Experimentations on encryption on top of Lustre

sbuisson@whamcloud.com

Experimentations on encryption on top of Lustre

Whamcloud

- What is encryption on top of Lustre?
- Various existing encryption solutions
  - Features review
  - Functional testing
  - Performance evaluation

Encryption on top of Lustre

Whamcloud

- What is encryption on top of Lustre?
  - use case:
    - provide special directory to users, to safely store secrets
  - goals:
    - protect data in transit between clients and servers
    - protect data at rest
  - challenge:
    - quick-and-easy solution to implement

Possible encryption solutions

- ext4
- eCryptfs
- EncFS
- gocryptfs
- dm-crypt/LUKS

Possible encryption solutions

Whamcloud

ext4

- file-based encryption handled directly at FS level, per directory
- AES-256
- master encryption key stored in kernel keyring
- from kernel 4.1
  - available in Ubuntu from 16.04
  - not backported to CentOS 7.5

Possible encryption solutions

Whamcloud

- eCryptfs

  - stacked file system, kernel space
  - AES-256
  - master encryption key stored in kernel keyring
  - from kernel 2.6.19

Possible encryption solutions

- EncFS
  - stacked file system, user space: FUSE
  - AES-192
  - master encryption key stored internally
  - available in any CentOS or Ubuntu distro

Possible encryption solutions

Whamcloud

- gocryptfs

  - stacked file system, written in GO, user space: FUSE
  - AES-256
  - master encryption key stored internally
  - available in recent Ubuntu, or via binaries

Possible encryption solutions

Whamcloud

- dm-crypt/LUKS

LUKS
Linux Unified Key Setup

  - block device encryption layer, kernel space
  - AES-256
  - master encryption key stored internally (key slots)
  - from kernel 2.6

Encryption solutions comparison

Whamcloud

name	type	need for privileges	content encryption	file name encryption	proper sparse file handling	hidden directory structure	file name length limit (bytes)	support for multiple keys
ext4	kernel space	no	each file individually	Yes	Yes	No	255	No
eCryptfs	kernel space, stacked	no	each file individually	Yes	No	No	143	No
EncFS	user space, FUSE	no	each file individually	Yes	Yes	No	190	No
gocryptfs	user space, FUSE	no	each file individually	Yes	Yes	No	255	No
dm-crypt/LUKS	kernel space, block device encryption layer	sudoers permissions	per-device encryption key	N/A	N/A	hidden in block device	imposed by chosen FS	Yes

Encryption solutions comparison

Whamcloud

- Caveats
  - ext4
    - filesystem cache not invalidated when key revoked
  - eCryptfs
    - deprecated since RH 6.6, still supported in Ubuntu, and present in upstream kernel
    - long-standing bugs when used on top of networked filesystems
  - EncFS
    - security audit from 2014 (v1.7) points to vulnerabilities, many not addressed as of v1.9
  - dm-crypt/LUKS
    - space available for encryption limited by block device size
    - need to protect against LUKS header corruption

Functional tests with encryption on top of Lustre

Whamcloud

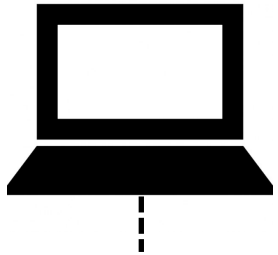
- Methodology
  - sanity.sh
  - sanityn.sh
- Must skip tests:
  - that depend on striping info
  - that rely on fid2path mechanism
  - that make use of Changelogs
    - some info may differ
  - that make use of jobstats
    - process doing IOs may differ

Encryption setup on top of Lustre

Whamcloud

- ext4

Ext4 File System



Lustre client

`/lustre/loopback` — Loopback file containing private data

`losetup`
`mkfs.ext4`
`mount`
`e4crypt add_key`

`/lustre/private` — Access to private data in clear

Functional tests

- ext4
  - sanity.sh
    - 109 failed out of 475
      - operations on files/dirs inside mount point are just read/write ops as seen by Lustre (loopback file)

      ⇒results difficult to interpret
    - some instability
      - due to Lustre master branch + Ubuntu 18.04 kernel?

