

# pNFS Access Permissions Check

IETF 77 NFSv4 WG Meeting

March 23, 2010

**Sorin Faibish** [sfaibish@emc.com](mailto:sfaibish@emc.com)

**David Black** - EMC

**Mike Eisler** - NetApp

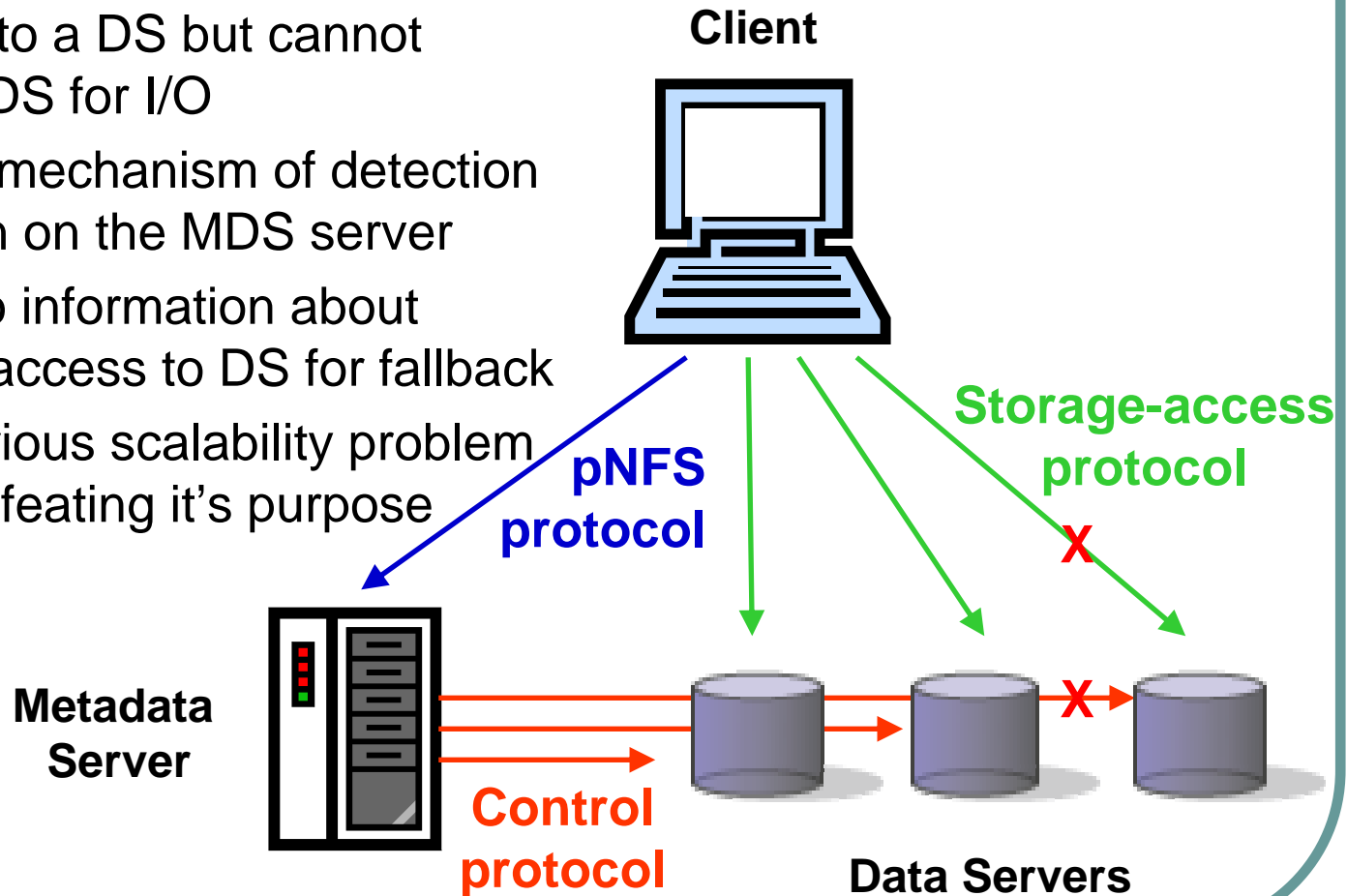
**Jason Glasgow** - Google

# Outline

- Problem Statement
- Protocol Gaps
- Proposed Remedies
- Error Reporting
- Permission Checks
- Implementation Details
- Questions and Discussion

