

Introdução E/S Paralela no SDUMONT Lustre

Escola Supercomputador SDUMONT
Programa de Verão 2021

André Ramos Carneiro (andrerc@lncc.br)
Bruno Alves Fagundes (brunoaf@lncc.br)

Roteiro:

- Referências
- O Sistema de Arquivos Lustre
- Arquitetura
- I/O + Stripe
- Lustre no SDumont
- Utilitário lfs
 - Configuração de Stripe
 - Comandos Básicos
- Melhores Práticas e Recomendações

Lustre

Referências:

- Conteúdo do minicurso:

- http://www.lncc.br/~andrerc/intro_ES_SD-2021.tar.gz
- http://www.lncc.br/~andrerc/intro_ES_SD-2021.zip

- Mais sobre o Lustre:

- <https://www.lustre.org/>
- https://wiki.lustre.org/Main_Page

- CENAPAD-RJ: <http://www.cenapad-rj.lncc.br/tutoriais/materiais-hpc>

O Sistema de Arquivos Lustre

- Parallel Filesystem OpenSource
- Projetado para ser escalável e alto *throughput*
 - Arquivos grandes
 - HPC “tradicional”
- Capaz de lidar com um grande volume de dados e um enorme número de arquivos
- Alta disponibilidade
- Coerência de dados e metadados
- Armazenamento Baseado em Objetos
- Filesystem ext4 (modificado) ou zfs
- Conformidade com o Padrão POSIX

O Sistema de Arquivos Lustre

- Rede heterogênea de alto desempenho
 - InfiniBand, TCP (10GE e IpoIB), Myricom, Cray Seastar, Omni-Path
- Lista de Controle de Acesso (**ACL**), atributos estendidos (**xattr**)
- *Locks* de arquivo e metadados, na granularidade de byte
- *Striping*
- MPI I/O – ADIO (Abstract-Device Interface for I/O)
- Crescimento Escalável da Capacidade
 - → Mais Servidores = Mais Armazenamento + Largura de Banda
- NFS e CIFS
- disaster/recovery
- Monitorador de desempenho

Object Based Storage

Object Based vs Block Based

BLOCK STORAGE

BLOCK # ↓ ↑ BLOCK #



vs.

