NetApp®

Go further, faster®

NFS Server-side Copy
Design and Standardization

James Lentini
Advanced Technology Group
NetApp, Inc.

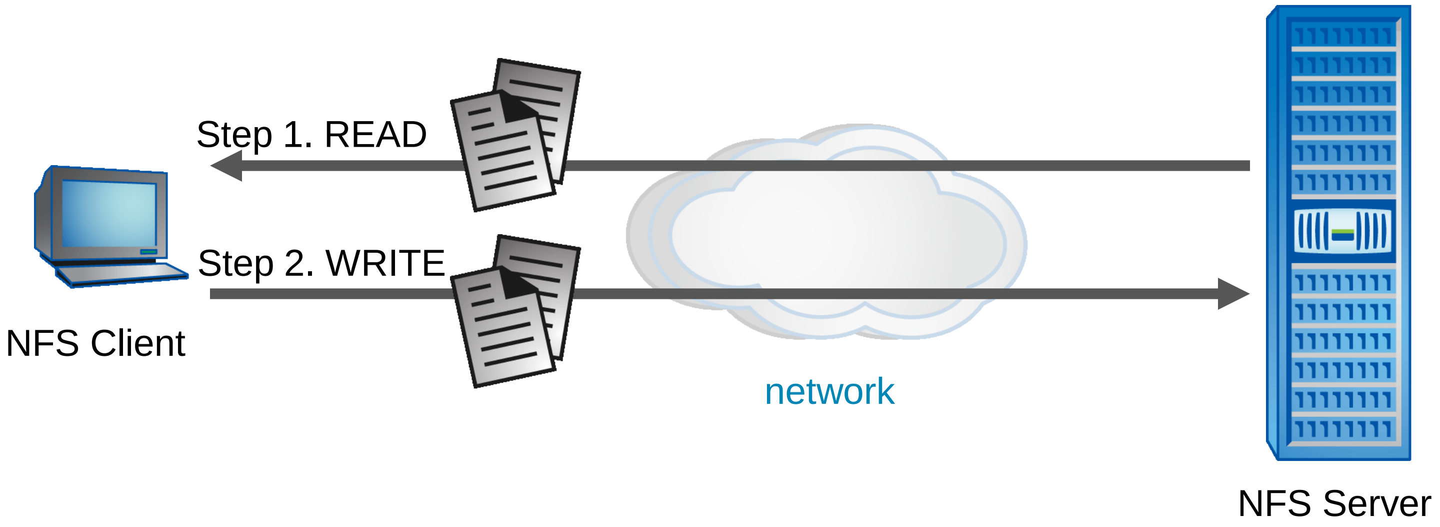
jlentini@netapp.com

Summary

- NFS server-side copy offload is a set of operations that allow:
  - Copying a file on a single NFS server
  - Copying a file between two NFS servers.
- Server-side copy is a possible feature for NFSv4.2.

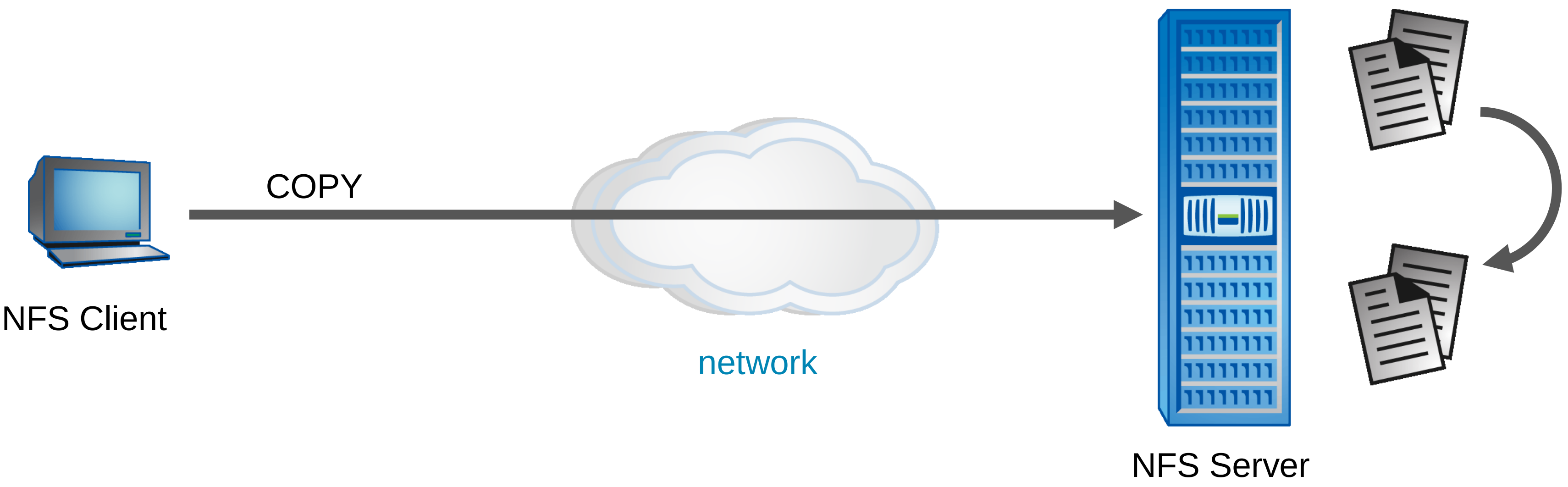
Copying with NFSv2/v3/v4[.1]