Functional tests

- ext4
  - sanityn.sh
    - test 1 failed!

```
sanityn test_1: @@@@@@ FAIL: Check create
```

```
test_1() {
        touch $DIR1/$tfile
        [ -f $DIR2/$tfile ] || error "Check create"
        chmod 777 $DIR2/$tfile
        $CHECKSTAT -t file -p 0777 $DIR1/$tfile ||
           error "Check attribute update for 0777"

        chmod a-x $DIR2/$tfile
        $CHECKSTAT -t file -p 0666 $DIR1/$tfile ||
           error "Check attribute update for 0666"

        rm $DIR2/$tfile
        $CHECKSTAT -a $DIR1/$tfile ||
           error "Check unlink - removes file on
                  other mountpoint"
}
```

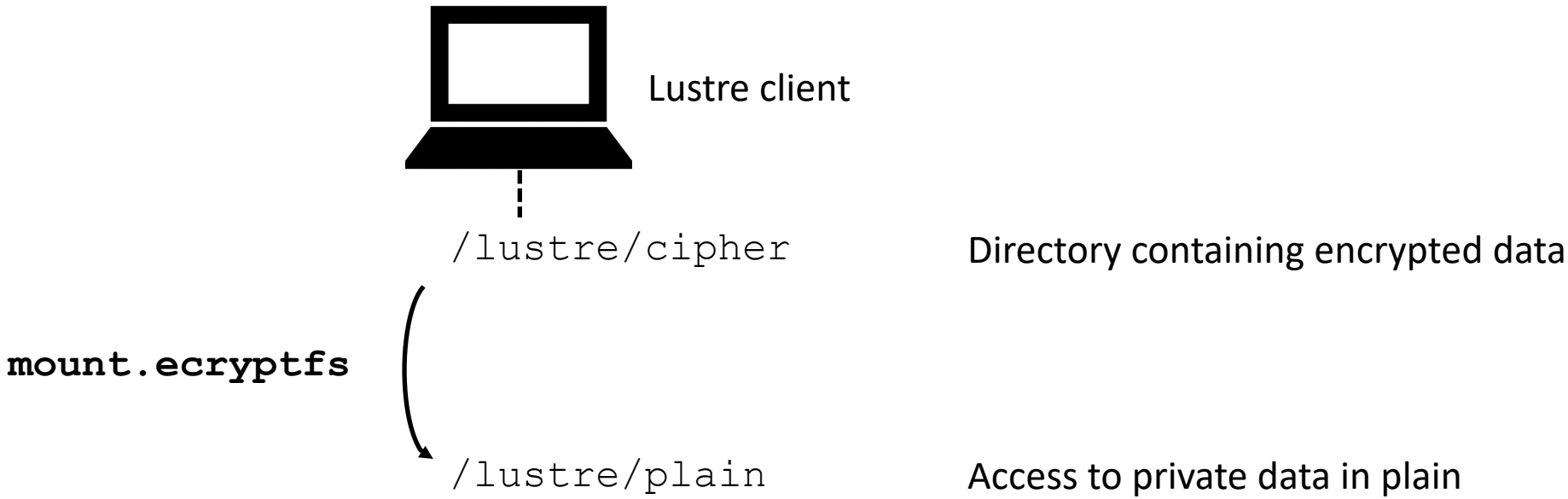
- inodes kept in cache and never checked for underneath change

  ⇒cannot work on top of parallel file system like Lustre

Encryption setup on top of Lustre

Whamcloud

- eCryptfs



Functional tests

- eCryptfs
  - sanity.sh
    - 40 failed out of 475, problems with:
      - sparse files
      - file name length
      - atime
      - file attributes
      - direct IO
    - lot of instability
      - not only due to Lustre master branch + Ubuntu 18.04 kernel

Functional tests

Whamcloud

- eCryptfs

  - sanityn.sh
    - test 1 failed!