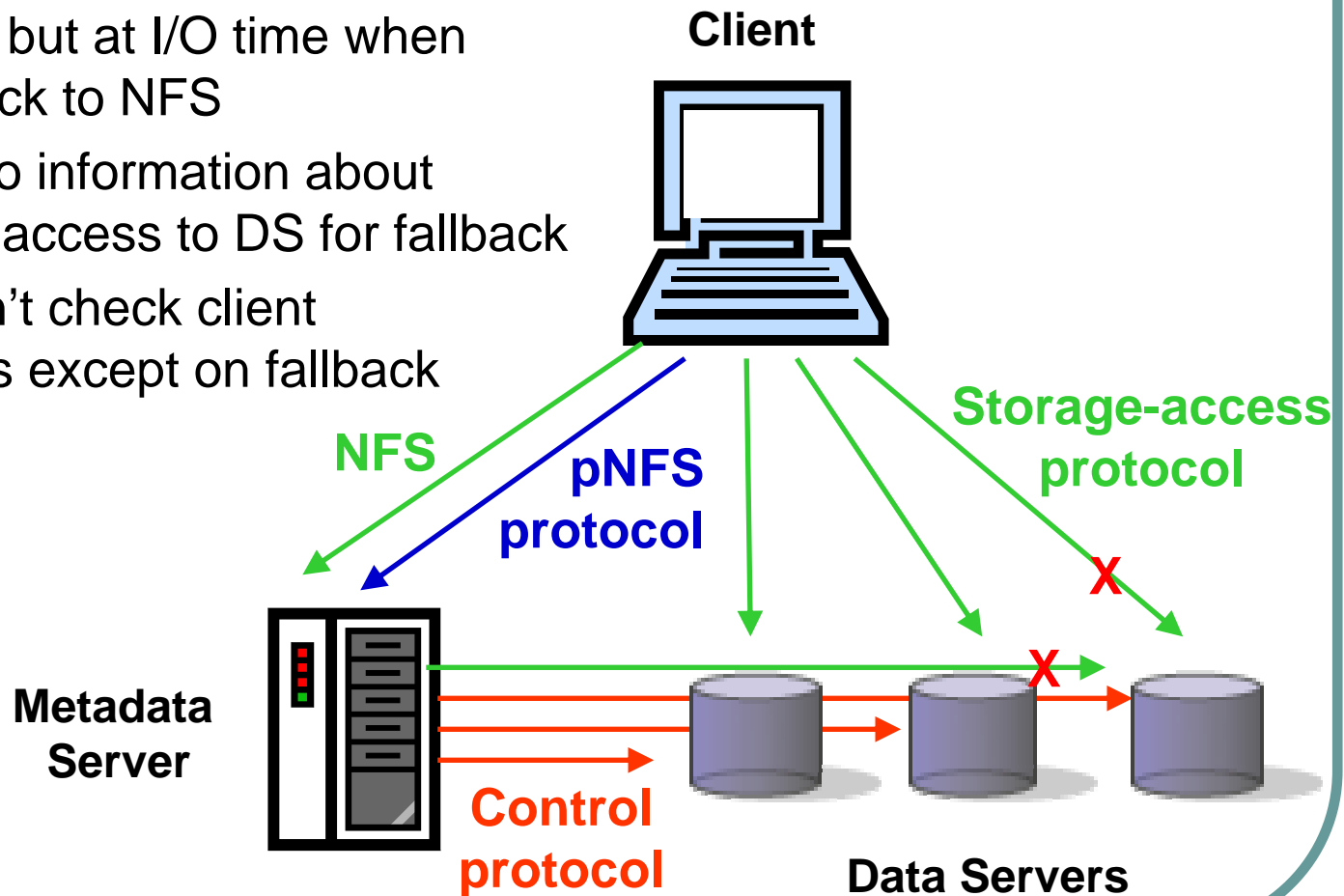
# Problem Statement

- pNFS clients can receive from MDS valid layout to a DS but cannot access the DS for I/O
- There is no mechanism of detection or correction on the MDS server
- MDS has no information about permission access to DS for fallback
- This is a serious scalability problem for pNFS defeating it's purpose



# Problem Statement (cont.)

- Permission denial is not detected at mount time but at I/O time when client fallback to NFS
- MDS has no information about permission access to DS for fallback
- MDS doesn't check client permissions except on fallback detection



# Protocol gaps

1. There is no error report mechanism for client and MDS for permission access issues
2. MDS can deliver valid layout to clients that have no permission to a DS without check
3. There is no correction mechanism of the MDS to recall a layout and remove the DS with issues
4. The permission problem is not reported at mount time (/ is pNFS mounted) and may have a performance penalty during I/O
5. No guarantees that fallback to MDS will succeed
6. pNFS specification does not address the protocol between the MDS and DS

# Proposed Remedies (protocol)

1. A protocol change is needed as both server and client needs modifications. Optimizations don't work: ***draft-faibish-nfsv4-pnfs-access-permissions-check-02***
2. Add access permission error reporting to both client and server using new LAYOUTRETURN command
3. Add a new CB\_LAYOUTRECALL command and LAYOUTRETURN command requiring the client to perform a permission check and return all layouts for DS with permission issues
4. Leave the detection of permission problem condition as a recommendation for the server implementation.
5. On detection the server will remove the DS from the valid DS list configuration or flag it as inaccessible and will recall all the layouts that include that DS and send new layouts excluding the DS to clients

