Network Status Monitor (NSM)

(Have you thought about it lately?)

Tom Talpey

Network Appliance, Inc.

tmt@netapp.com

Outline

- A deeply flawed protocol
- An inconsistently implemented protocol
- The problems with correctness
- Some suggestions

Protocol specification

- NSM (statmon, statd) provides notification to NLM clients that they must reclaim their locks after a server failure
- Used by NLMv4/NFSv3 and NLMv3/NFSv2 clients
  - All locks are *advisory*
- X/Open Protocols for Interworking; XNFS Version 3W
  - NSM (*http://www.opengroup.org/onlinepubs/009629799/chap11.htm*)
  - File locking (*http://www.opengroup.org/onlinepubs/009629799/chap9.htm*)
  - PMAP (*http://www.opengroup.org/onlinepubs/009629799/chap6.htm*)
- NFS Illustrated, Brent Callaghan, Addison-Wesley Dec 1999 ISBN: 0-201-32570-5

Here’s the protocol

- You might think there are several important SM_* ops
- Ignore them all, it’s just…
  - NLM_LOCK
  - SM_NOTIFY
- (CERT proscribes any other SM_* originating from non-loopback)

Monitor

- NLM_LOCK { caller_name, lock stuff }
- Carries explicitly:
  - client hostname – *usually undecorated*
- Implicitly:
  - Client IP address
  - Client’s chosen server NLM IP address
  - This IP address may not be the same as the address the client actually mounted

Monitor

- Server remembers each “client of interest”
- Server stores client address information
  - Often, in directory entries and symlinks
  - Or, in a database
- Notifies all such clients after any restart
- Grace period for reclaim, etc.

Notify

- SM_NOTIFY { mon_name, state }
- Sent by server to client statd, which decodes and forwards for client NLM processing
- Carries explicitly:
  - Server hostname – *format important!*
  - Server generation number (ho hum)
- Implicitly:
  - Server IP address
  - Server’s idea of valid client IP address

Notify

- Clients see notify, decode server from mon_host in message
- Match server to interesting mount
- Reclaim locks on that mount

What if Notify doesn’t succeed

- Processes can die (!)
  - SIGLOST on some systems
- Locks are silently lost
  - And taken by others
- Data corruption

What else can go wrong

- Undecorated server name
  - – Client fails to recognize if outside local domain
- Network interruption
  - – Notification failure
- Multipath partial failure
  - – Notification on bad pipe
- DCHP (IP) address changes
  - – Server notifies wrong address

Server notify procedure

- On server restart…
- Start at least your portmapper and NLM services
  - Yes, some servers forget
- For all clients in the notify list
  - Resolve the hostname or IP address(es)
  - Resolve the client statd port
  - Perform the actual SM_NOTIFY
  - If unsuccessful, remember client for later
- Start grace period (after first notify pass)
  - While continuing to attempt earlier failed notifications
  - Include any alternate client addresses, and/or simply retry
- Exit grace period
  - While continuing to attempt earlier failed notifications

The broadcast problem

- Broadcast sounds nice!
- But it won’t work… because we have to resolve the statd port
  - No, we can’t broadcast pmap_callit
    - It’s insecure and won’t pass much more than NULL
  - And besides they may not be on our subnet

The “done” problem

- The SM_NOTIFY procedure is void
  - So a reply means nothing!
  - Any client with a functioning statd will reply.
  - The UDP transport is unreliable.
  - A portmap failure is slightly more useful, in fact.
- Even if reclaims start soon after notify…
  - How do we know when the client is done?

The recall-done problem

- How does the server know client reclaim is done?
- It can't:
  - There's no protocol to indicate done
  - Even if non-reclaim ops arrive, they might be simple retries, etc
  - The client could fail, become disconnected, etc.
- It can take a long time, with many clients

The payload problem

- What to tell the client in the notify
- Well, any bumped, odd# state is ok! ☺
- The server_name must be resolvable at the client to:
  - The address which the NLM client actually used
  - To allow the client to recognize which server to reclaim

I told you it was deeply flawed

- It’s only a notify
- There’s no useful reply or completion
- It’s dependent on implementation practice
- Which isn’t well-understood (or even thought about!)
- And even less well-followed

Client reclaim behavior *examples*

(accuracy not guaranteed)

	Linux 2.4 w/ nfs-utils 1.01	Linux 2.4 w/ nfs-utils 1.07	Linux 2.6.?? w/ nfs-utils 1.07	***Brand X***	***Brand Y***	***Brand Z***
Client format for server name	Dotted quad	Dotted quad	Dotted quad	FQDN	FQDN	FQDN
Response time to NOTIFY if server name not resolved	Long	Long	Long	Short	Long	Short
Reclaim works if server name resolved	**No** – statd uses unprivileged port	Yes	**No** – lockd does not recognize server name	Yes	Yes	Yes
Reclaim works if server sends dotted quad	N/A	Yes	N/A	No	Yes	No
Reclaim requires NLM active before notify	N/A	Yes	N/A	No	Yes	No
Frequency of NLM portmap retry	None	None	None	<10ms	None	2 sec
Behavior of reclaim on lost locks	Reclaim denied, application unaware	N/A	N/A	Reclaim denied, application unaware	Reclaim denied, SIGLOST to application	Reclaim denied, SIGLOST to application

What’s a server to do?

- What to remember
  - IP *and* name (in case IP doesn’t work)
  - Don’t believe the client’s passed-in name, try to fully decorate it and/or reverse-map the IP
- What to send when notifying
  - Server FQDN!
- Be incredibly patient in trying
- Be liberal before (and after) giving up
  - Allow late reclaims, if no conflicts

What’s a server to do?

- Who to send notify to
  - Client IP address is a good start
  - Looking up client FQDN if portmap “fails”
  - Notify any and all aliases
    - Important for not-unlikely path failure
- How hard to try
  - As hard as you can
- Be liberal post-grace

What’s a client to do?

- What to send when locking
  - Send your FQDN as caller_name
  - hostname().domainname() not just hostname()
- Remember the server IP aliases
- Be liberal in recognizing SM_NOTIFY
- It doesn’t hurt to reclaim
  - The server might (re-)grant it
  - If not, the lock loss will be explicit

What’s a client to do?

- What to look for in notifies
  - FQDN server name (best)
    - Possibly resolving to multiple addresses
  - Dotted quad (problematic)
    - Possibly resolving to IP alias of actual mount (not NLM)
    - Check for this!
  - “Other” (just plain difficult)
    - Undecorated server name, etc.

Think about it now…

- Ignorance is not bliss.
- This is a failure-on-failure scenario and not easy to detect.

- You may hear this again from your users.
- After their data is lost or corrupted…

Summary

- More than you thought?
- Think about what addresses your server stores
  - And what server_names it notifies with
- Think about what your client does
  - And how liberal it is
- Test against many combinations/versions
  - Something, somewhere will screw it up.
- Or of course… **Just use NFSv4!!!!**