- The NFS client reads and writes the file over the network.
- Wastes client and network resources.



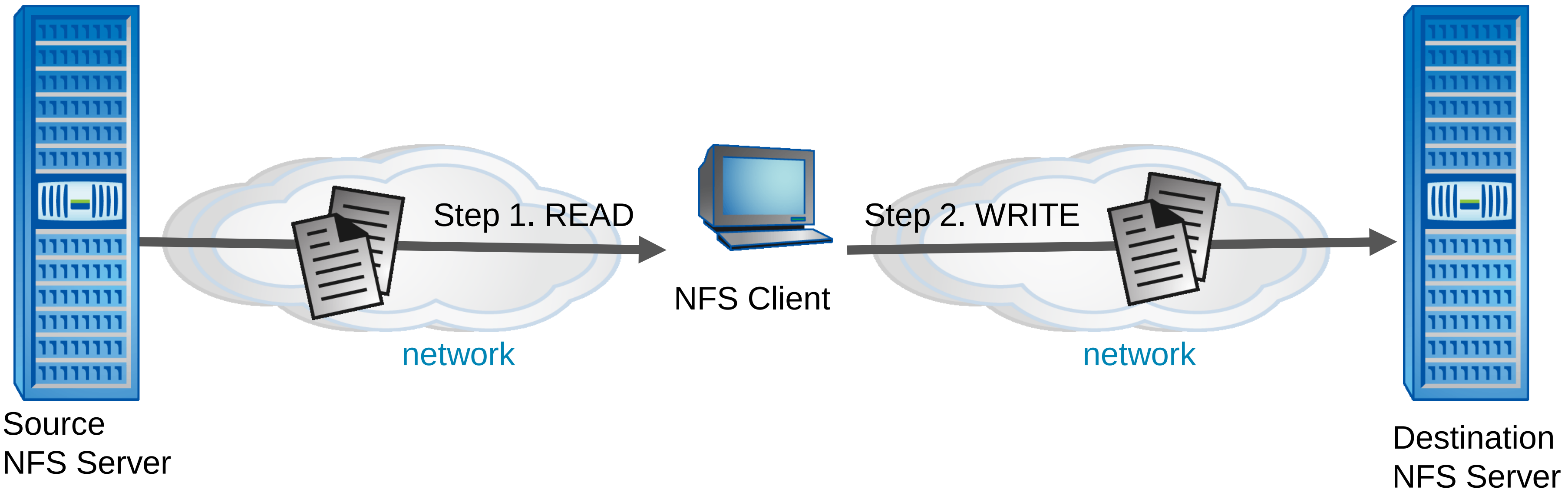
Copying with Server-side Offload

- The NFS client instructs the server to perform the copy.
- Saves client and network resources.