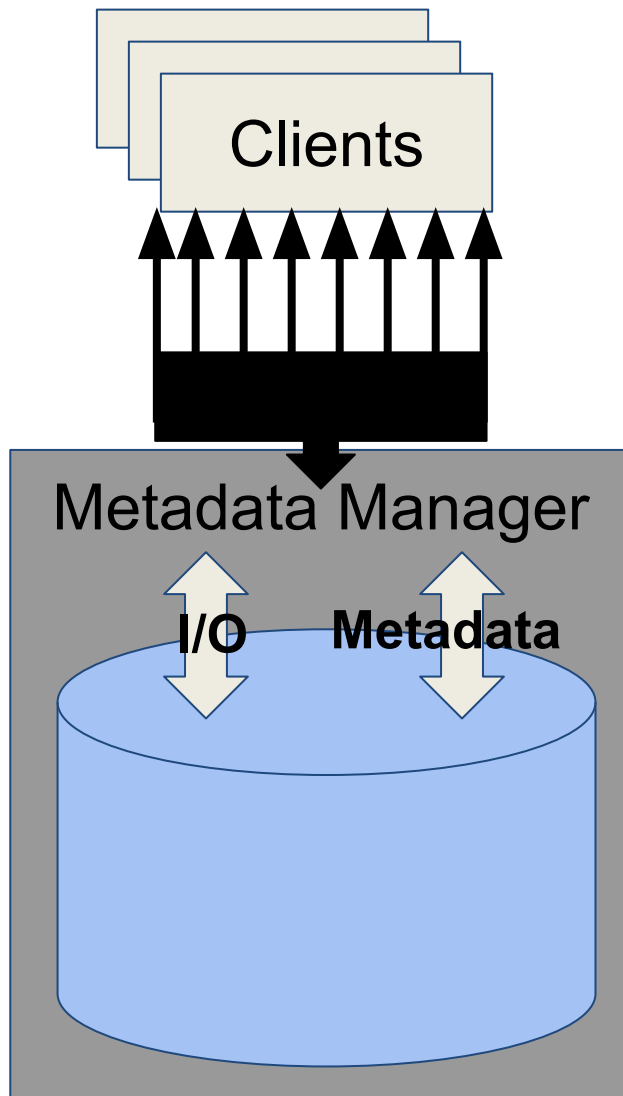
OBJECT STORAGE

DATA ↓ ↑ METADATA/
OBJECT ID

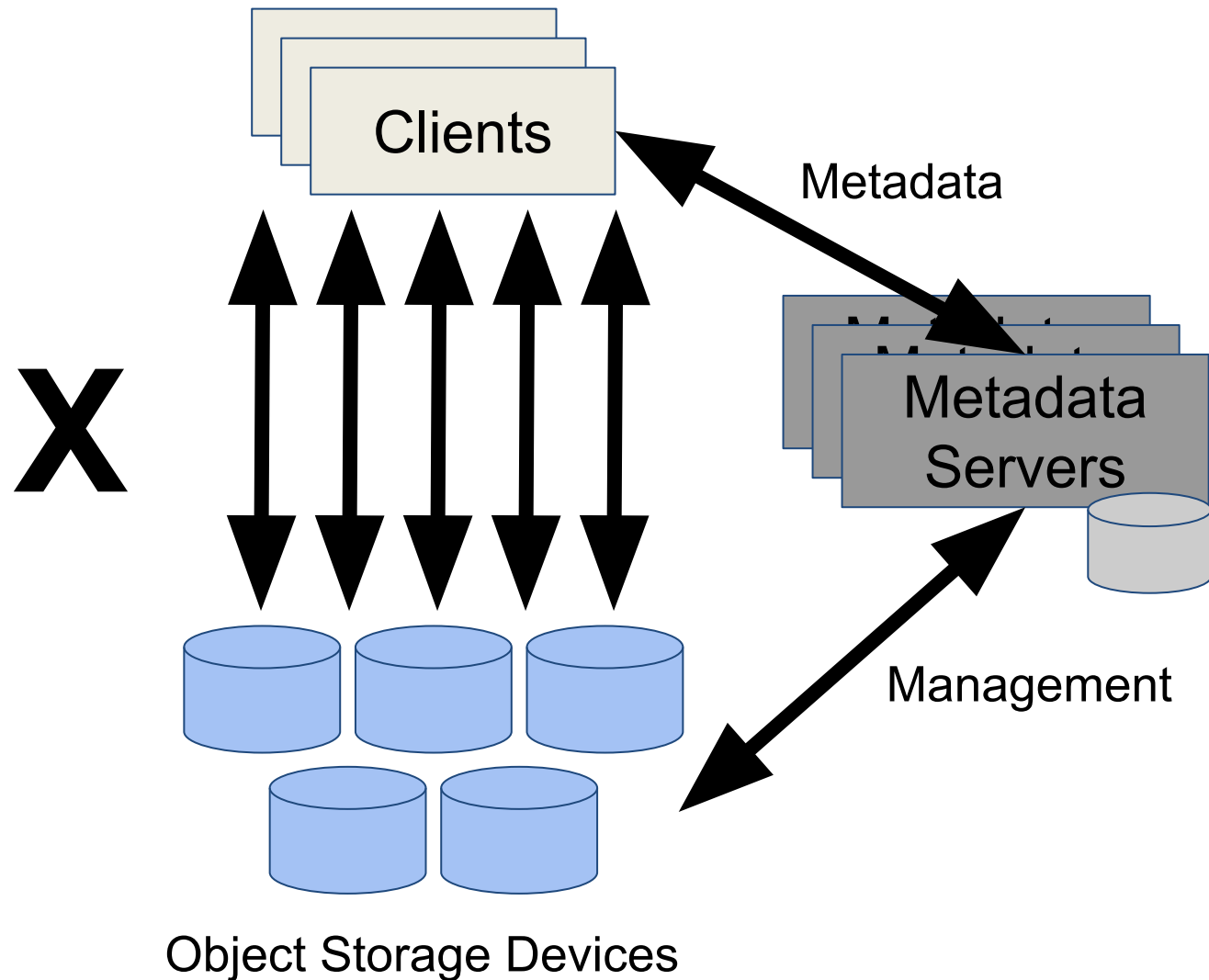


Object Based Storage

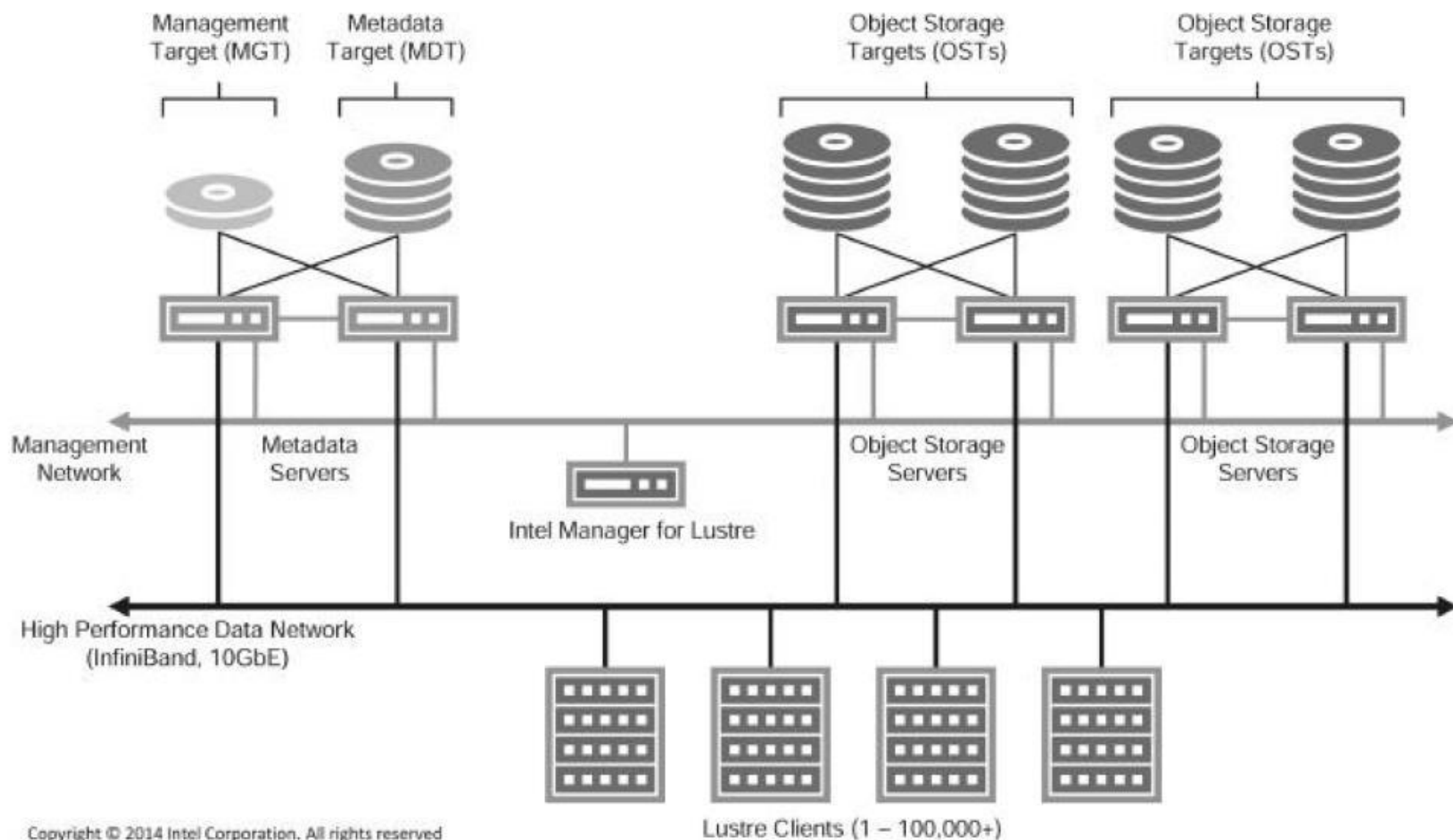
Sequential I/O



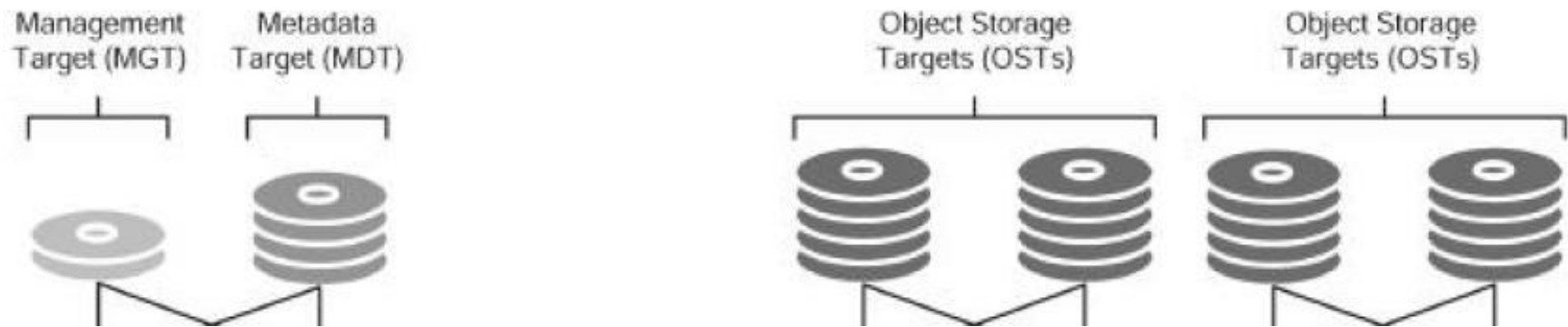
Parallel I/O



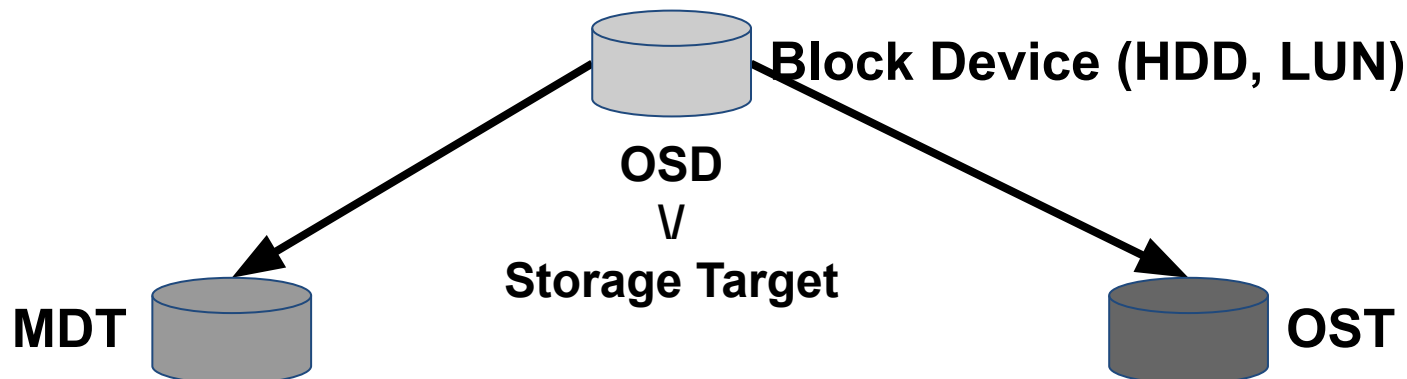
Arquitetura Lustre



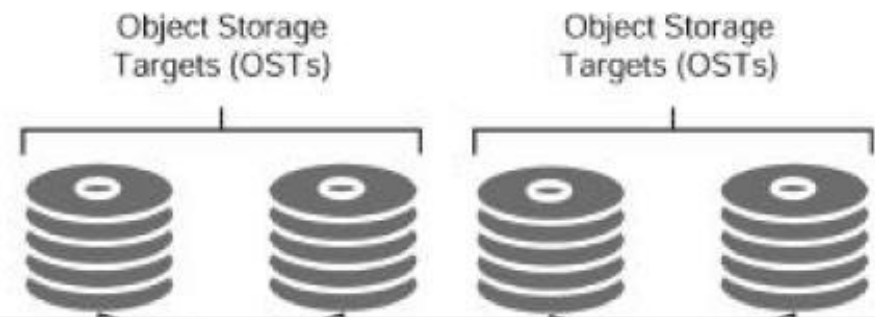
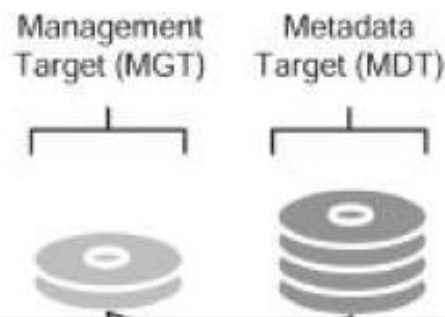
Arquitetura Lustre - Objects & Targets



- Objetos Lustre:
 - (1) objetos de **dados**: arrays de bytes para armazenar os dados dos arquivos
