Lustre Client Encryption

10/2020

sbuisson@whamcloud.com

Lustre Client Encryption

- What is encryption for Lustre and solution proposal: fscrypt
- Features available with Lustre 2.14 : content encryption
- Remaining work
  - Performance optimizations
  - Name encryption
  - Compatibility with future releases

What is encryption for Lustre?

- Use case:
  - Provide special directory for each user, to safely store sensitive files
- Goals:
  - Protect data in transit between clients and servers
  - Protect data at rest
- Solution proposal
  - Conform to fscrypt kernel API
    - Current users are ext4, F2FS, and UBIFS
    - Core principle: pages in the page cache always contain clear text data
  - Make use of fscrypt userspace tool

Lustre Client Encryption – merged in 2.14

- Ability to encrypt file content
  - Encrypt on write, decrypt on read

- Ability to set encryption policies on directories
  - Support new IOCTLs from fscrypt userspace tool
  - Handle encryption context atomically

Lustre Client Encryption – patches

- LU-12275 sec: reserve flags for client side encryption
- LU-12275 osd: make osd layer always send complete pages
- LU-12275 sec: add llcrypt as file encryption library
- LU-12275 sec: documentation for client-side encryption
- LU-12275 sec: enable client side encryption
- LU-12275 sec: control client side encryption
- LU-12275 sec: encryption for write path
- LU-12275 sec: decryption for read path
- LU-12275 sec: deal with encrypted object size
- LU-12275 sec: support truncate for encrypted files
- LU-12275 tests: exercise file content encryption/decryption

Lustre Client Encryption – patches continued

- LU-12275 sec: ioctls to handle encryption policies
- LU-12275 sec: introduce null algo for filename encryption
- LU-12275 sec: force file name encryption policy to null
- LU-12275 sec: atomicity of encryption context getting/setting
- LU-12275 sec: encryption support for DoM files
- LU-12275 sec: check if page is empty with ZERO_PAGE
- LU-12275 sec: O_DIRECT for encrypted file
- LU-12275 sec: restrict fallocate on encrypted files
- LU-12275 sec: ldiskfs not aware of client-side encryption
- LU-12275 sec: encryption with different client PAGE_SIZE
- LU-12275 sec: verify dir is empty when setting enc policy*

Lustre Client Encryption

- Client encryption implementation compatible with:
  - File holes
    - Existence and location of holes in files not hidden
  - Truncate
    - On clear text, on client side only
  - MMAP
    - Pagecache for an encrypted file contains the plaintext
  - Direct IO
    - By twisting pages being used for sending RPCs
  - File mirroring
  - DoM
  - Mixed 4KB/64KB PAGE_SIZE clients

Lustre Client Encryption – encryption context handling

- Encryption context handling
  - Per-file encryption context is stored in an xattr
    - Hopefully not changed after file creation
- Encryption context atomicity
  - ‘setting’ case
    - Send encryption context to the MDT along with create RPCs
      - Closing insecure window between creation and setting of encryption context
      - Saving a setxattr request for better performance
  - ‘getting’ case
    - Server returns encryption context upon granted lock reply
      - Making the encryption context retrieval atomic
      - Saving a getxattr request for better performance

Lustre Client Encryption – new ioctls for policies

- fscrypt API v2
  - Revokes pages from page cache when encryption key is removed
  - Landed in Linux 5.4
    - But we need to support CentOS 8 (4.18) / Ubuntu 18 (4.15) clients

- Solution retained
  - Include fscrypt API from Linux v5.4 inside Lustre tree
    - Renamed to llcrypt to avoid name conflicts
    - Available in libcfs module
  - Wire up new ioctls in llite