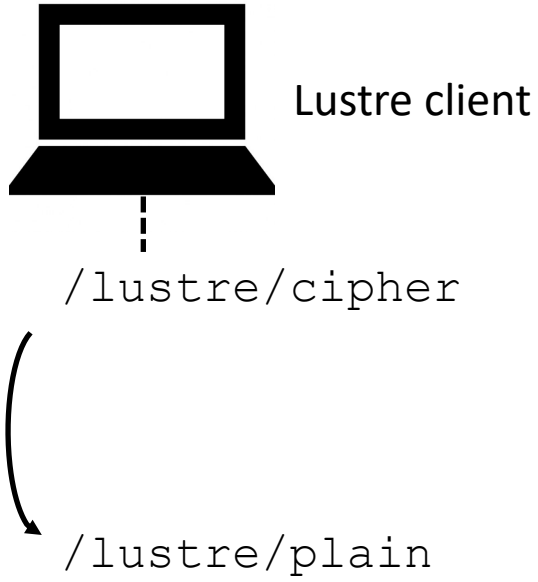
```
sanityn test_1: @@@@@@ FAIL: Check create
```

  - inodes kept in cache and never checked for underneath change

    ⇒cannot work on top of parallel file system like Lustre

Encryption setup on top of Lustre

- EncFS



Functional tests

- EncFS
  - sanity.sh
    - 35 failed out of 475, problems with:
      - file name length
      - atime
      - file attributes, ACLs
      - char/block device not supported
      - direct IO
  - possible to apply striping on ciphered file

```
encfsctl encode <rootdir> <file>
```

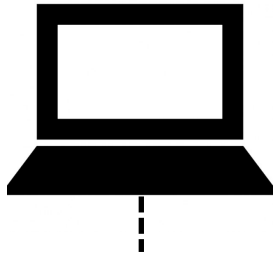
Functional tests

- EncFS
  - sanityn.sh
    - requires **`--nocache`** option
    - 17 failed out of 145, problems with:
      - openunlink
      - ops on executing file
      - erratic support of direct IO, works on occasions

Encryption setup on top of Lustre

Whamcloud

- gocryptfs



Lustre client

`/lustre/cipher` — Directory containing encrypted data

`gocryptfs -init`
`gocryptfs`

`/lustre/plain` — Access to private data in plain

Functional tests

Whamcloud

- gocryptfs

gocryptfs

  - sanity.sh

    - 43 failed out of 475, problems with:
      - hard links
      - file attributes, ACLs
      - only `user.` xattrs
      - char/block device not supported
      - direct IO

Functional tests

Whamcloud

- gocryptfs

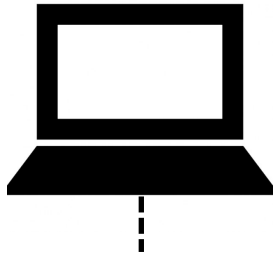
gocryptfs

  - sanityn.sh
    - requires **`-sharedstorage`** option
    - 14 failed out of 145, problems with:
      - openunlink
      - hard links
      - only `user.` xattrs

Encryption setup on top of Lustre

- dm-crypt/LUKS

LUKS
Linux Unified Key Setup



Lustre client

`/lustre/loopback` — Loopback file containing private data

`losetup`
`cryptsetup`
`mkfs.ocfs2`
`mount`

`/lustre/private` — Access to private data in clear

Functional tests

LUKS
Linux Unified Key Setup

Whamcloud

- dm-crypt/LUKS
  - sanity.sh
    - 30 failed out of 475
      - operations on files/dirs inside mount point are just read/write ops as seen by Lustre (loopback file)

      ⇒results difficult to interpret
  - possible to set striping on loopback file

Functional tests

LUKS
Linux Unified Key Setup

Whamcloud

- dm-crypt/LUKS
  - sanityn.sh
    - need shared storage FS (ocfs2) to see modifications from other Lustre clients
    - 14 failed out of 145, problems with:
      - openunlink
      - ops on executing file
      - ‘loopback-file effect’