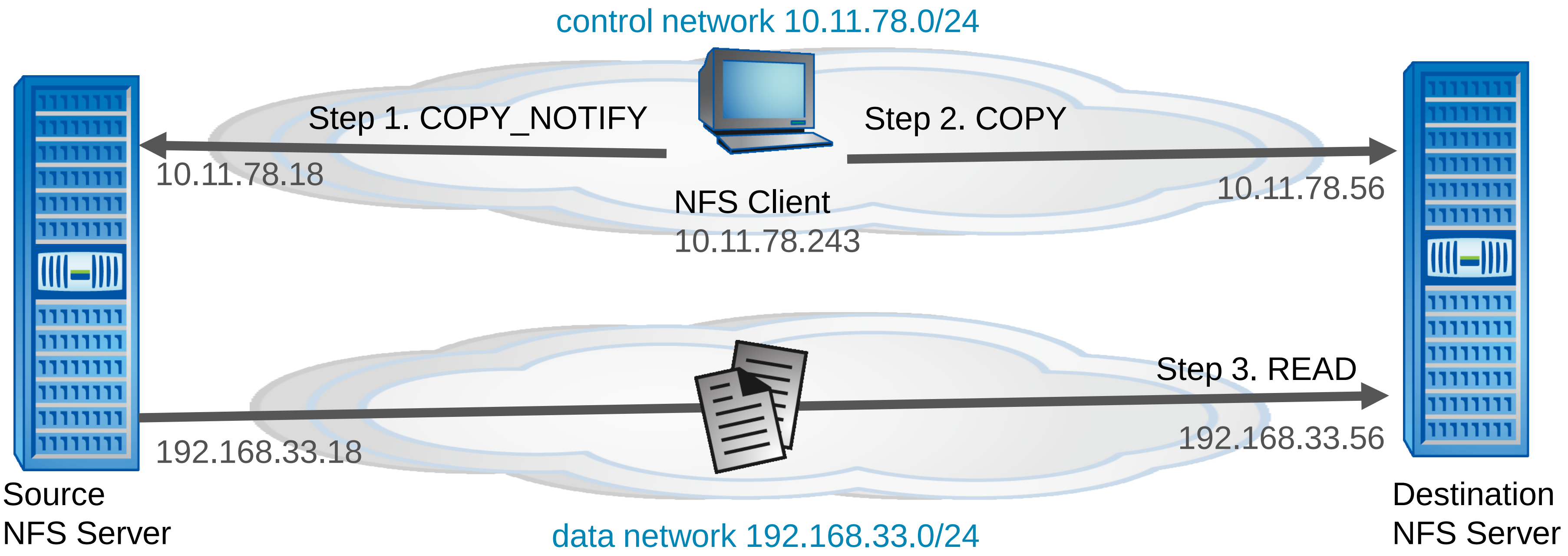
Copying between NFS Servers with NFSv2/v3/v4[.1]

- Client reads the file from the source server and writes the file to the destination server.
- Client is an extra network hop between the source and destination.



Copying between NFS Servers with Server-side Offload

- Client sets up the copy between the servers.
- Removes client hop and (optionally) allows a high performance server data network to be used.



Uses Cases

- In general, this feature is useful whenever data is copied from one location to another.
- **File Restore**: It is useful when copying the contents of a backup to the active file system.
- **Virtualized Environments:** Copy offload allows a hypervisor to efficiently:
  - Backup a VM’s storage
  - Clone a VM’s storage
  - Migrate a VM’s storage

Design Choice: Supported Object Types

- What types of objects will the copy operations support?
  - Files?
  - Directories?
  - Namespace junctions?
- Proposal is to support copies of regular files.
- Simplifies the protocol
- Directory copies can by synthesized using multiple file copies and directory creates.
- Namespace junctions can be copied using the FedFS ONC RPC Admin protocol.

Design Choice: Synch vs. Asynch

- Does the NFS server perform the copy synchronously or asynchronously?
- Large files could require significant time to copy.
  - Problematic for a synchronous model.
- Proposal allows for both synchronous and asynchronous copies
  - Server decides what type to use