 - (2) objetos de **índice**: contêineres que armazenam pares de chave-valor – implementando a estrutura de diretórios POSIX
- Objetos implementados pelo Lustre OSD (object storage device)
 - **ldiskfs** (ext4) ou **zfs**



Arquitetura Lustre - Targets & LDLM



MDT - Metadata Target

- Metadados (nomes, permissões de layout de arquivo)
- Operações Aleatórios / Baixa latência
- DOM: Data-on-MDT (small file)

OST - Object Storage Target

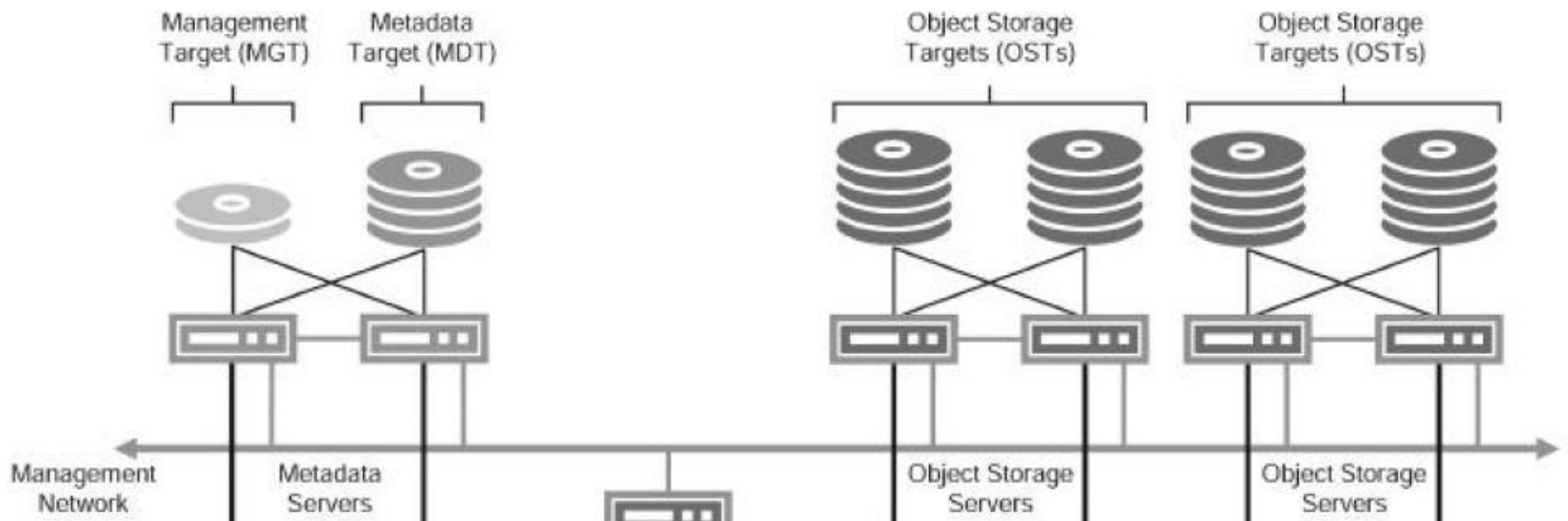
- Dados dos arquivos
- Alta Capacidade + Throughput

