Isolation and Multi Level Security on Lustre

DataDirect Networks Japan, Inc.

Shuichi Ihara (sihara@ddn.com)

2016/10/18

Multi Level Security

http://www.acq.osd.mil/ecp/Articles/FP_2016FEB.html

Security is important for HPC Storage

- **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

DDN contributions on Lustre Security and Resource Management

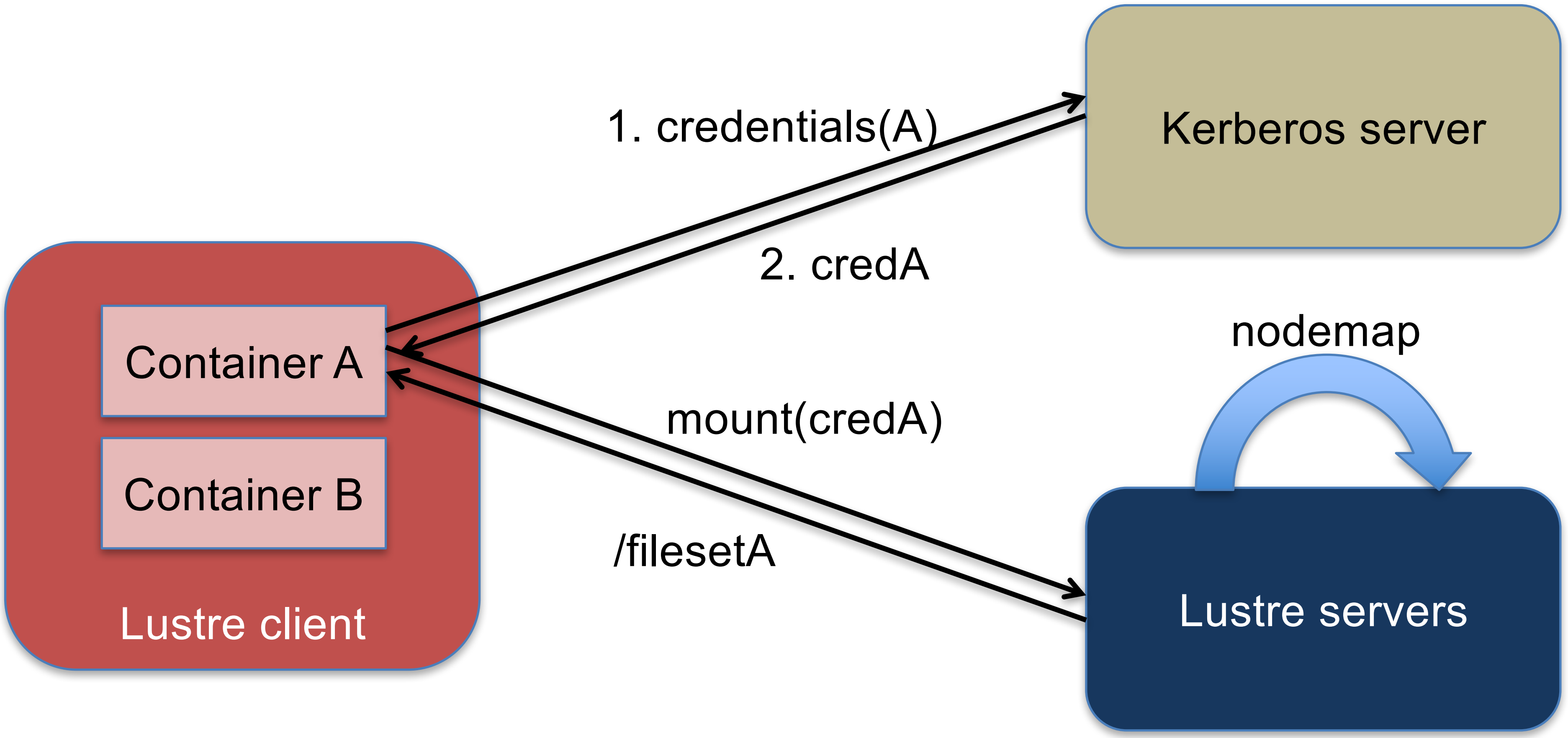
- **Integration with Kerberos Authentication (Lustre-2.7)**
- **Subdirectory Mount (Lustre-2.9)**
- **Quota for Subdirectory (Lustre-2.10)**
- **Docker Integration**
  - With Subdirectory mount (Lustre-2.9)
  - With Kerberos Authentication (Lustre-2.10)
  - QoS for Docker container(Lustre-2.10+)
- **MLS (Multi Level Security) for Lustre**