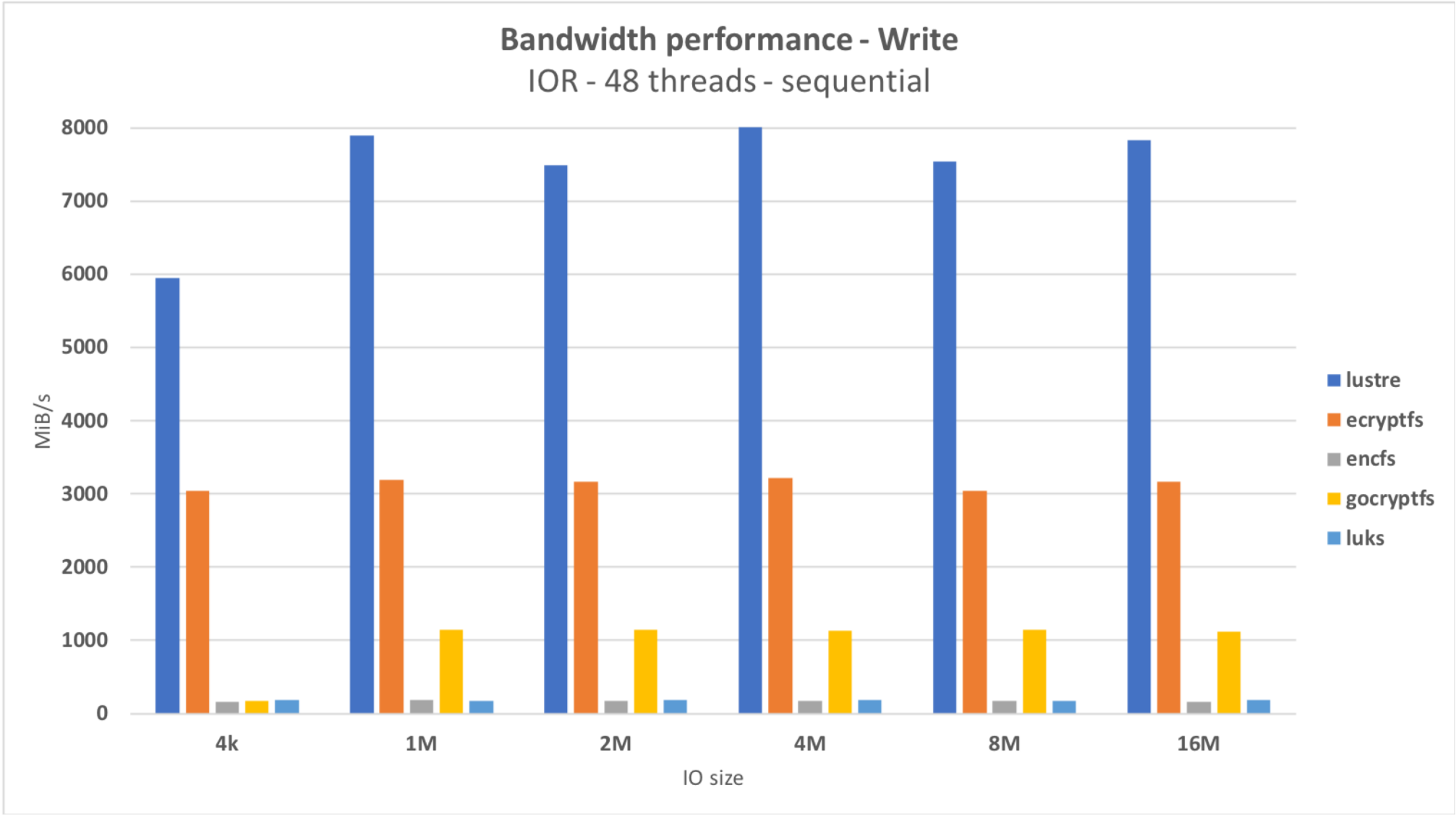
Performance evaluation with encryption on top of Lustre

Whamcloud

- Testbed
  - Client
    - Skylake 48 cores, Intel(R) Xeon(R) Platinum 8160 CPU @ 2.10GHz
    - 96 GB RAM
    - ConnectX-4 Infiniband adapter, EDR network
  - Server
    - 48 OSTs
- Methodology
  - IOR, file per process, sequential IO
  - IOR, file per process, random IO

