SELinux MLS on Lustre for enhanced isolation

DataDirect Networks, Inc.

2016/09

Sebastien Buisson sbuisson@ddn.com
Initial requirement for Isolation

- Today, HPC storage is NOT just scratch and user home directory use case is commonplace
  - Same cluster with various use cases
  - Dedicated hardware not efficient
  - Secured data accessible/visible ONLY to people who have credentials and are authorized
What is Lustre Isolation?

▶ **Lustre Isolation:**
  - Provides isolated namespaces from a single filesystem

▶ **Lustre Isolation combines features of:**
  - Containers
    - Each container mounts Lustre as a client
    - ‘root’ user is allowed inside containers
  - Kerberos
    - Each container authenticates with its own credentials
  - Subdirectory mount
    - Each container is allowed to mount only a portion of the namespace
    - Allowance depends on client’s credentials
Lustre: Isolation

1. credentials(A)

2. credA

mount(credA)

Lustre client

Kerberos server

nodemap

Lustre servers

Container A

Container B

/filesetA
What benefits from Lustre Isolation?

▶ Containers avoid static distribution of client nodes
  => dynamic container images instantiation
  • No need to dedicate groups of clients to each population
  • Every client is available for any population
  • Several populations can share same client nodes at the same time

▶ Lustre Isolation enables:
  • Different populations of users on the same file systems
  • Isolation of these different populations of users
    ⇒ Isolation makes Lustre multi-tenant
Taking Lustre Isolation a step further

- Ability to isolate users from the same population
  - Prevent users from accessing others’ data
  - Flexibly adjust access capabilities
  - But still share the same file system root

⇒ Use SELinux MLS to enforce data confidentiality
SELinux support on Lustre client side

► We already have Targeted policy support!
  • Initial landing in 2.8
  • Optimizations available in 2.9+

► Now we need to support MLS on Lustre client
SELinux concepts

▶ Targeted policy

- Targeted policy defines confined and unconfined domains for processes and users.

- It requires to store security information permanently in file extended attribute, to remember security context inherited from the user and process that created the file.
SELinux concepts

- Multi-Level-Security (MLS) policy
  - Adds the concept of security levels in addition to domains

![Diagram showing data flows and security labels]

- Bell-LaPadula model with write equality (as implemented in RHEL)
SELinux concepts

▶ Difference between targeted and MLS policies:
  • Targeted policy protects the OS
  • MLS policy protects the data

▶ From a file system perspective
  • MLS works on clients like Targeted policy
    o Use of `security.selinux` xattr to store security context
      `system_u:object_r:default_t:s2:c17`
SELinux concepts

- Distributed file systems specificity:
  - Really need to make sure data is always accessed by nodes with SELinux MLS policy enforced
    - Otherwise data is not protected

⇒ Make sure SELinux cannot be disabled by root
  - `secure_mode_policyload` SELinux boolean

⇒ Check SELinux status on client
SELinux status on client

- We need to make sure:
  - SELinux is enforced
    - /sys/fs/selinux/enforce
  - The right policy module is loaded
    - /etc/selinux/config
  - The policy is not altered
    - Binary representation of policy at:
      /etc/selinux/<name>/policy/policy.xx
SELinux status on client

Build “SELinux status” info

- With new usermode helper ‘l_getsepol’
  - because need to read and parse files
  - because no SELinux API available in kernel to get this info

- Called from Lustre client code

- “SELinux status” info in the form:
  <1-digit enforcement>:<policy name>:<policy checksum>

- Write “SELinux status” info to
  /proc/fs/lustre/<obd type>/<obd name>/srpc_sepol
SELinux status on client

- SELinux status must be checked:
  - At connect time
  - Every time the client accesses Lustre namespace
    - open
    - create
    - unlink
    - rename
  - Every time the client might access security context
    - getxattr
    - setxattr

⇒ add “SELinux status” info to these requests
SELinux status on client

- **On Lustre server’s side**
  - store “SELinux status” reference information
    - in new ‘sepol’ field of nodemaps
      - can be different for different groups of nodes
  - compare “SELinux status” info received from client with ‘sepol’ stored in nodemap
    - match => process request normally
    - no match => return Permission Denied (EACCES)
MLS on Lustre client

▶ What about performance?

• R&D test-bed
  o Environment
    – 1 client node, 1 server for Lustre MDS, OSS embedded in SFA 14KE
  o Hardware
    – Client node
      » 16 cores
      » 128 GB RAM
      » IB 4X FDR
    – MDS node
      » 48 cores
      » 128 GB RAM
      » RAID 6 10 x 900GB 10K SAS
  o Software
    – CentOS 7.2 (3.10 kernel)
    – Lustre master (2.8.57)
    – MOFED 3.3

• Objective
  o impact over metadata performance
MLS on Lustre client

mdtest - dpp - Create
8 threads

- master - SELinux disabled
- master - SELinux enabled
- MLS support
- MLS support - optimized
MLS on Lustre client

mdtest - dpp - Stat
8 threads

- master - SELinux disabled
- master - SELinux enabled
- MLS support
- MLS support - optimized
MLS on Lustre client

mdtest - dpp - Remove
8 threads

- master - SELinux disabled
- master - SELinux enabled
- MLS support
- MLS support - optimized
Lustre MLS support – code status

- Work in progress
  - Needs further optimizations
  - Code cleanup
- Will push to Community when done

Interested in early evaluation?
Please contact us!
Lustre enhanced Isolation – use case

- **Customer requirement to deliver "science as a service" to:**
  - internal groups
  - external commercial customers

- **Typical workload represented by the cgbox project**
  - encapsulates the core Cancer Genome Project analysis pipeline in a Docker image
  - [https://github.com/cancerit/cgbox](https://github.com/cancerit/cgbox)
Lustre enhanced Isolation – use case

▶ ‘cgp’ population only sees datastore subdirectory

▶ /datastore/input
  • Needs to be readable for every member of the ‘cgp’ population
  ⇒ Set security context’s level of directory to s0

▶ /datastore/output/<id>
  • Accessible read/write for members of the same team
  ⇒ Run container with:
    --security-opt label:level:s1:cxxx
Lustre enhanced isolation

- We are able to enhance isolation feature for Lustre

- By
  - leveraging SELinux MLS policy
  - controlling SELinux status at the Lustre level
Thank You!
Keep in touch with us

Team-jpsales@ddn.com
@dnn_limitless
company/datadirect-networks

102-0081
東京都千代田区四番町6-2
東急番町ビル 8F
TEL:03-3261-9101
FAX:03-3261-9140
Thank You!
Keep in touch with us

Team-jpsales@ddn.com
@ddn_limitless
company/datadirect-networks

102-0081
東京都千代田区四番町6-2
東急番町ビル 8F

TEL:03-3261-9101
FAX:03-3261-9140
MLS on Lustre client

mdtest - dpp - Create

- SELinux disabled
- SELinux enabled
- MLS support
- MLS support - optimized

Number of threads: 1, 2, 4, 8, 16

Op/s: 0, 5000, 10000, 15000, 20000, 25000, 30000