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



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
- **Container doesn't know other container's security level**
  - No associated with each container's security level

⇒ **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

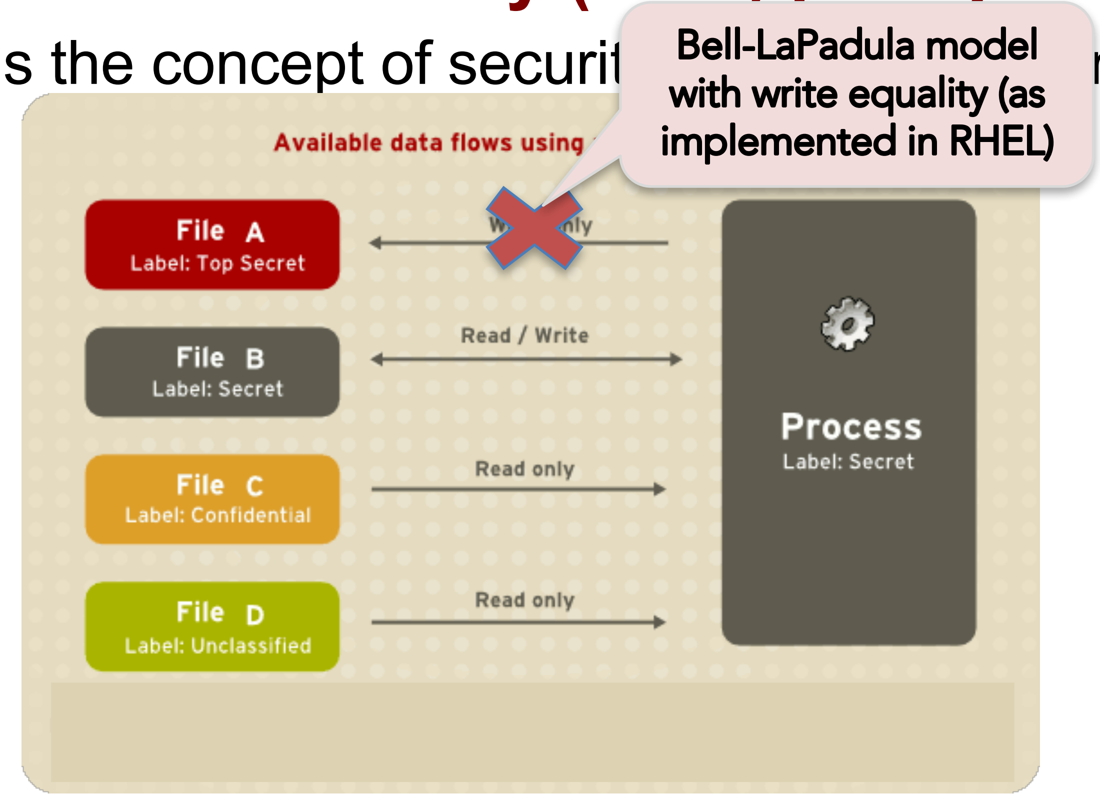
▶ Targeted policy

- Targeted policy defines confined and unconfined domains for processes and users.
- It requiers 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 securit n to domains



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
    - 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

- ▶ **Enable SELinux!**
- ▶ **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

selinux 無効
selinux 確認
selinux とは
selinux 無効 centos7
selinux centos7
selinux 無効 centos6
selinux ログ
selinux centos
selinux android
selinux ubuntu