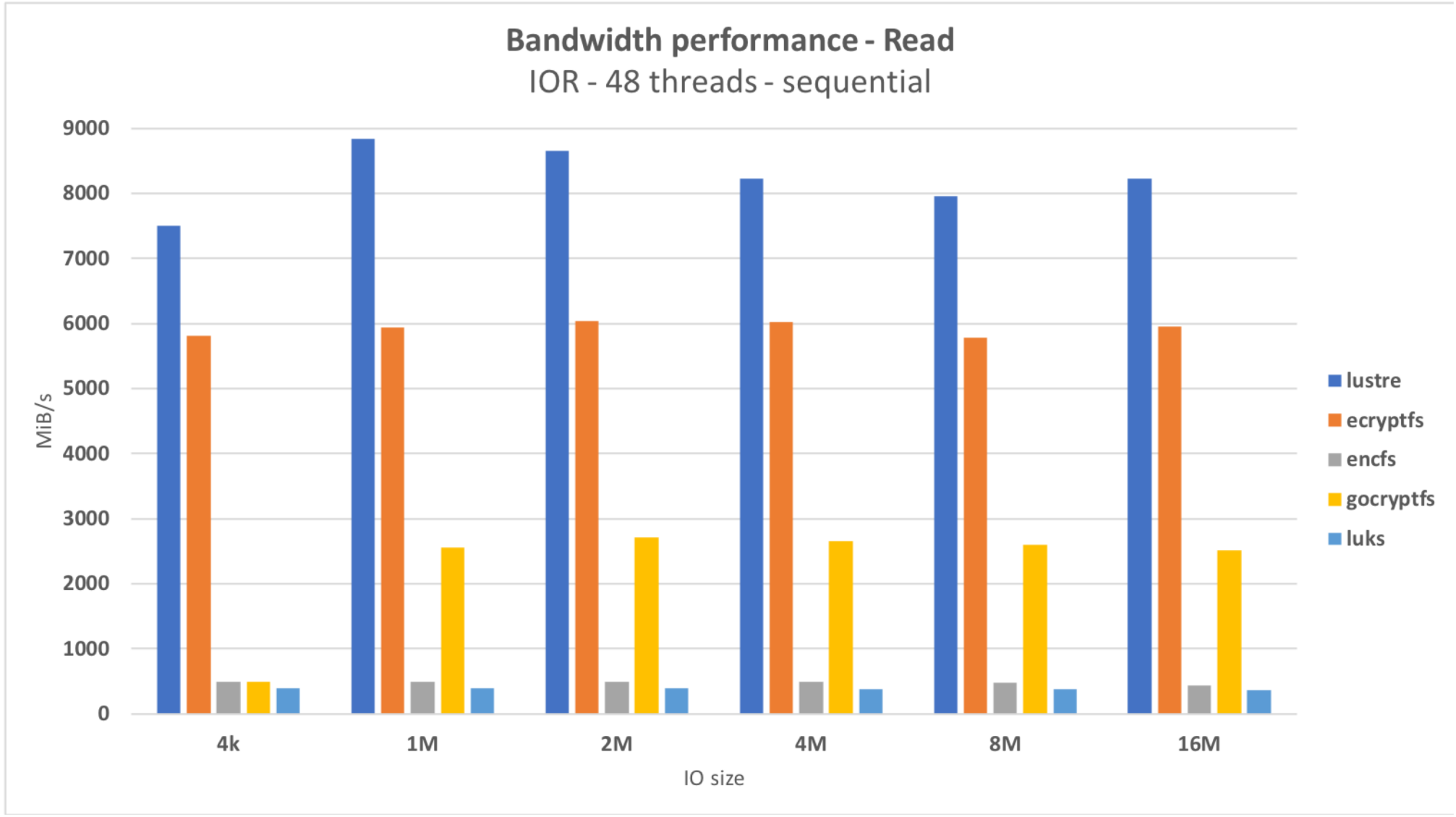
Performance evaluation with encryption on top of Lustre

