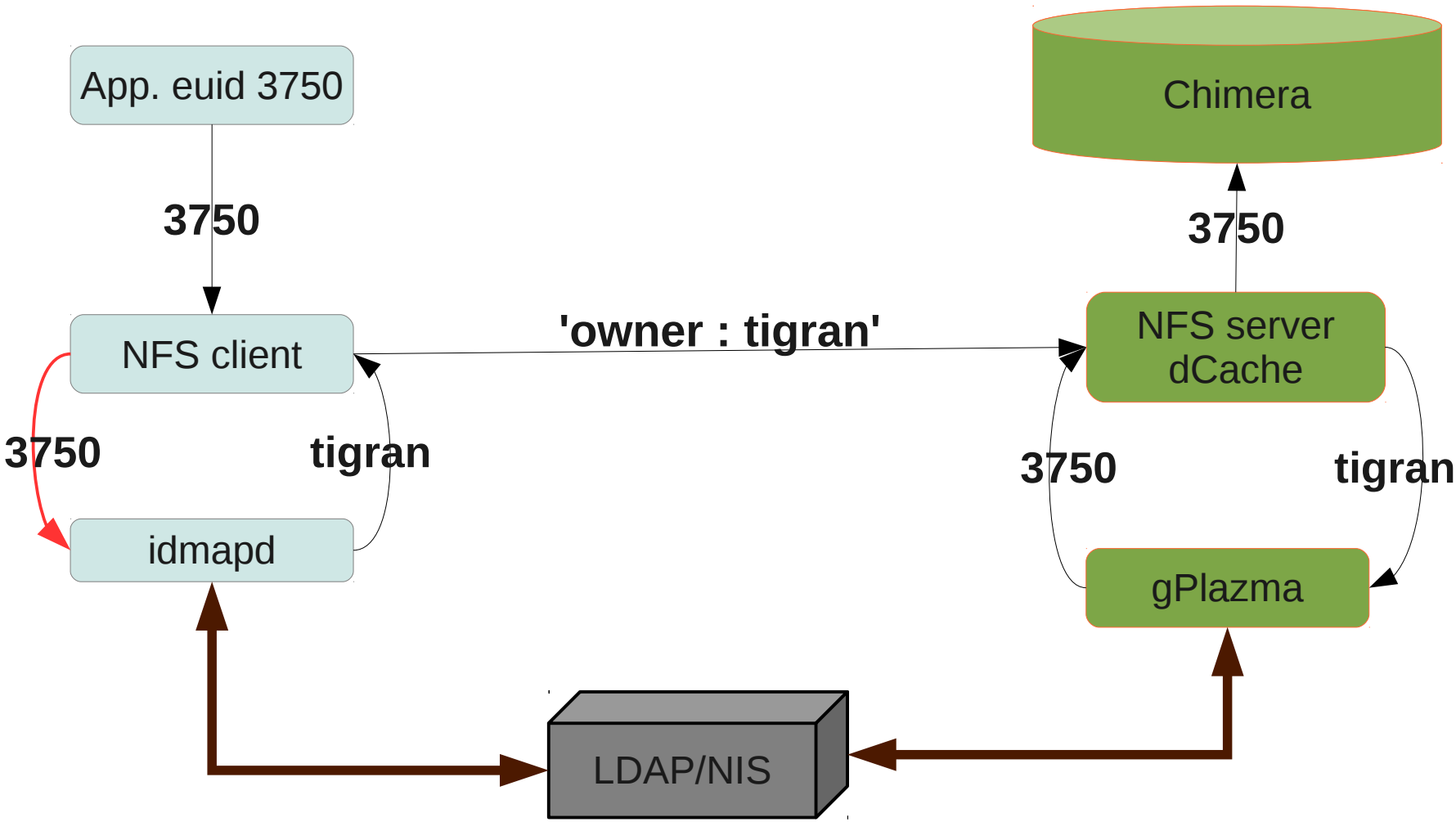
idmapping


App. euid 3750
3750
NFS client
3750
tigran
idmapd
'owner : tigran'
Chimera
3750
NFS server
dCache
3750
tigran
gPlazma
LDAP/NIS