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

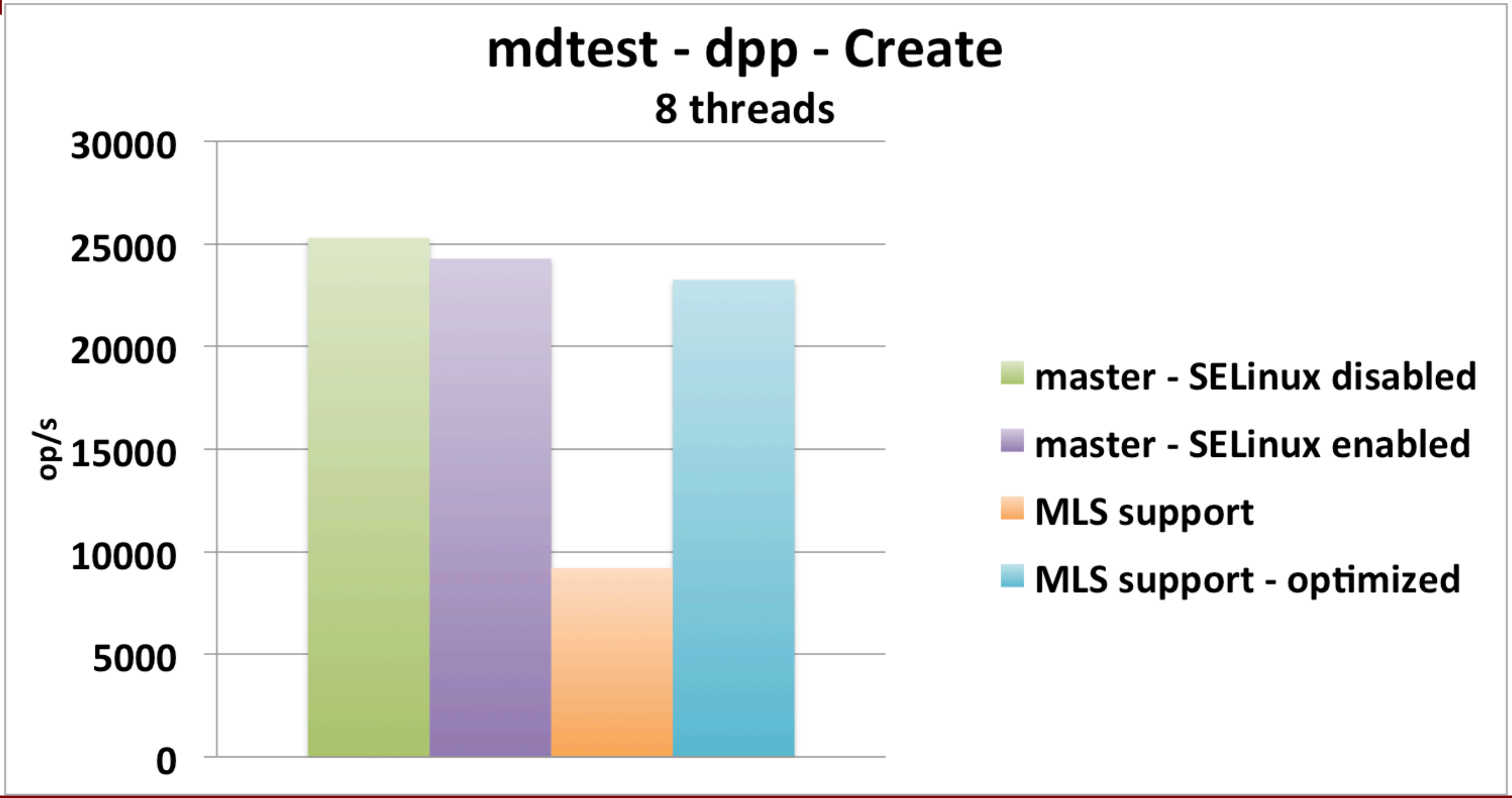
- store “SELinux status” reference information
  - in new ‘sepol’ field of nodemaps
    - can be different for different groups of nodes
- compare “SELinux status” info received from client with ‘sepol’ stored in nodemap
  - match => process request normally
  - no match => return Permission Denied (EACCES)