LDLM (Lustre Distributed Lock Manager)

- Serviço de *lock* fornecido pelos storage targets (MDT ou OST)
- *Lock* e arquivo, objeto e *byte range*
- Serializa operações conflitantes
- Garante coerência do cache distribuído
- Escala com o crescimento do *filesystem*

- *Locks* de Leitura
- *Locks* de Escrita
- *Locks* exclusivos
- *Locks* de Extensão
- *Locks* IBITs

Arquitetura Lustre - Servers



MDS - Metadata Server

- Exportam os MDTs.
- Operações no namespace (busca, criação e manipulação de atributos)
- Controla seleção e designação do OST
- Balanceamento da Utilização
- **DNE** (Distributed Namespace)
 - MDT0 = /data
 - MDT1 = /data/project_alpha

OSS - Object Storage Server

- Exportam os OSTs.
- Fornecem serviço de I/O de arquivo
- Tratamento de requisições de rede para um ou mais OSTs locais.

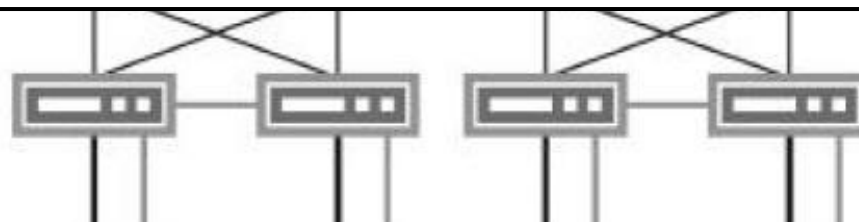
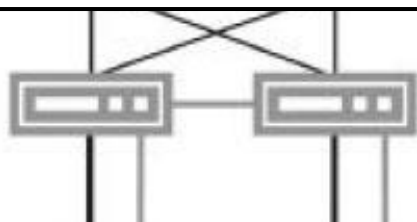
Arquitetura Lustre - Clients

MDS

- MGC e OSC (1 / OST)