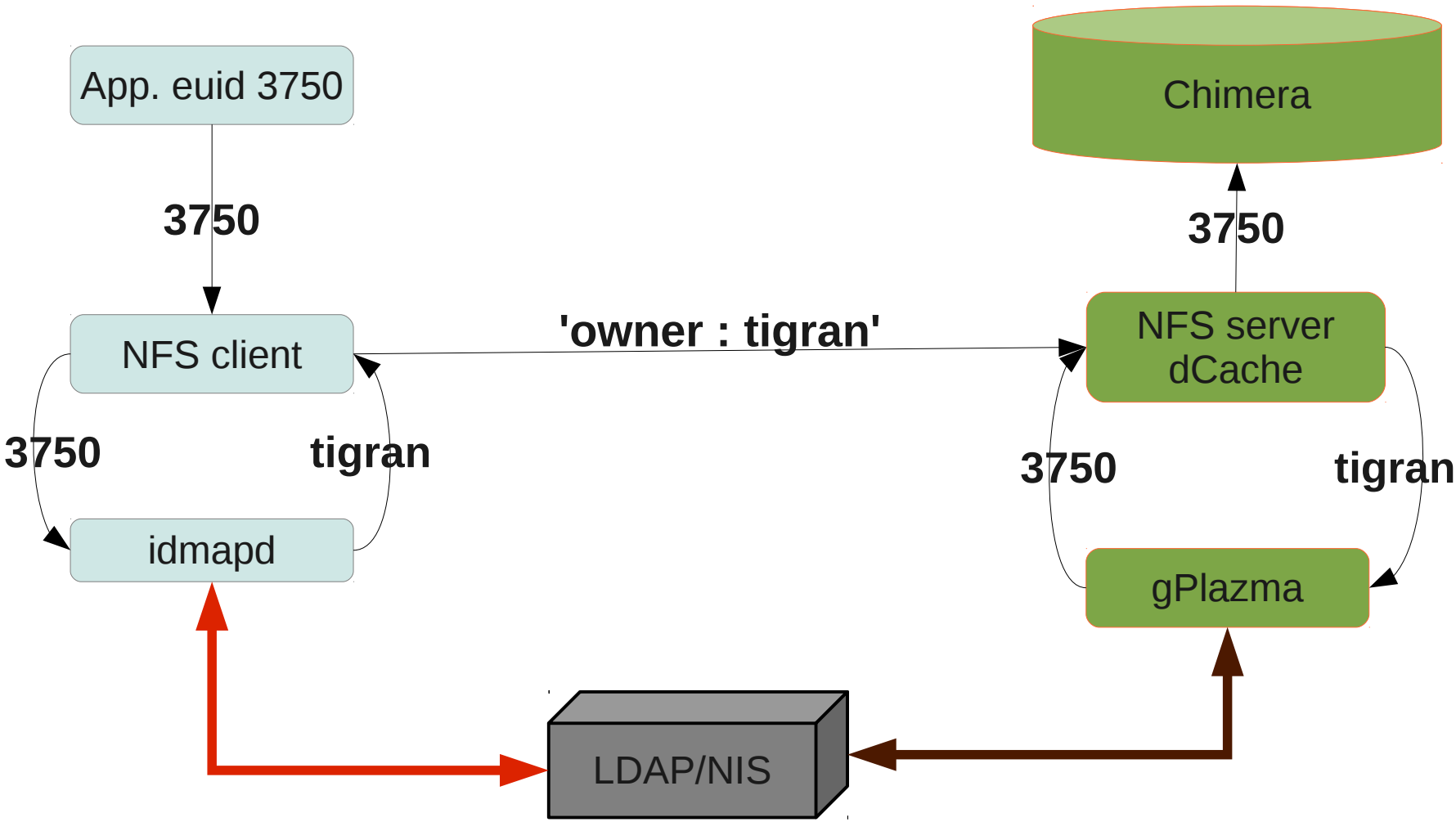
idmapping


App. euid 3750
3750
NFS client
3750
tigran
idmapd
'owner : tigran'
Chimera
3750
NFS server
dCache
3750
tigran
gPlazma
LDAP/NIS

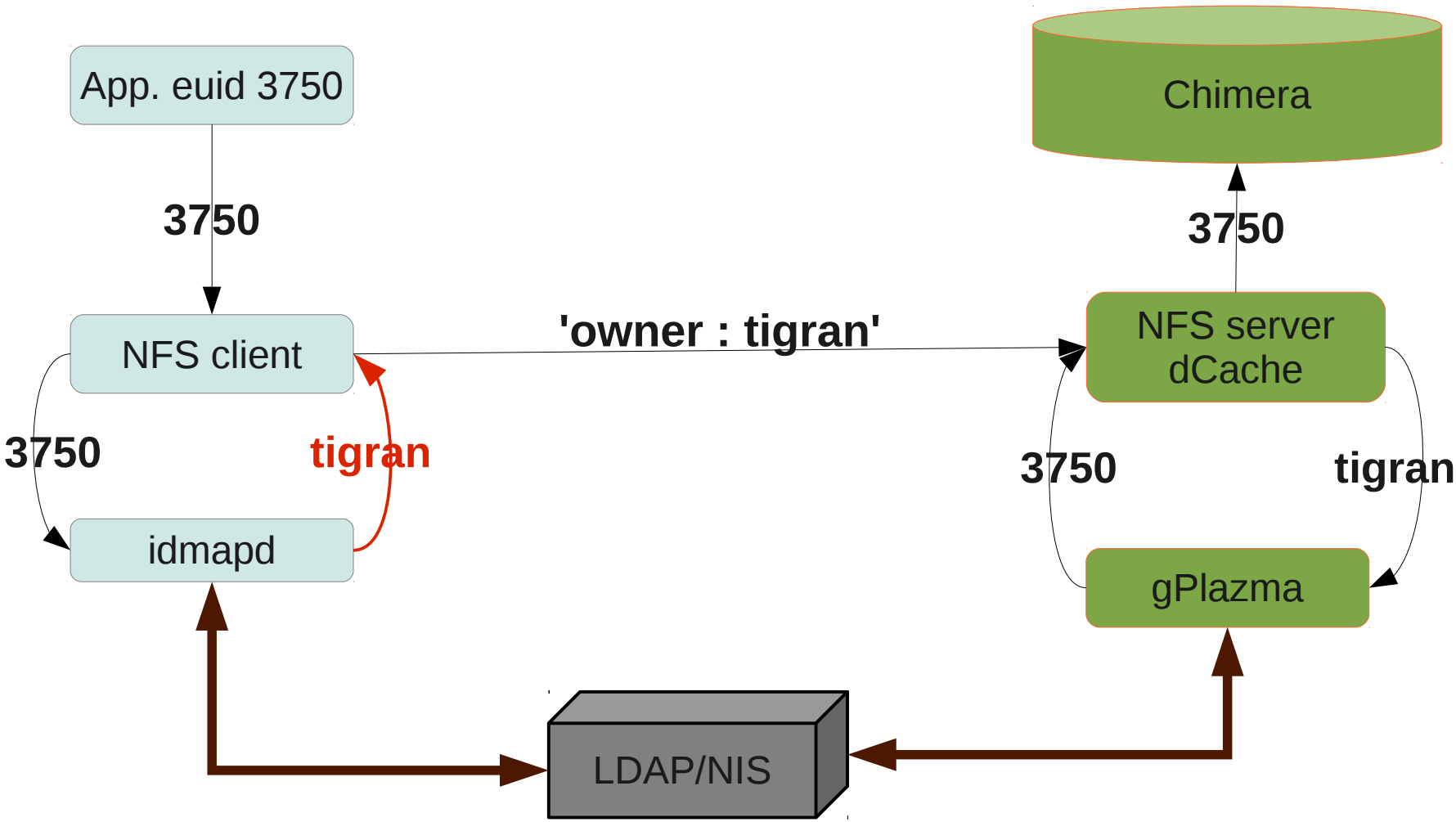
idmapping


App. euid 3750
3750
NFS client
3750
tigran
idmapd
'owner : tigran'
Chimera
3750
NFS server
dCache
3750
tigran
gPlazma
LDAP/NIS