MLS on Lustre client

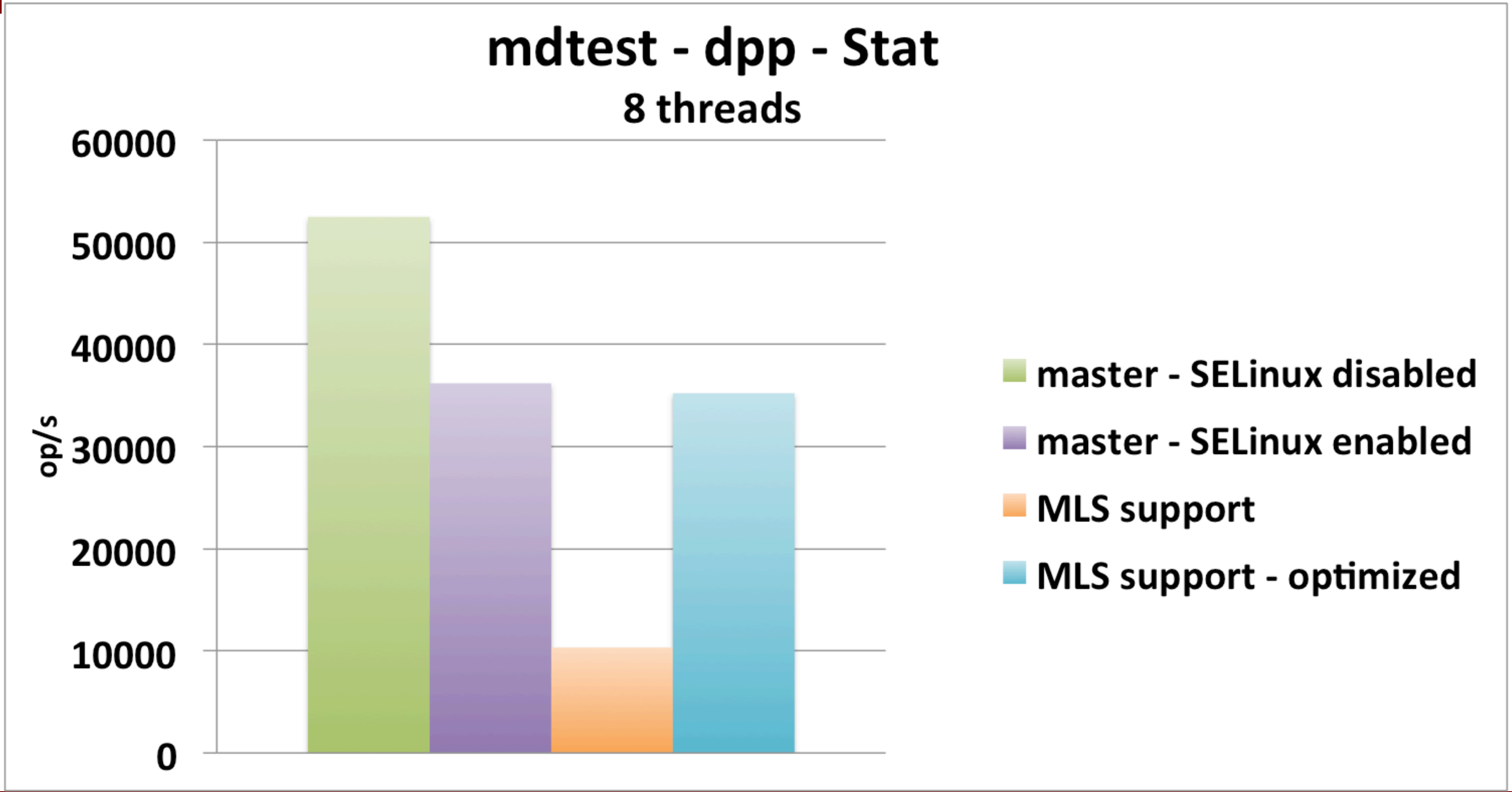
▶ What about performance?

- R&D test-bed
  - Environment
    - 1 client node, 1 server for Lustre MDS, OSS embedded in SFA 14KE
  - 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
  - Software
    - CentOS 7.2 (3.10 kernel)
    - Lustre master (2.8.57)
    - MOFED 3.3
- Objective
  - impact over metadata performance