Design Choice: Server-to-server Protocol

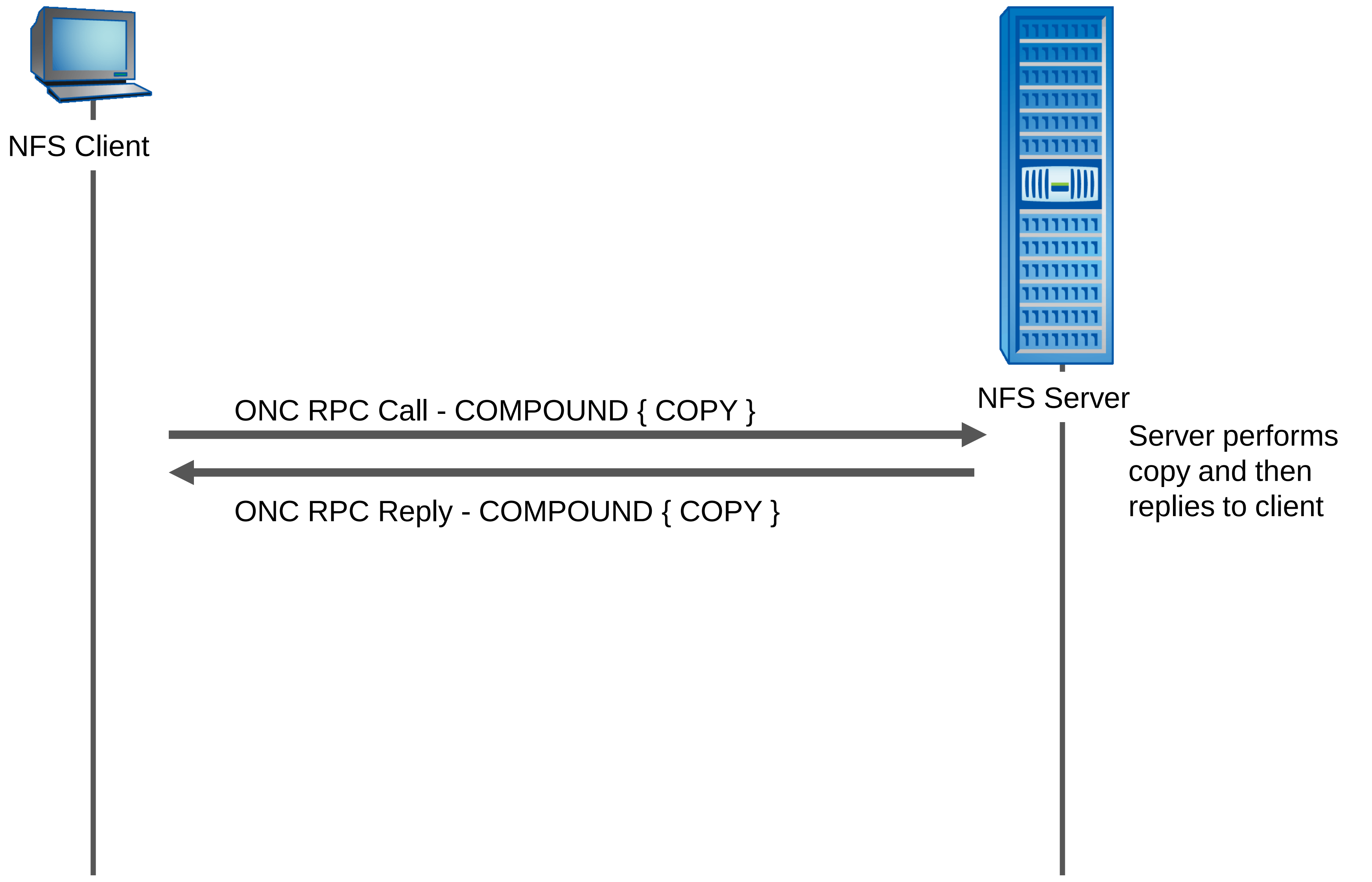
- The protocol supports intra- and inter- server copies
  - intra-server copy: source and destination file on the same fileserver
  - inter-server copy: source and destination file on different fileservers
- The proposal doesn't require a particular server-to-server copy protocol.
- NFSv4.1 is a good candidate for heterogeneous environments.
  - Standard protocols (FTP, HTTP, …) in addition to NFS are also supported.
- Proprietary protocols are possible in homogeneous environments.
  - The source and destination server may be using a clustered file system, no data may actually need to be copied or may have the same file system format allowing physical block-level replication.