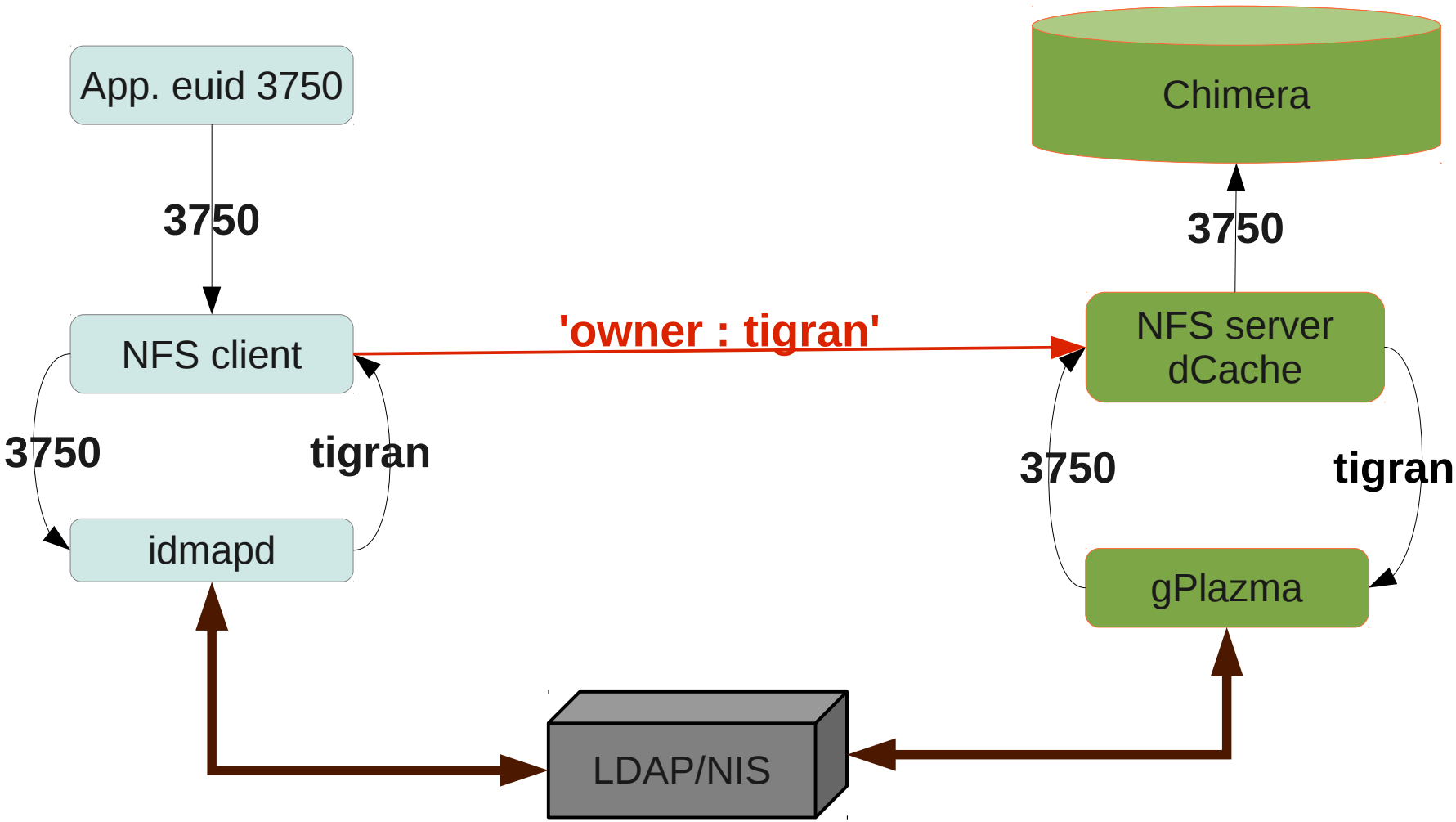
idmapping


App. euid 3750
3750
NFS client
3750
tigran
idmapd
'owner : tigran'
Chimera
3750
NFS server
dCache
3750
tigran
gPlazma
LDAP/NIS