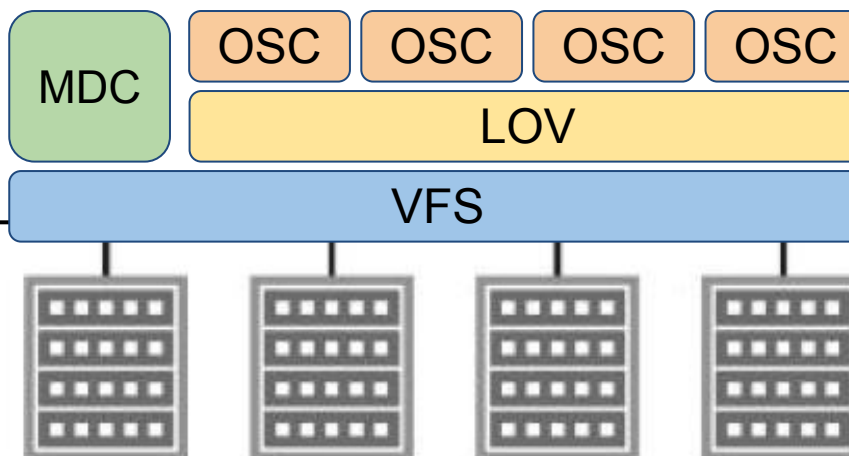
OSS

- MGC



Compute Nodes

- MGC, MDC e OSC (1 / OST)
- VFS/LOV

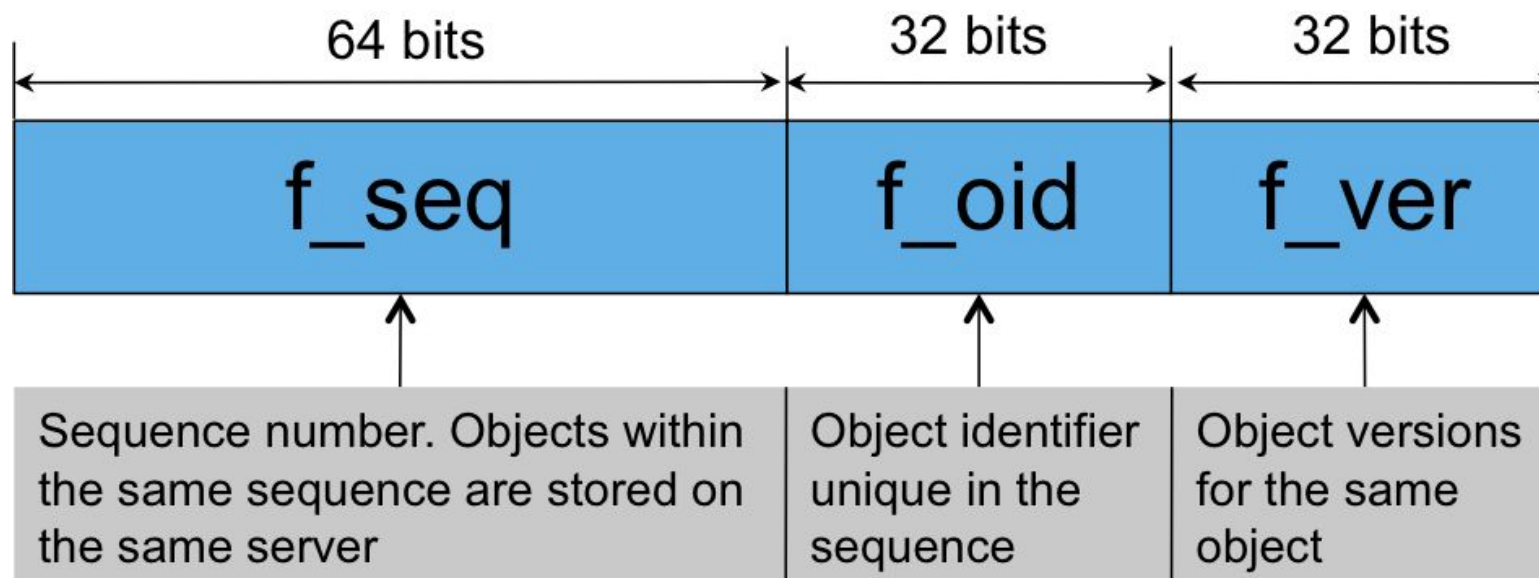


High Performance Data Network
(InfiniBand, 10GbE)

Lustre Clients (1 – 100,000+)

- Comunicação entre clientes e servidores através do **LNET** (**L**ustre **N**etworking)
 - **LND** (**L**ustre **N**etwork **D**river): Abstração das redes físicas (IB / TCP/IP)
 - Troca de mensagens (two-sided)
 - RDMA (one-sided)
 - Zero-copy
 - Movimentação dos dados
 - Roteamento entre redes
 - Lustre RPC (ptlrpc)

Armazenamento e Operações de E/S

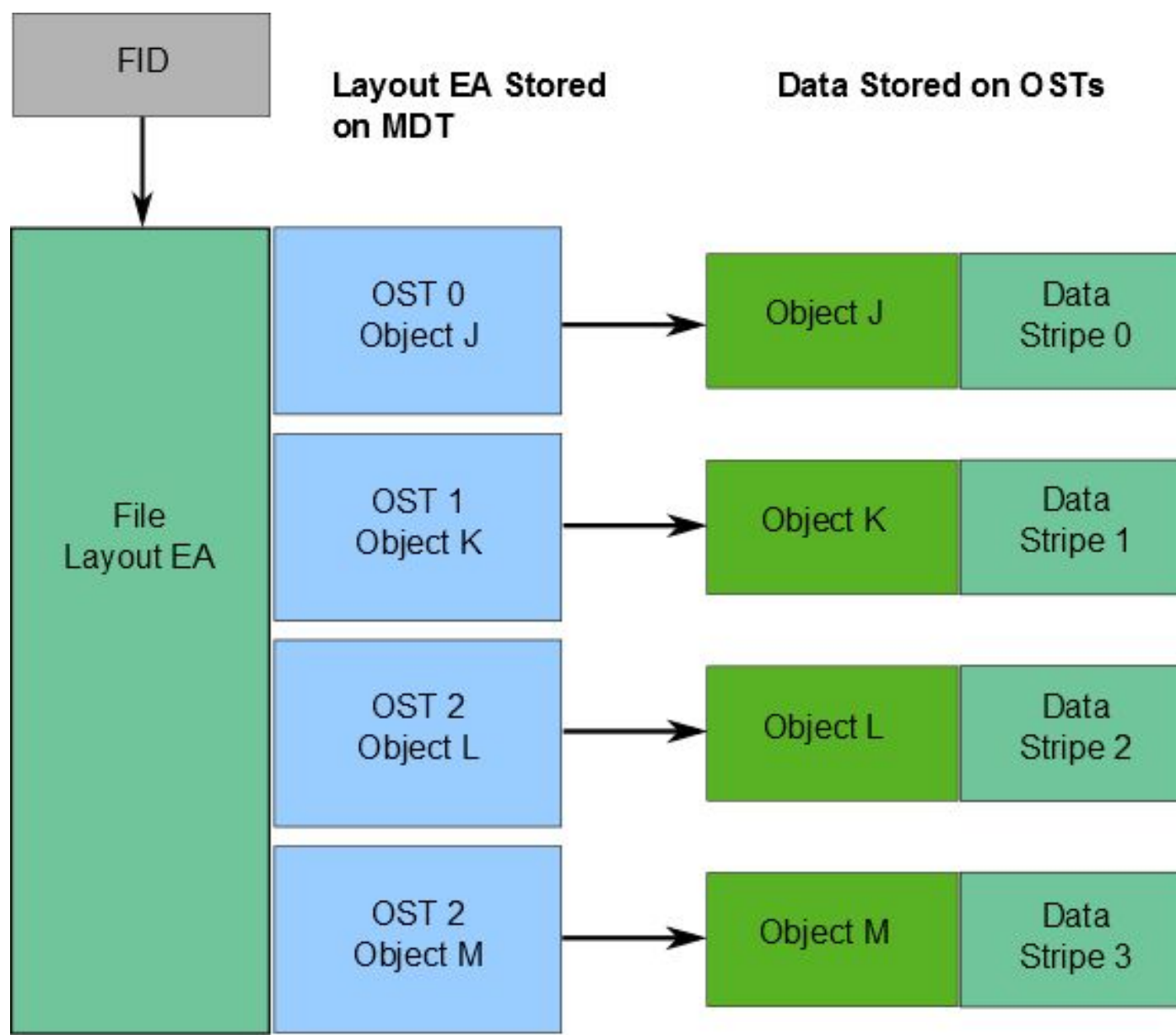


- FID – File Identifiers: Introduzido na versão 2.0. Substitui os números de inodes para identificar arquivos ou objetos de forma única
 - **f_seq**: 64 bits para localizar o storage target (MDT)
 - **f_oid**: Um ID de objeto (OID – Object ID) de 32 bits
 - **f_ver**: O número da versão em 32 bits (registra as alterações realizadas)