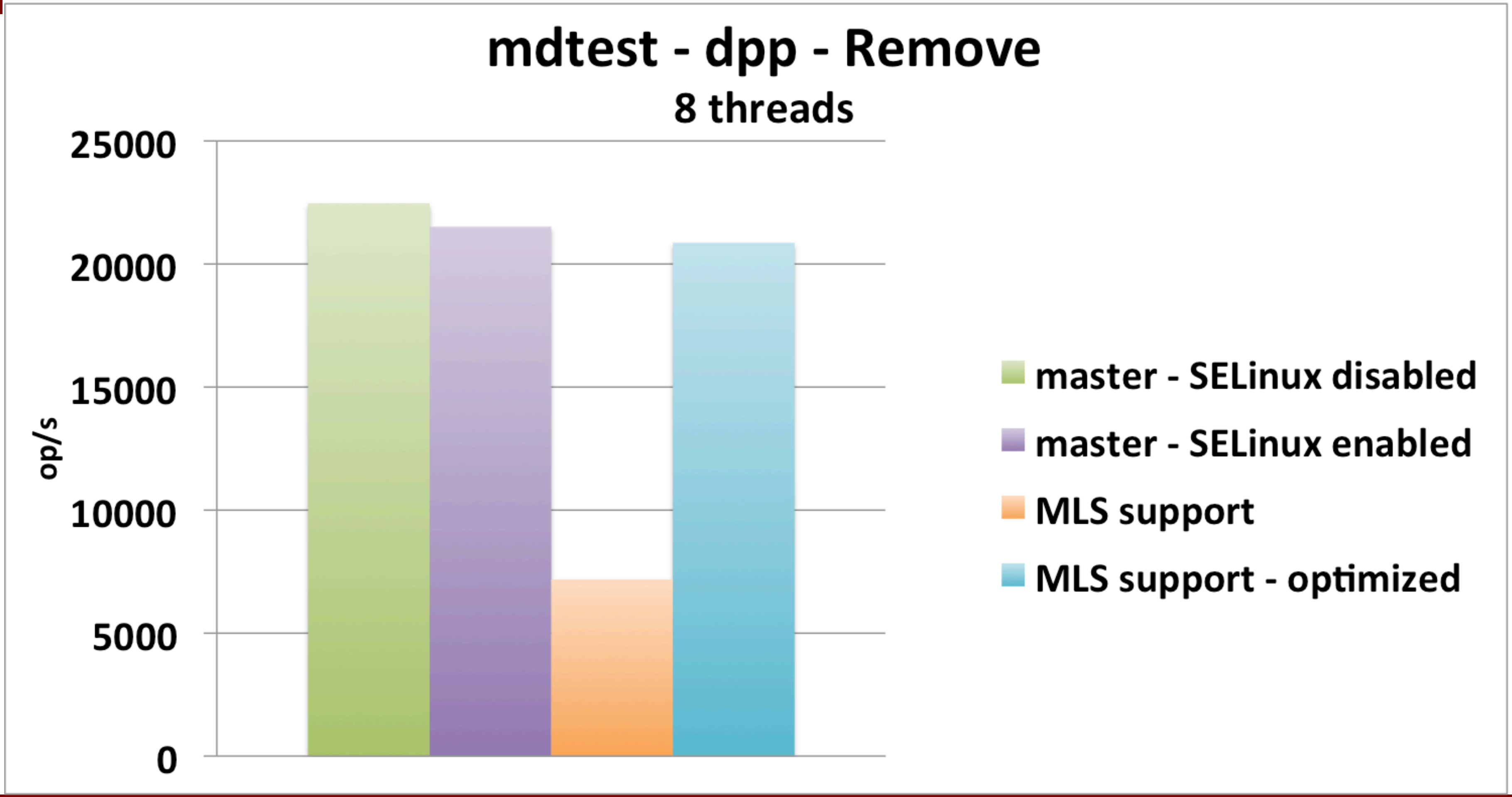
MLS on Lustre client



MLS on Lustre client



MLS on Lustre client



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 *cgpbox* project**
  - encapsulates the core Cancer Genome Project analysis pipeline in a Docker image
  - https://github.com/cancerit/cgpbox