Protocol

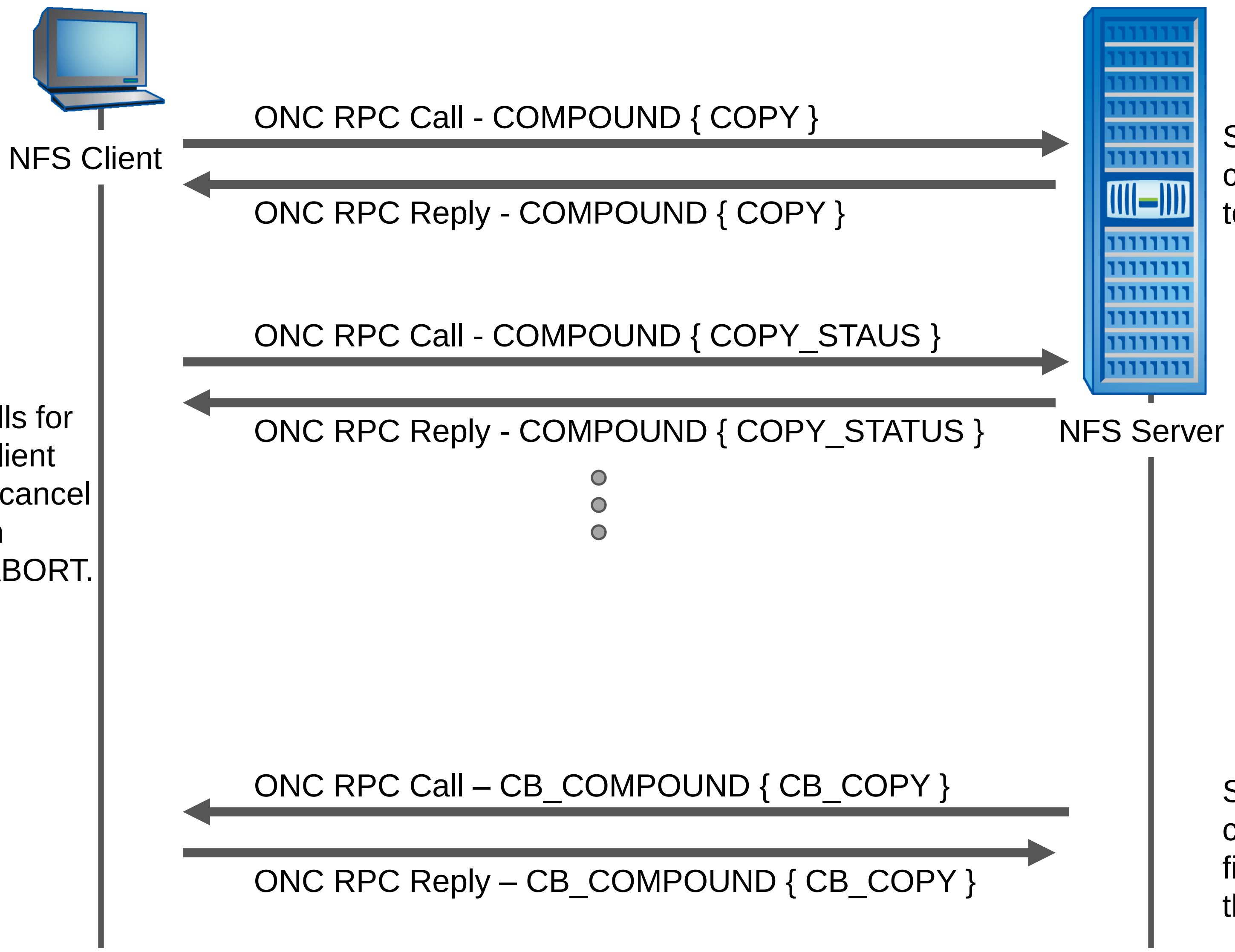
Operations

- **COPY_NOTIFY**: For inter-server copies, the client sends this operation to the source server to notify it of a future file copy from a given destination server for the given user.
- **COPY_REVOKE**: Also for inter-server copies, the client sends this operation to the source server to revoke permission to copy a file for the given user.
- **COPY**: Used by the client to request a file copy.
- **COPY_ABORT**: Used by the client to abort an asynchronous file copy.
- **COPY_STATUS**: Used by the client to poll the status of an asynchronous file copy.
- **CB_COPY**: Used by the destination server to report the results of an asynchronous file copy to the client.