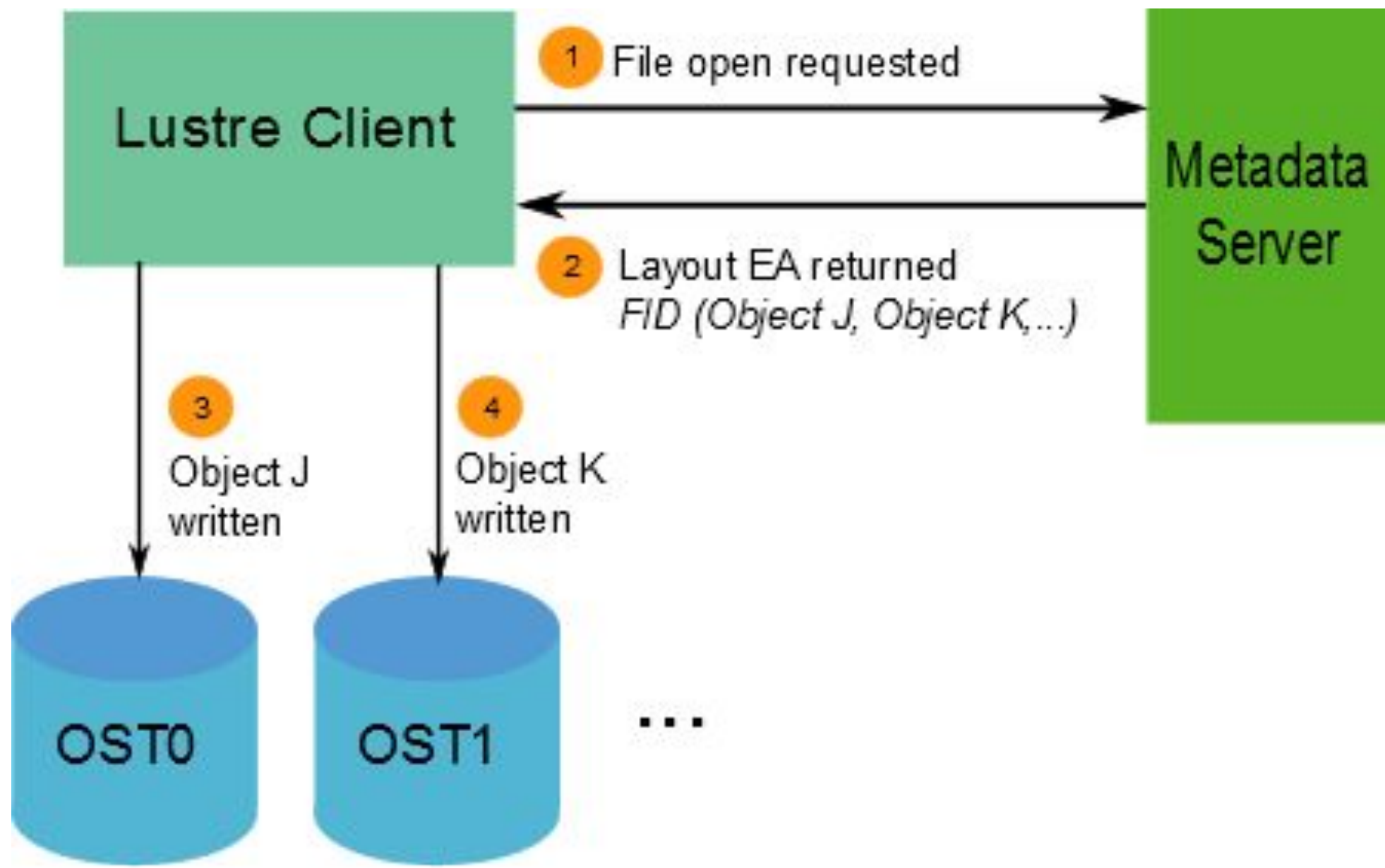
Armazenamento e I/O no Lustre FS

- **Layout EA:** Atributo estendido que armazena informação sobre onde os dados do arquivo estão localizados no(s) OST(s)
 - Armazenado no Objeto MDT, identificado pelo FID do arquivo.
- Se o arquivo é um arquivo regular, o objeto MDT possui um relacionamento que aponta de 1-para-N objeto(s) OST
- Se o objeto MDT apontar para mais do que um objeto → arquivo foi dividido em stripes, sendo cada objeto (stripe) armazenado em um OST diferente