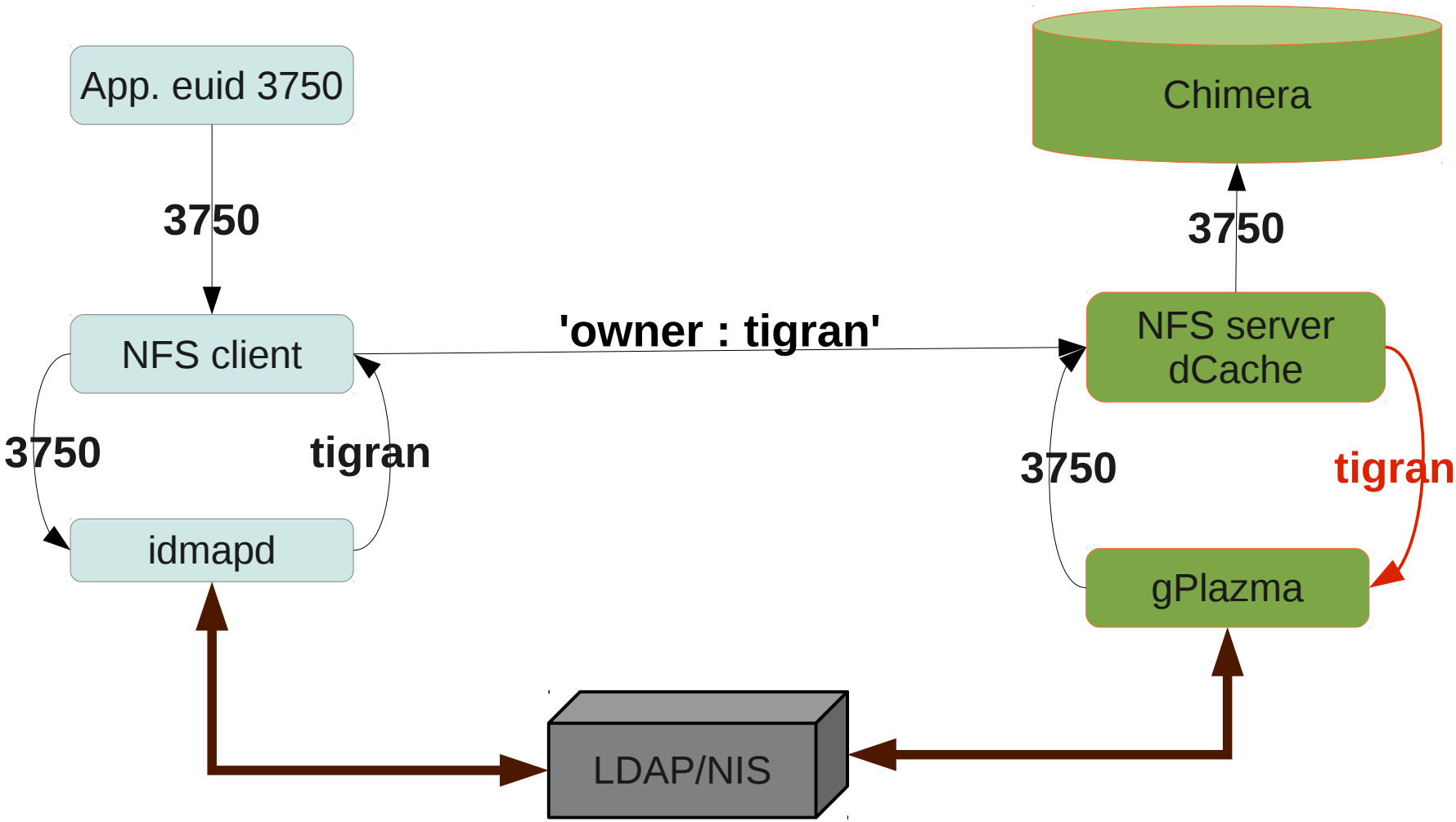
idmapping


App. euid 3750
3750
NFS client
3750
tigran
idmapd
'owner : tigran'
Chimera
3750
NFS server
dCache
3750
tigran
gPlazma
LDAP/NIS