Whamcloud



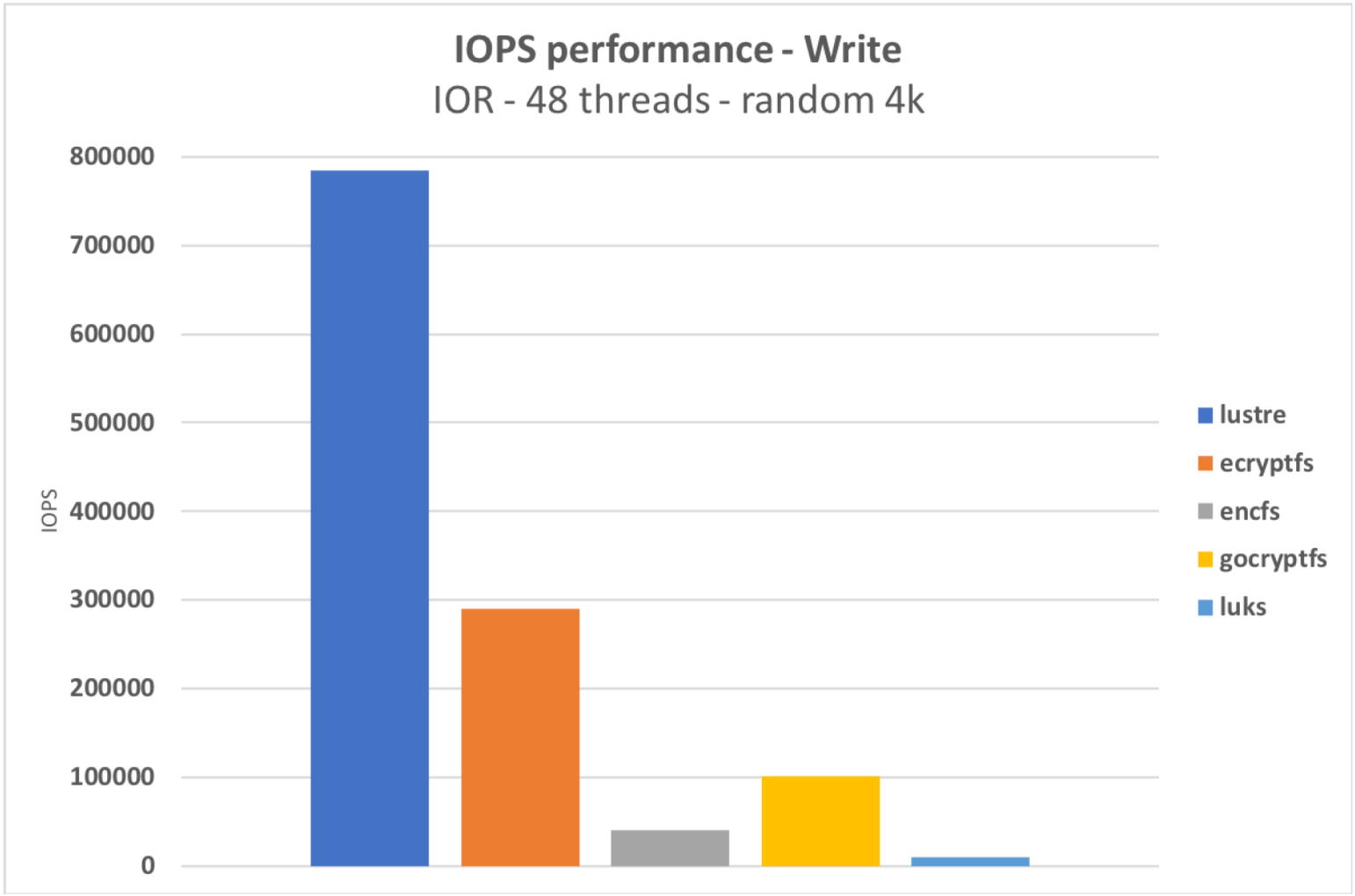
Performance evaluation with encryption on top of Lustre