Synchronous Intra-server Copy


NFS Client
NFS Server
ONC RPC Call - COMPOUND { COPY }
ONC RPC Reply - COMPOUND { COPY }
Server performs copy and then replies to client

Asynchronous Intra-server Copy

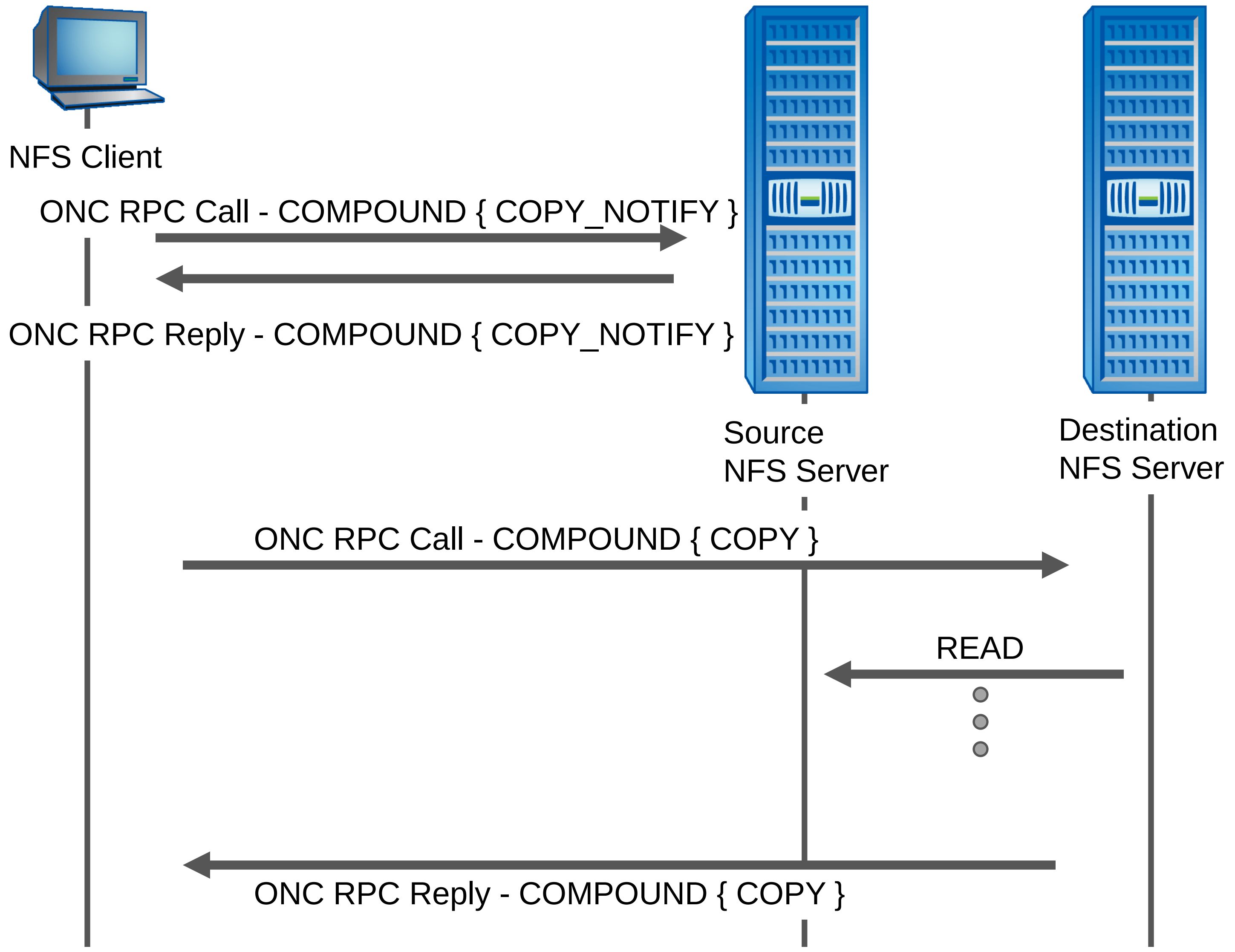

NFS Client
ONC RPC Call - COMPOUND { COPY }
ONC RPC Reply - COMPOUND { COPY }
Server starts the copy and replies to client.
ONC RPC Call - COMPOUND { COPY_STAUS }
ONC RPC Reply - COMPOUND { COPY_STATUS }
NFS Server
ONC RPC Call – CB_COMPOUND { CB_COPY }
ONC RPC Reply – CB_COMPOUND { CB_COPY }

Client polls for status. Client can also cancel copy with COPY_ABORT.