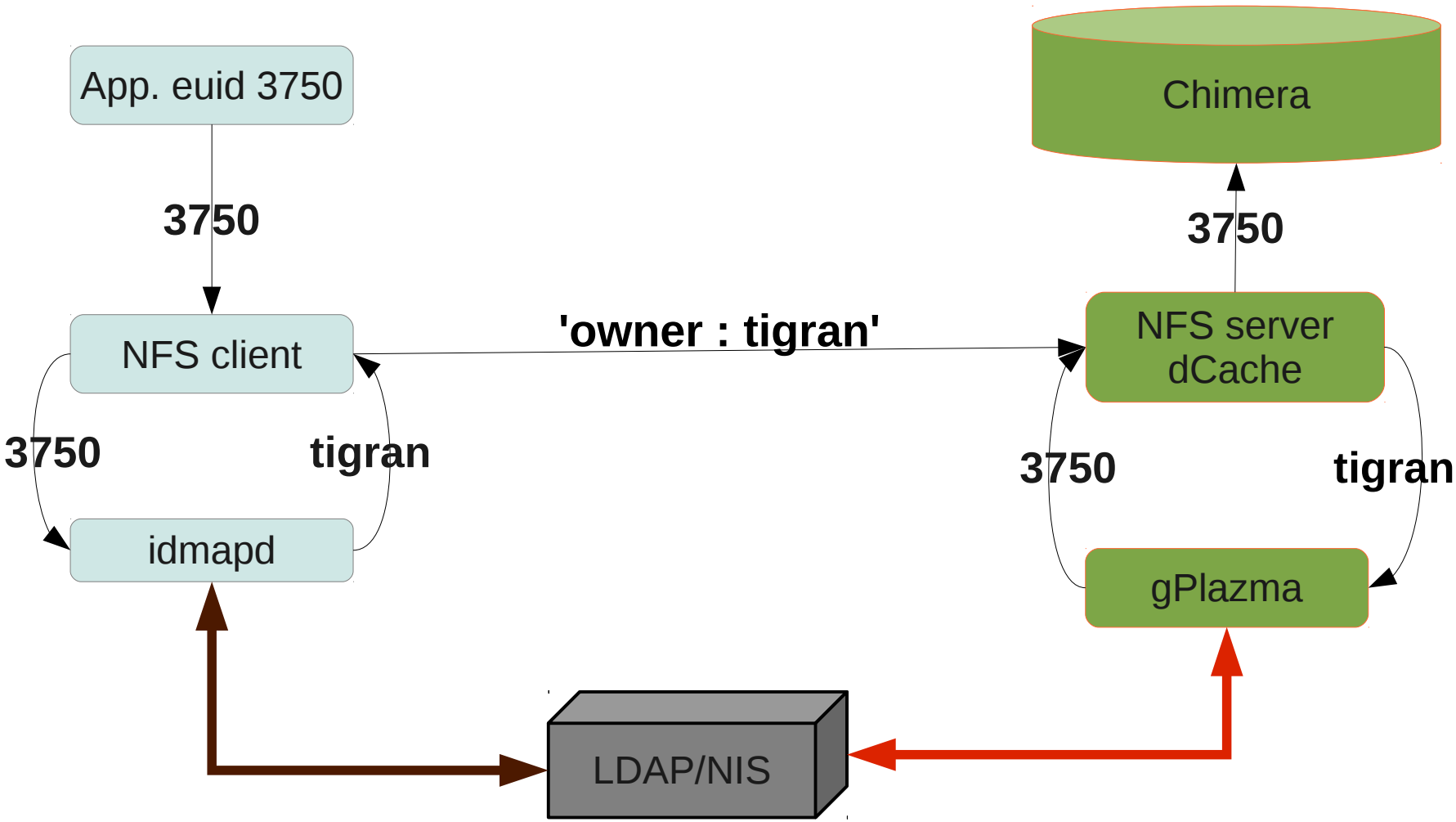
idmapping


App. euid 3750
3750
NFS client
3750
tigran
idmapd
'owner : tigran'
Chimera
3750
NFS server
dCache
3750
tigran
gPlazma
LDAP/NIS

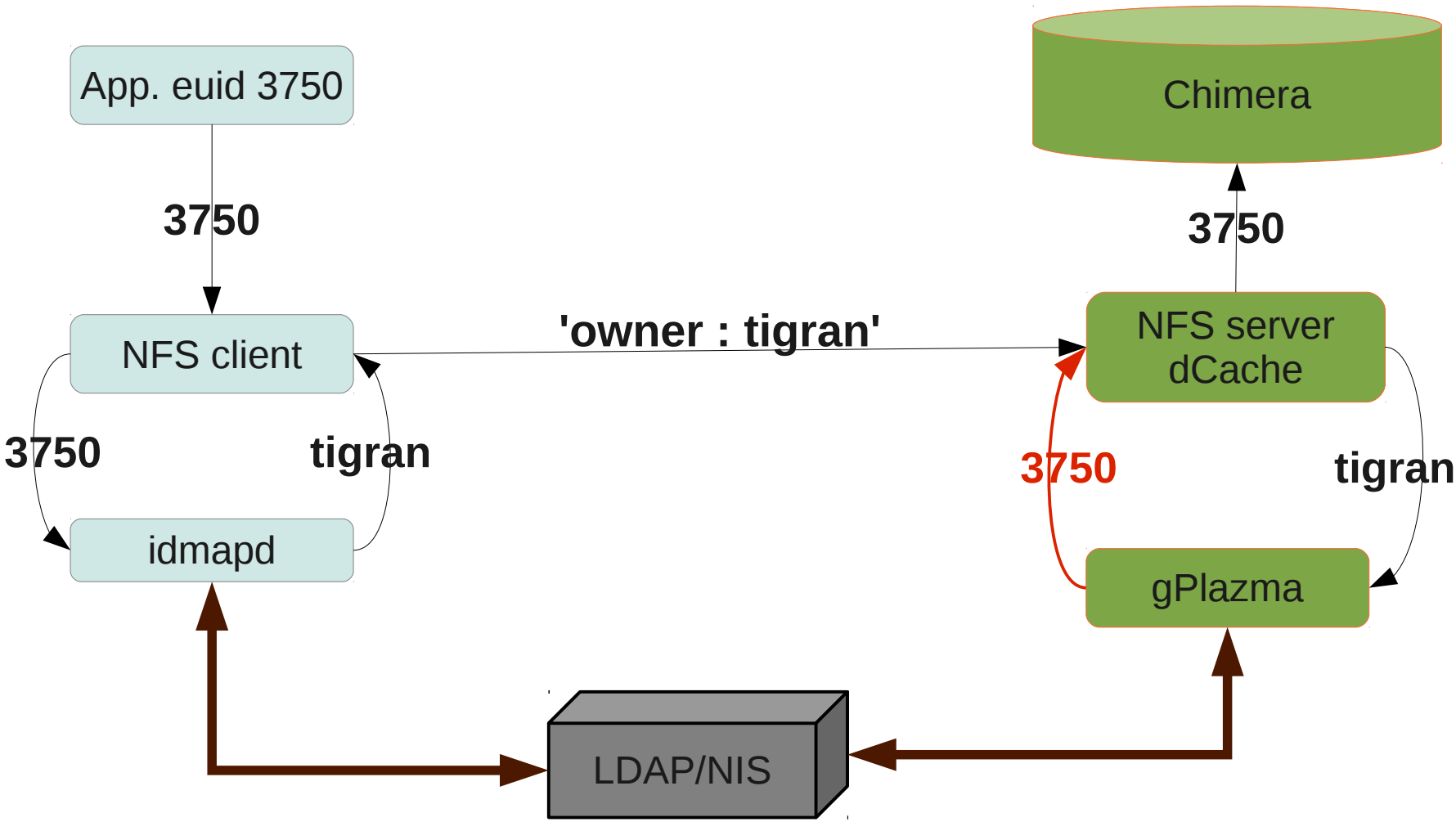
idmapping


App. euid 3750
3750
NFS client
3750
tigran
idmapd
'owner : tigran'
Chimera
3750
NFS server
dCache
3750
tigran
gPlazma
LDAP/NIS