Whamcloud



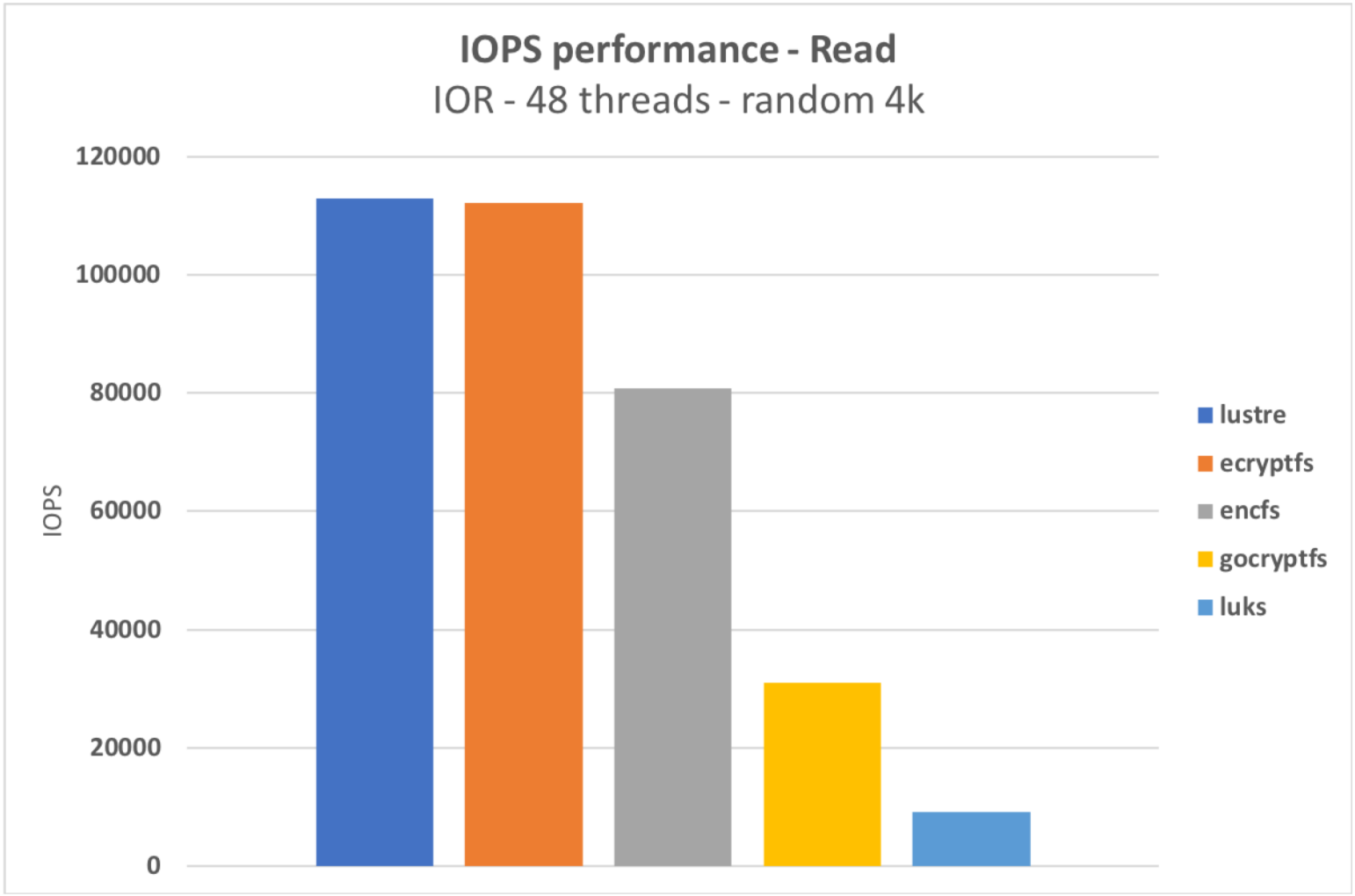
Performance evaluation with encryption on top of Lustre

Whamcloud



Performance evaluation with encryption on top of Lustre

Whamcloud



Performance evaluation with encryption on top of Lustre

Whamcloud

- Reasons behind numbers

	eCryptfs	EncFS 370% in top	gocryptfs 420% in top	dm-crypt/LUKS ('loop' at 100% in top)
Write CPU usage	78% in ecryptfs_encrypt_page (37% in _aesni_enc1)	28% in security routines (3% in libcrypto.so.1.0.0)	16% in AES routines (7% in crypto/aes.gcmAesEnc)	76% in osq_lock (1,17% in _aesni_enc4)
Read CPU usage	72% in ecryptfs_decrypt_page (18% in _aesni_dec4)	25% in security routines (0,7% in libcrypto.so.1.0.0)	12% in AES routines (6% in crypto/aes.gcmAesDec)	17% used by 'loop' (13% in _aesni_dec4)

Conclusion

Whamcloud

- This is just early stage of evaluation.
  - metadata performance evaluation
  - possible other encryption solutions
- Pretty low performance level
  - noticeable exception of eCryptfs, but **not** suitable for use on top of Lustre
- Advantage of availability and simplicity
  - wait for implementation inside Lustre for good performance?
- Key management is closely-related hot topic.

Thank you!

sbuisson@whamcloud.com