Armazenamento e I/O no Lustre FS



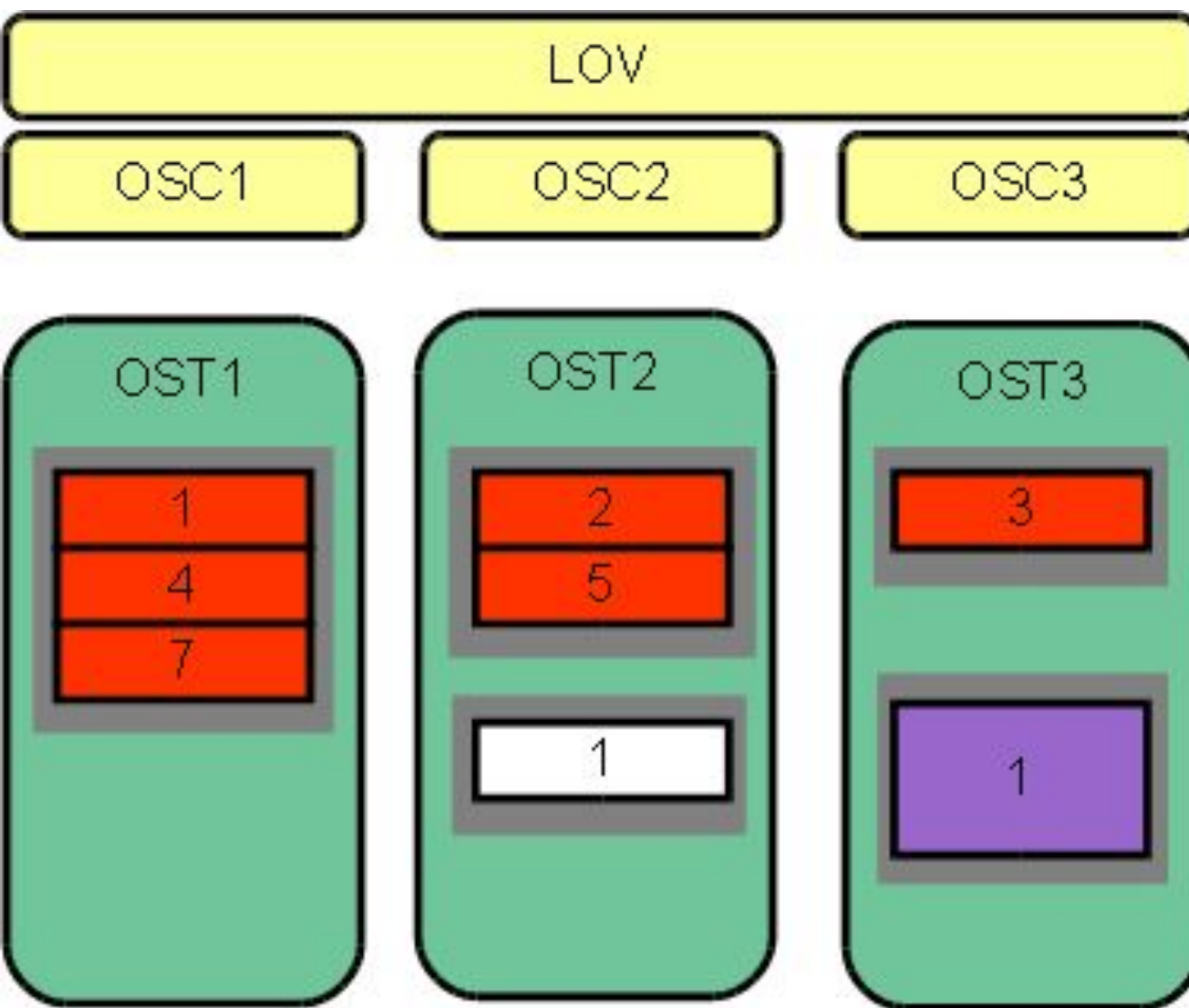
Armazenamento e I/O no Lustre FS



Armazenamento e Striping

- Dividir arquivos em stripes (“faixas”), utilizando um esquema *RAID 0*, sendo cada uma armazenada em objetos/OSTs diferentes
- Utilizado para melhorar o desempenho
 - Largura de banda agregada para um único arquivo excede a largura de banda de um único OST/OSS
 - OST não possui espaço suficiente para armazenar o arquivo inteiro
- `stripe_count`: determina em quantos pedaços (chunks)/objetos de dados um arquivo será dividido
- `stripe_size`: determina a quantidade de dados que será armazenada em cada pedaço/objeto

Armazenamento e I/O no Lustre FS Striping



Unified File System Namespace
logical object volume

File A data



File B data



File C data



Object



Armazenamento e I/O no Lustre FS

Striping

- O tamanho máximo do arquivo não é limitado pelo tamanho de um único target
- O valor **padrão** para o `stripe_count` é **1**
 - Máximo é 2000 (limitado pelo número de OSTs).
- O valor padrão para o `stripe_size` é **1 MiB**
 - Máximo é **16 TiB** no *ldiskfs* e **256 PiB** no *ZFS*
- O tamanho máximo do arquivo: **32 PiB** no *ldiskfs* e de **8 EiB** no *ZFS*
- A largura de banda de I/O para acessar um único arquivo é a largura de banda agregada de I/O para os objetos do arquivo

Lustre do SDUMONT

- Solução ClusterStor 9000 da Xyratex (Seagate -> Cray -> HPE) - V 3.2
- 2 nós de Administração
- 2 nós **MGS** / **MDS** (Ativo/Ativo Alternados)
- 10 nós **OSS** (5 pares Ativo/Ativo)
- Utilizando interconexão Infiniband
 - [172.20.250.103@o2ib](#), [172.20.250.104@o2ib](#):/cstor → /scratch
- Tamanho total do volume: **1.7 PB**
- **Throughput* máximo: 45 GiB/s
- Versão do Lustre: **2.12**

lfs – Configurar o Striping

- `lfs setstripe -S stripe_size -c stripe_count filename|dirname`
 - Tamanho em **bytes**
 - Sufixo de unidade: **K**ibibytes, **M**ebibytes, **G**ibibytes
- Exemplo – configura o *layout* para um diretório. Todos os arquivos que forem criados dentro do diretório possuirão o mesmo layout:
 - Arquivos existentes no diretório não terão seu *layout* alterado!

```
$ cd $SCRATCH
$ mkdir full_stripe
$ lfs setstripe -S 4M -c -1 full_stripe
##
$ mkdir single_stripe
$ lfs setstripe -S 1M -c 1 single_stripe
```

lfs – Configurar o Striping

- **lfs** *getstripe* filename|dirname
- Exemplo – lista informações do diretório (único stripe):

```
$ cd $SCRATCH
$ lfs getstripe -d single_stripe
single_stripe
stripe_count:    1 stripe_size:    1048576 stripe_offset:    -1
```

lfs – Configurar o Striping

- **lfs** *getstripe* filename|dirname
- Exemplo – cria um arquivo e lista suas informações (único stripe):

```
$ dd if=/dev/zero of=single_stripe/teste_single_stripe bs=1M count=1k  
$ lfs getstripe single_stripe/teste_single_stripe
```

```
single_stripe/teste_single_stripe
```

```
lmm_stripe_count:    1
```

```
lmm_stripe_size:     1048576
```

```
lmm_layout_gen:      0
```

```
lmm_stripe_offset:   3
```

obdidx	objid	objid	group
3	1167913	0x11d229	0

lfs – Configurar o Striping

- **lfs** *getstripe* filename|dirname
- Exemplo – lista informações do diretório (full stripe):

```
$ lfs getstripe -d full_stripe
full_stripe
stripe_count:    -1 stripe_size:    4194304 stripe_offset:    -1
```

lfs – Configurar o Striping

- Exemplo – cria um arquivo e lista suas informações (full stripe):

```
$ dd if=/dev/zero of=full_stripe/teste_full_stripe bs=4M count=256
$ lfs getstripe full_stripe/teste_full_stripe
full_stripe/teste_full_stripe
lmm_stripe_count:    10
lmm_stripe_size:    4194304
lmm_layout_gen:      0
lmm_stripe_offset:   8
```

obdidx	objid	objid	group
8	1167882	0x11d20a	0
2	1167720	0x11d168	0
9	1166249	0x11cba9	0
1	1167274	0x11cfaa	0
7	1165803	0x11c9eb	0
5	1152459	0x1195cb	0
3	1167915	0x11d22b	0
6	1167210	0x11cf6a	0
4	1166409	0x11cc49	0
0	1167722	0x11d16a	0

lfs – Alterar o layout do Striping

- **lfs migrate** -S stripe_size -c stripe_count filename|dirname
- Exemplo - Alterar o layout do arquivo para 4 stripe e 4MB

```
$ lfs getstripe single_stripe/teste_single_stripe  
single_stripe/teste_single_stripe
```

```
lmm_stripe_count: 1  
lmm_stripe_size: 1048576
```

```
...
```

obdidx	objid	objid	group
1	924972680	0x3721f688	0

```
$ lfs migrate -S 4M -c 4 single_stripe/teste_single_stripe
```

```
$ lfs getstripe single_stripe/teste_single_stripe
```

```
lmm_stripe_count: 4  
lmm_stripe_size: 4194304
```

```
...
```