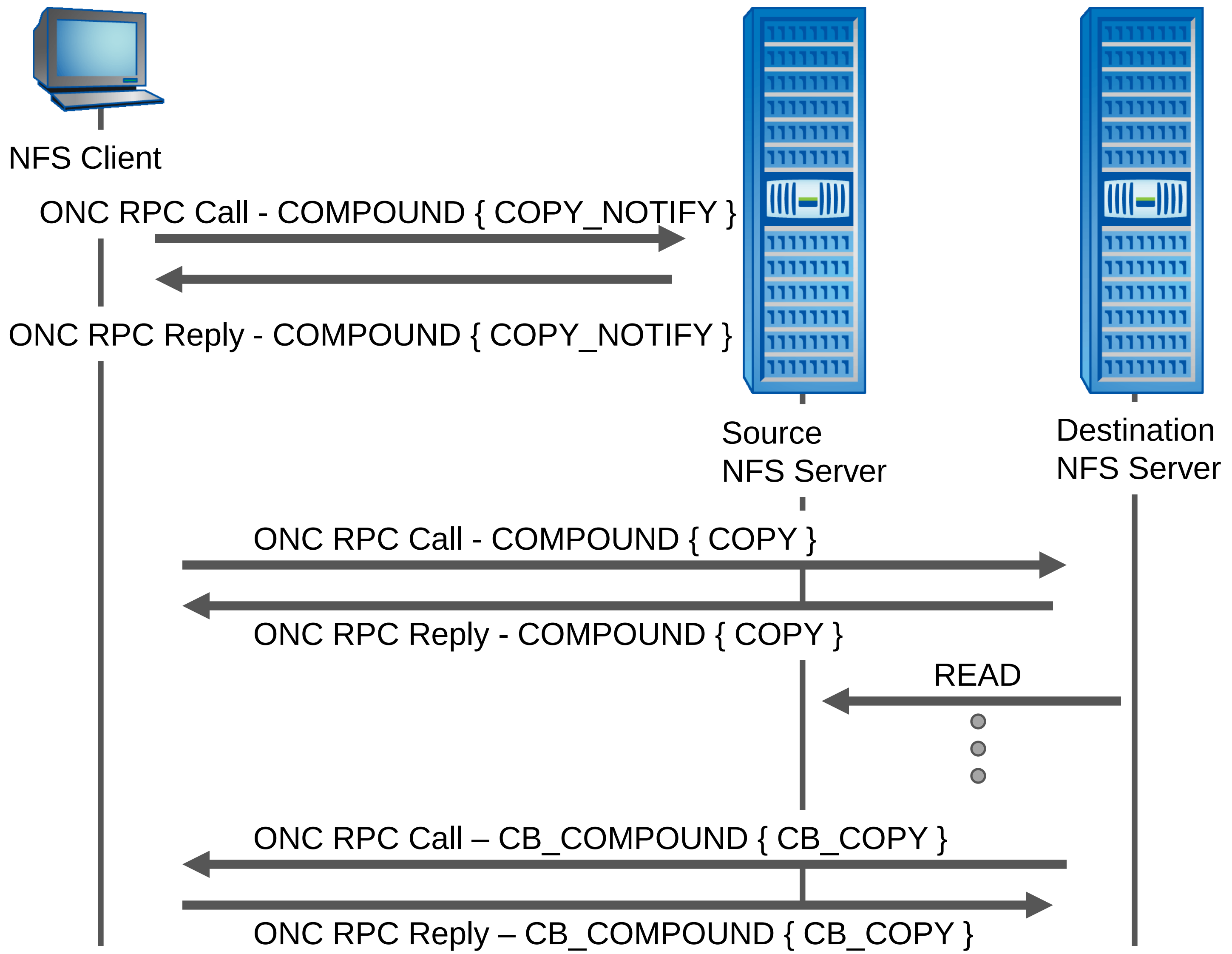
Server starts the copy and replies to client.