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
```

Summary

- **We are able to enhance isolation feature for Lustre**
  - leveraging SELinux MLS policy
  - controlling SELinux status at the Lustre level
- **MLS feature works in conjunctions with other Lustre security features**
  - Kerberos authentication
  - Sub directory mount
- **Enabling SElinux on Lustre Sever**
  - Strict network security level checking on server and client

Thank You!

Keep in touch with us

Team-jpsales@ddn.com

@ddn_limitless

company/datadirect-networks

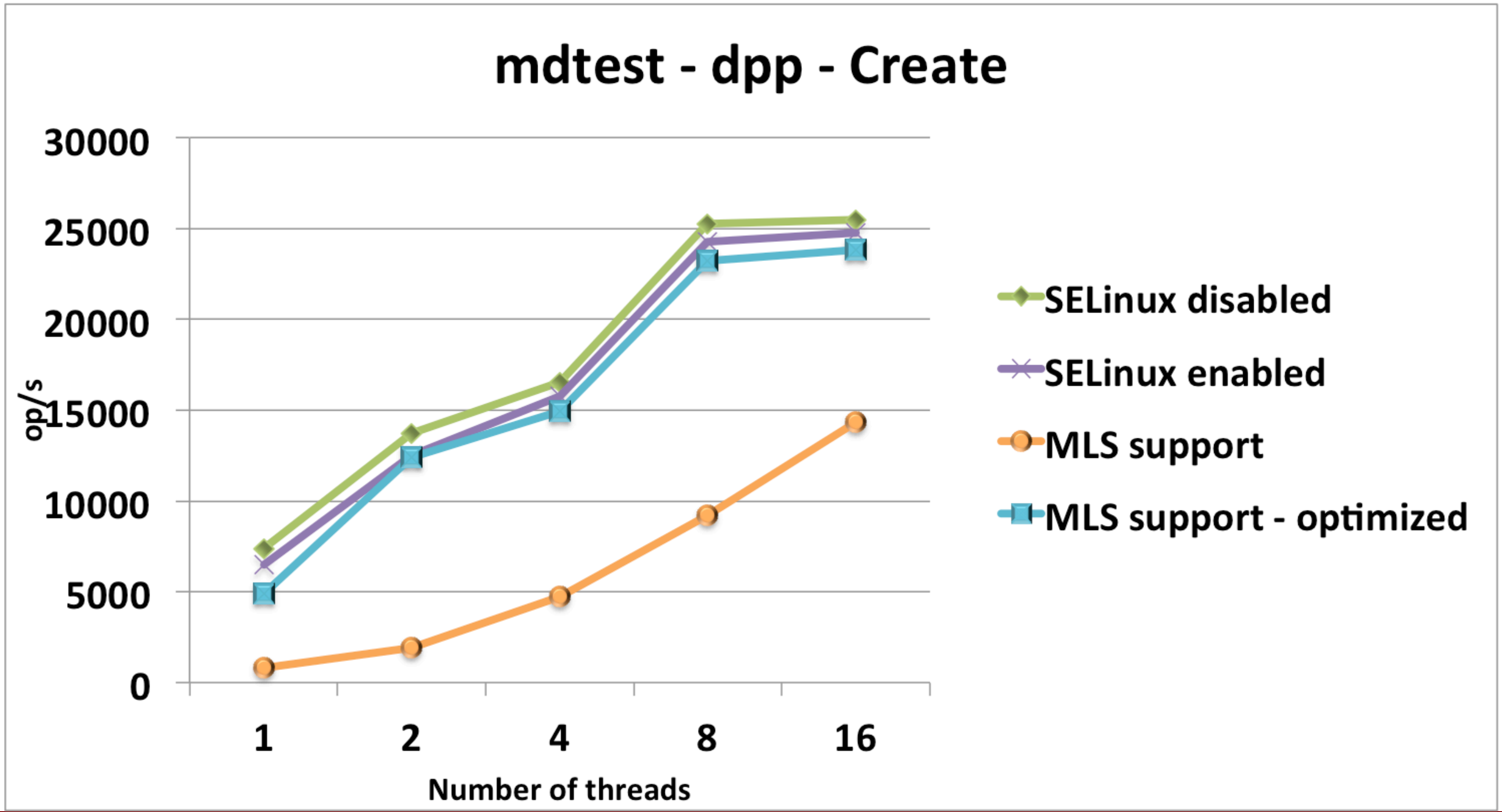
102-0081
東京都千代田区四番町6-2
東急番町ビル 8F

TEL:03-3261-9101
FAX：03-3261-9140

DDN STORAGE

ddn.com

MLS on Lustre client


mdtest - dpp - Create
op/s
Number of threads
SELinux disabled
SELinux enabled
MLS support
MLS support - optimized