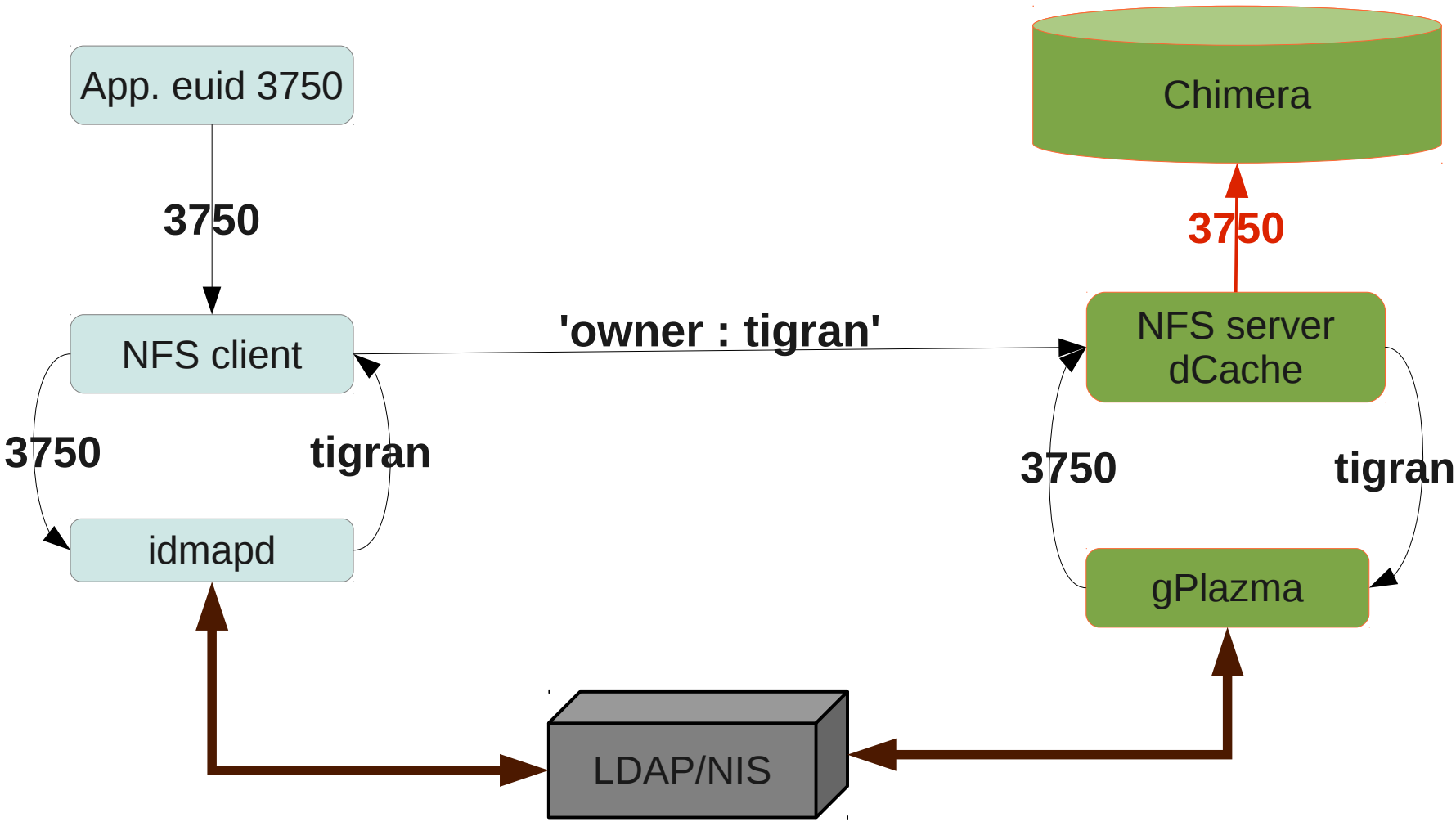
idmapping


App. euid 3750
3750
NFS client
3750
tigran
idmapd
'owner : tigran'
Chimera
3750
NFS server
dCache
3750
tigran
gPlazma
LDAP/NIS

idmapping


App. euid 3750
3750
NFS client
3750
tigran
idmapd
'owner : tigran'
Chimera
3750
NFS server
dCache
3750
tigran
gPlazma
LDAP/NIS

idmapping


App. euid 3750
3750
NFS client
3750
tigran
idmapd
'owner : tigran'
Chimera
3750
NFS server
dCache
3750
tigran
gPlazma
LDAP/NIS

Why so complicated?

- Your favorite OS may not use numeric id
  - MS Windows uses principals
- Your numeric ID on client and server may be different
  - My ID on laptop *500*, NIS *3750*