# Error Reporting (I-D)

1. Add client error reporting to LAYOUTRETURN opaque for permission access denial before fallback to NFS – enhance 4.1 same as object layout (I-D 3.1).
2. Same error reporting mechanism will be used in combination with the new CB\_LAYOUTRECALL/LAYOUTRETURN commands after agreement of the WG

# Permission Checks (I-D)

1. Introduce a new LAYOUTRETURN command LAYOUT4\_RET\_REC\_FSID\_NO\_ACCESS for which the client returns all the layouts for the denied device and report a new error NFS4ERR\_DEVICE\_PERM\_DENY (I-D 3.2).
2. Introduce a new CB operation CB\_LAYOUTACCESSCHECKRECALL with which the server will ask the client to check permissions (I-D 3.3)



# Permission Checks

1. Detection of the problem will be left as an implementation (count of fallbacks to NFS for a certain DS/device)
2. The protocol will define the permission checks and the response to permission issues. The new LAYOUTRETURN command is discussed in I-D section 4.1 - 4.3 for 3 cases of permission access denial:
  - Access denied at mount time
  - Access denied at I/O time
  - Access denied to the MDS at I/O time

## Permission Checks (cont.)

3. The new `CB_LAYOUTRECALL` command is used in 4.3 in the case when the I/O fails to the MDS and the MDS will remove the faulty devices from the database and send a new layout to the clients with layouts on that device
4. In section 4.3 the new CB command is sent to clients that didn't yet try to do I/O to the known faulty device. On fallback to NFS the server will send a new layout to client excluding the DS/device with permission denial.

# Permission Checks (discussion)

- Server unilaterally send a CB\_LAYOUTRECALL of all the layouts on the device with the permission issue to all clients that have valid layouts on that DS and send a new layout on next LAYOUTGET command of the client
- Server perform a permission check of himself to access to the DS and log permission error and remove DS/device from configuration
- Server sends a new CB\_LAYOUTRECALL asking for client permission check to a specific DS/device and ask return of the layout on that device on error
- A client can send a LAYOUTRETURN command for the layout on DS to which it has a permission issue and fallback to NFS for that device
- Server will differentiate from a normal fallback or a permission related fallback of many clients.

# Implementation: error report

- For file layout type define the opaque body:

```
struct nfsv4_1_file_layoutreturn4 {  
    deviceid4    lrf_deviceid;  
    nfsstat4     lrf_status;  
};
```

MDS will check size of the opaque lrf\_body if non-zero=error

- For the block layout type the specific structure:

```
struct pnfs_block_layoutreturn4 {  
    deviceid4    lrf_deviceid;  
    nfsstat4     lrf_status;  
};
```

MDS will check size of the opaque lrf\_body if non-zero=error

- For the object layout type opaque already exist see:  
draft-ietf-nfsv4-pnfs-obj-12

Can be added as an extension of object layout to file and block in 4.1

# Implementation: new LAYOUTRETURN

- Add new constant:  
`const LAYOUT4_RET_REC_FSID_NO_ACCESS = 4;`
- Add a new error code:  
`NFS4ERR_DEVICE_PERM_DENY`
- Add new LAYOUTRETURN `layoutreturn_type4`:  
`LAYOUT4_RET_REC_FSID_NO_ACCESS`
- To address the backward compatibility may require a client to do two layout return operations to deal with servers that don't understand the new `layoutreturn_type4`

We propose this implementation for 4.2

# Implementation: new LAYOUTRCALL

- Add new structure:  
layoutaccesscheck\_device4; deviceid4, nfsstat4
- Use same error code:  
NFS4ERR\_DEVICE\_PERM\_DENY
- Add new LAYOUTRECALL:  
CB\_LAYOUTACCESSCHECK
- To address the backward compatibility may require a client to do two layout return of all layouts regardless if it doesn't recognize the CB it will treat it as a normal LAYOUTRECALL CB.

We propose this implementation for 4.2

# Questions & Discussions

- Is the permission check needed or is error reporting enough? Error is not enough.
- Is this issue a protocol change or just an optimization? Protocol change required.
- Are the proposed protocol changes too complex for the pNFS protocol? Reasonably simple as we implemented.
- Are new layout commands needed or should modify existing commands? Yes; both new layoutreturn and layoutrecall.
- Will you support this draft and review it?

# Acknowledgements

Thanks to:

Jason Glasgow defined the implementation details and reviewed the draft

Mike Eisler for brainstorming the alternative implementation decisions

David Black for help with my first draft publication and extremely thorough review