obdidx	objid	objid	group
1	924973004	0x3721f7cc	0
7	928675919	0x375a784f	0
5	911874379	0x365a194b	0
3	931125730	0x377fd9e2	0

lfs – Comandos Básicos

- `lfs df`: exibe a utilização do espaço no Lustre

```
$ lfs df -h
```

UUID	bytes	Used	Available	Use%	Mounted on
cstor-MDT0000_UUID	2.8T	479.9G	2.3T	18%	/scratch[MDT:0]
cstor-OST0000_UUID	168.0T	55.5T	110.8T	34%	/scratch[OST:0]
cstor-OST0001_UUID	168.0T	50.4T	115.9T	31%	/scratch[OST:1]
...					
cstor-OST0008_UUID	168.0T	55.5T	110.9T	34%	/scratch[OST:8]
cstor-OST0009_UUID	168.0T	50.5T	115.8T	31%	/scratch[OST:9]
filesystem_summary:	1.6P	522.4T	1.1P	32%	/scratch

lfs – Comandos Básicos

- `lfs find`: Realiza busca no Lustre

```
$ lfs find . -atime -1 -name '*.f90'
```

- `lfs quota -u|-g <filesystem>`: Obtém a quota → Utilização do espaço no Lustre

```
$ lfs quota -g PROJETO /scratch/PROJETO
```

- `lfs cp`: Realiza cópias de arquivos e diretórios no Lustre **(descontinuado na 2.11)**
- `lfs ls`: Lista informações sobre arquivos e diretórios **(descontinuado na 2.11)**
- `lfs path2fid`: Exibe o FID para um determinado arquivo ou diretório:

```
$ lfs path2fid full_stripe/teste_full_stripe single_stripe/teste_single_stripe  
full_stripe/teste_full_stripe: [0x240004762:0x8def:0x0]  
single_stripe/teste_single_stripe: [0x240004762:0x8dec:0x0]
```

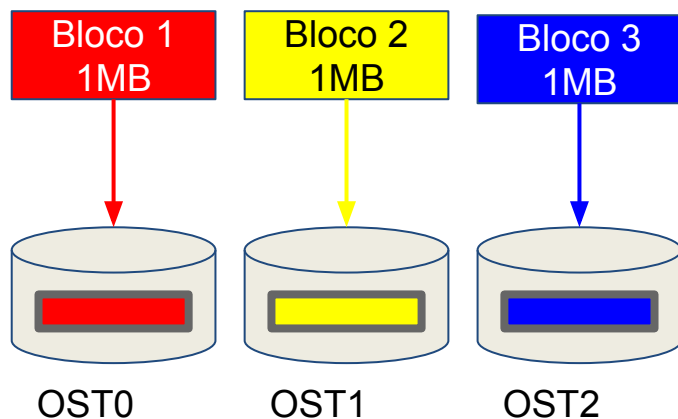
- Para uma lista completa dos comandos: `lfs help`
- Para informações específicas de um comando: `lfs help "comando"`

Melhores Práticas - striping

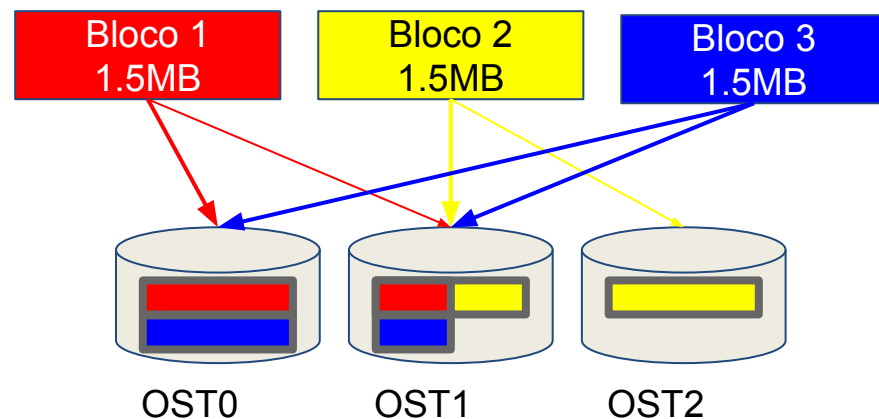
- Algumas razões para utilizar o striping incluem:
 - Fornece alta largura de banda para o acesso
 - quanto mais OSTs → mais vias para acessar o arquivo
 - Fornecendo espaço para arquivos muito grandes
 - quanto mais OSTs → mais espaço agregado para armazenamento
- Algumas razões para minimizar ou evitar o striping:
 - Aumento do overhead
 - Aumenta o número de *locks* e pode gerar “Disputas por I/O - Servidor”
 - Aumento do risco
 - Em caso de falha de um OSS/OST, parte do arquivo se perde
 - Todos arquivos (stripe = all) x Alguns arquivos (stripe = single)

Melhores Práticas - striping

- O `stripe_size` não possui efeito em um arquivo que possui `stripe_count` igual a 1
- O tamanho do stripe deve ser um múltiplo do tamanho da página (64KB)
- O menor tamanho recomendado de stripe é 512KB
- Um bom tamanho de stripe para I/O sequencial utilizando redes de alta velocidade é entre 1MB e 4MB
- O tamanho máximo para o stripe é de 4GB
- `stripe_count` = total de nós ou múltiplo do número de nós (16 nós -> 8 `stripe_count`)
- Escolher um padrão de stripe que leve em consideração os padrões de escrita da aplicação, evitando acesso desalinhado ao filesystem e contenção



1MB Stripe Size
3 Stripe Count



- Evite utilizar o comando “ls -l” (especialmente em diretórios)
 - “ls” se deseja verificar que um arquivo existe.
 - “ls -l arquivo” se deseja obter maiores informações de um arquivo específico.
- Evite ter um grande número de arquivos em um único diretório
- Evite acessar pequenos arquivos no Lustre (< 10MB)
- Utilize um “Stripe count” igual a 1 para diretórios com muitos arquivos pequenos
- Mantenha o código fonte no **HOMEDIR**
 - Compilar dentro do HOME → Mover para o SCRATCH
- Aumente o “stripe count” para o acesso paralelo ao mesmo arquivo
- Limite o Número de processos realizando E/S Paralela
- Experimente diferentes valores para o stripe count/size para as operações de escritas coletivas do MPI
 - MPI File write all / MPI File write at all / MPI File write ordered

FIM

Dúvidas?