Existing servers

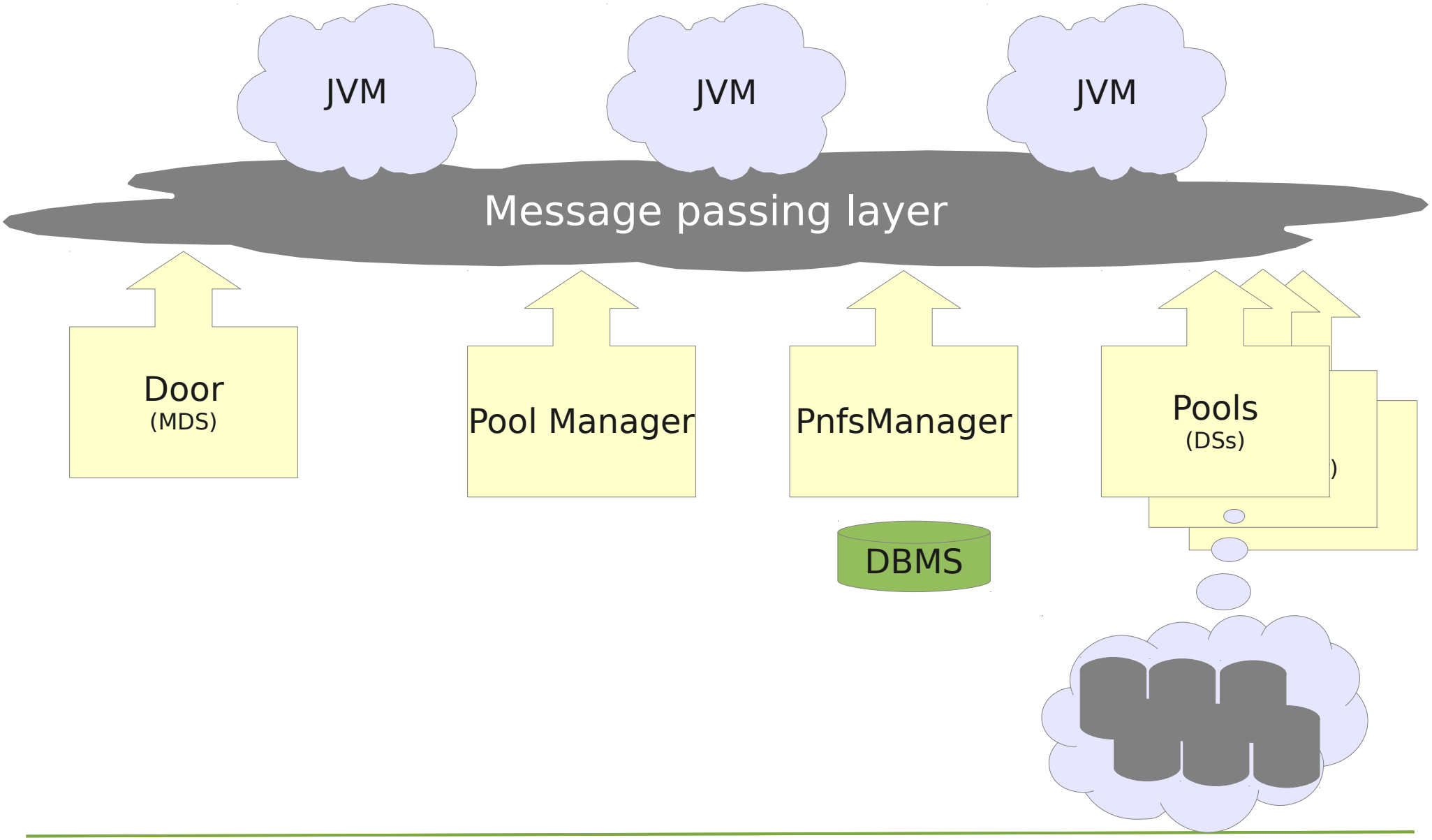
- **NETAPP** (ONTAP 8.1 cluster mode, running at DESY)
- **dCache**
  - Production ready 1.9.12
  - In production since XXX
- SONAS (IBM), Panasas, EMC, LinuxBox will be ready by 2Q 2013

Existing clients

- Linux
  - 2.6.39 first usable kernel
  - Part of RHEL 6 (SL6), Fedora >= 15, Debian-unstable
  - Oracle UEK2 for RHEL5 (sl5) + updates!
    - No AFS and some other kernel modules
- Windows 7 64 bit
  - Opensource client from CITI
- VMware ESX (pNFS client in VMware hypervisor)

dCache implementation

dCache NFS in one slide



Implementation Status

- No file striping yet
- Supports RPCGSS_SEC (kerberos5 only)
  - Krb5, krb5i and krb5p
  - Even with Windows AD as KDC
- Supports IPv6
- Can be integrated with NIS and/or local passwd file
  - Implemented as gPlazma plugins.

Still Missing

- Extended attribute support
  - Will provide native access (setfattr/getfattr) to checksum, AL and RP
- Only unix permission bits are supported
  - ACL expected by dcache-2.4, set/getfacl works today.
- No striping
  - "Striping on read" will come soon.
- No IO through Meta Data Server (door)
  - A problem if pool crashed/restarted

Limitations

- dCache files are immutable
- No support of byte range locks
- No support of multiple writers

ACLs with NFSv4.1 in dCache

- Will be available in dcache 2.4 (or 2.3)
- Main focus on predictable semantic
  - Unix mode and ACL coexistence
  - NFSv3, FTP and NFS4 coexistence

Prerequisites for NFSv4.1

1. Configured gPlazma
2. Kerberos5 keytab with nfs principals for NFS door and all pools.
3. Open TCP port 2049 in NFS door
4. Open TCP range 'net.lan.port.min:net.lan.port.max' on pools
   - Default is 33115:33145 (shared with dcap and xrootd)
5. Client with pNFS aware kernel (SL6.2 and FC16 recommended)
6. Configured rpc.idmapd to match *NFS DOMAIN* with server
7. Kerberos5 keytab with host principal
8. Start dCache, mount on client and access the data.