Server issues callback with final results of the copy

Synchronous Inter-server Copy


NFS Client
ONC RPC Call - COMPOUND { COPY_NOTIFY }
ONC RPC Reply - COMPOUND { COPY_NOTIFY }
Source NFS Server
Destination NFS Server
ONC RPC Call - COMPOUND { COPY }
READ
ONC RPC Reply - COMPOUND { COPY }

Asynchronous Inter-server Copy



NFS Client
ONC RPC Call - COMPOUND { COPY_NOTIFY }
ONC RPC Reply - COMPOUND { COPY_NOTIFY }
Source NFS Server
Destination NFS Server
ONC RPC Call - COMPOUND { COPY }
ONC RPC Reply - COMPOUND { COPY }
READ
ONC RPC Call – CB_COMPOUND { CB_COPY }
ONC RPC Reply – CB_COMPOUND { CB_COPY }

Security

Security

- Requirements:
  - flexible enough to allow for different server-to-server copy protocols.
  - compatible with using NFSv4.x as the server-to-server copy protocol.
  - no pre-configuration between the source and destination.
  - support mutual authentication between the participants (client, source server, and destination server).
- Supported mechanism:
  - RPCSEC_GSSv3 (IETF draft) for strong security
  - host-based security (e.g. AUTH_SYS)

RPCSEC_GSSv3 Security (1)

- Three new RPCSEC_GSSv3 privileges:
  - copy_from_auth_priv: established by the client on the source server to allow a copy operation from the specified destination server on behalf of the given user.
  - copy_to_auth_priv: established by the client on the destination server to allow a copy operation from the specified source server on behalf of the given user.
  - copy_confirm_auth_priv: for ONC RPC server-to-server copy protocols, established by the destination server on the source server to allow a copy operation on behalf of the given user.

RPCSEC_GSSv3 Security (2)

- Client establishes copy_from_auth_priv, source server creates <"copy_from_auth", user id, destination> record. Client sends COPY_NOTIFY using the copy_from_auth RPCSEC_GSSv3 handle. Source server annotates record with source filehandle.
- Client establishes copy_to_auth_priv, destination server creates <"copy_to_auth", user id, source> record. Client sends a COPY using the copy_to_auth RPCSEC_GSSv3 handle.
- The destination establishes a copy_confirm_auth_priv on the source. Subsequent ONC RPC requests from the destination of the source use the copy_confirm_auth_priv handle.

Host-based Security

- Without real security, only a minimal level of protection is possible.
- Unique URLs used to encode the destination's copy privilege and identify a specific copy.
- Source server returns URLs in COPY_NOTIFY reply:

  nfs://10.11.78.18//_COPY/10.11.78.56/_FH/0x12345

  nfs://192.168.33.18//_COPY/10.11.78.56/_FH/0x12345
- Destination server will identify itself by performing these operations:

  COMPOUND { PUTROOTFH, LOOKUP "_COPY" ; LOOKUP "10.11.78.56"; LOOKUP "_FH" ; OPEN "0x12345" ; GETFH }

Standardization

Standardization Status

- IETF Individual I-D by
  - James Lentini
  - Mike Eisler
  - Rahul Iyer
  - Deepak Kenchamanna
  - Anshul Madan
- Extensive feedback and comments on the NFSv4 WG mailing list starting in April, 2009.
- See draft-lentini-nfsv4-server-side-copy

  http://tools.ietf.org/html/draft-lentini-nfsv4-server-side-copy

Thank You

Questions?

Additional Information

Copy Offload Stateids

- Copy Offload Stateids: a new type of stateid to identify asynchronous copies.
- Valid until either:
  - the client or server restart.
  - the client issues a COPY_ABORT operation.
  - the client replies to a CB_COPY operation.
- A copy offload stateid's seqid MUST NOT be 0 (which would indicate the most recent offloaded copy). No use case for this.

NFS Client Support

- When does an NFS client use the server-side copy offload operations?
  - Changes may be needed to the OS’s user/kernel interface.
    - In Linux, reflink(2) (work in progress) looks promising. reflink(2) being proposed by OCFS2 developers for use by Oracle VM, see http://blogs.oracle.com/wim/2009/05/ocfs2_reflink.html
  - Some environments may be ready to take advantage of these operations right away (e.g. a hypervisor).

Additional Features

- Partial file copies
  - Source file offset, destination file offset, and length
- Space reservations
  - Storage spare for the entire destination file is guaranteed
- Guarded copies
  - The copy will fail if the destination file exists
- Metadata copy
  - The destination file will duplicate all required and recommended NFS attributes