Terminology

		dCache equivalent
MDS	Meta Data Server	dCache NFSv4.1 door
DS	Data server	dCache pool
QOP	Quality Of Protection	

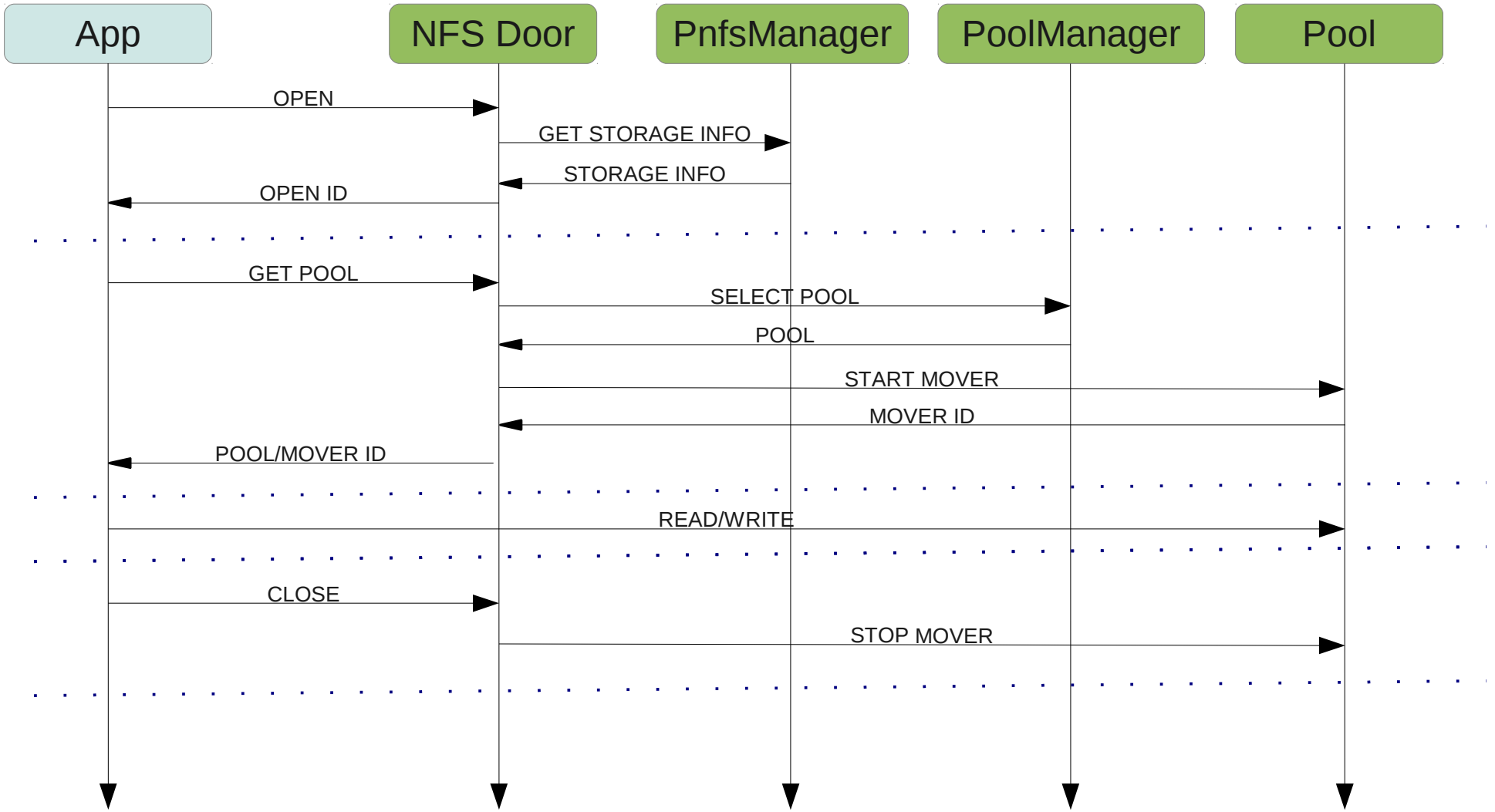
Typical IO scenario

- Open
- Read/write
- close

OPEN+READ/WRITE+CLOSE

- Open
  - Open/Create a file
  - Find appropriate pool
  - Start a mover
- Read/write
  - READ/WRITE
- Close
  - Release mover
  - Close the file

OPEN+READ/WRITE+CLOSE



Setup: enable door

```
# layout file
# nfs-4.1 section
(nfsdoorDomain/nfsv41)
nfs.domain=desy.de
nfsIoQueue=nfs
#  NFS uses direct access to  Chimera database
chimera.db.name = chimera
chimera.db.host = localhost
chimera.db.user = chimera
chimera.db.password =
#
```

Setup: tweak rpcbind

```
# /etc/sysconfig/rpcbind
# enable insecure mode
RPCBIND_ARGS="-i"
#
```

Setup: start door/pool

```
# layout file
(poolDomain/pool)
name=pool-xxx
...
poolIoQueue=nfs
#
# use one of the ports from range for movers.
# only single port used per pool
net.lan.port.min=33115
net.lan.port.max=33145
#
```

Setup: tweak pool

```
# layout file
(poolDomain/pool)
name=pool-xxx
...
poolIoQueue=nfs
#
# use one of the prots from range for movers.
# only single port used per pool
net.lan.port.min=33115
net.lan.port.max=33145
#
```

Setup: gplazma

- Identity plugin required
  - Nsswitch – uses local accounts, based on /etc/nsswitch.conf
    - Uses accounts from gplazma host!
  - Nis – uses site NIS server
- nfs.domain on door MUST match with 'Domain' on client
  - Linux conf at /etc/idmapd.conf

Setup: gplazma

```
# layout file
(gplazmaDomain/gplazma)
gplazma.nis.server = nisserv6.desy.de
gplazma.nis.domain = desy.de
#

# gplazma.conf
identity requisite nis
```

Setup: tweak idmap

```
# dcache.conf


# maximal number of entries in the cache
nfs.idmap.cache.size = 512
# cache entry maximal lifetime
nfs.idmap.cache.timeout = 30
# Time unit used for timeout.
# SECONDS | MINUTES | HOURS | DAYS)
nfs.idmap.cache.timeout.unit = SECONDS
```

Setup: read-to-go

- dcache start

Anatomy of NFS package