Lustre Client Encryption – new ioctls for policies

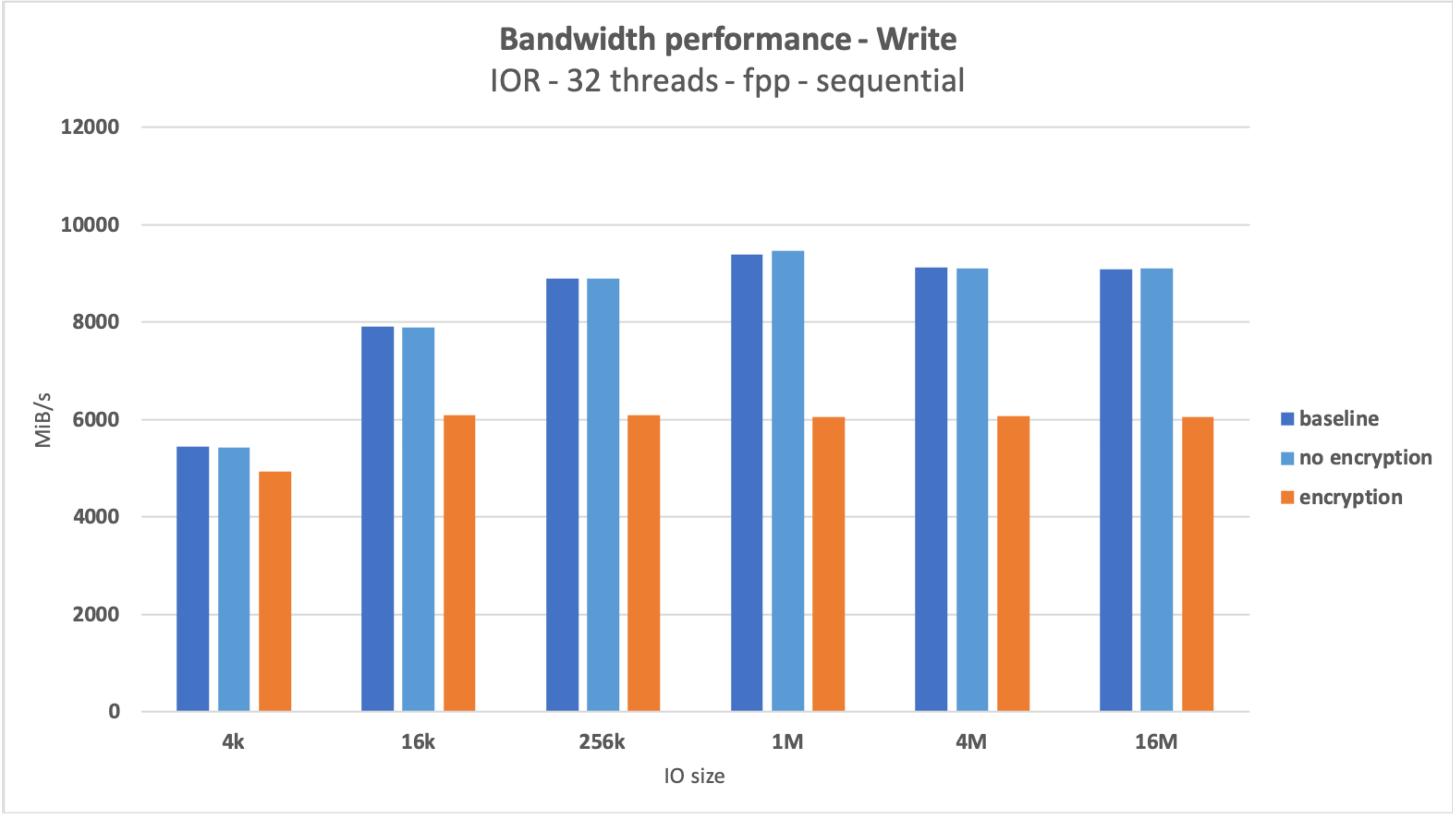
- fscrypt userspace tool
  - Works with Lustre out of the box, thanks to fscrypt API support
  - Supports encryption policies v2
  - Associates protectors (passphrase, raw key, pam) to policies

```
# fscrypt setup /mnt/lustre
$ fscrypt encrypt /mnt/lustre/vault
$ fscrypt lock /mnt/lustre/vault
$ fscrypt unlock /mnt/lustre/vault
$ fscrypt metadata change-passphrase
    --protector=/mnt/lustre:7626382168311a9d
$ fscrypt metadata add-protector-to-policy
    --protector=/mnt/lustre:2c75f519b9c9959d
    --policy=/mnt/lustre:16382f282d7b29ee
```

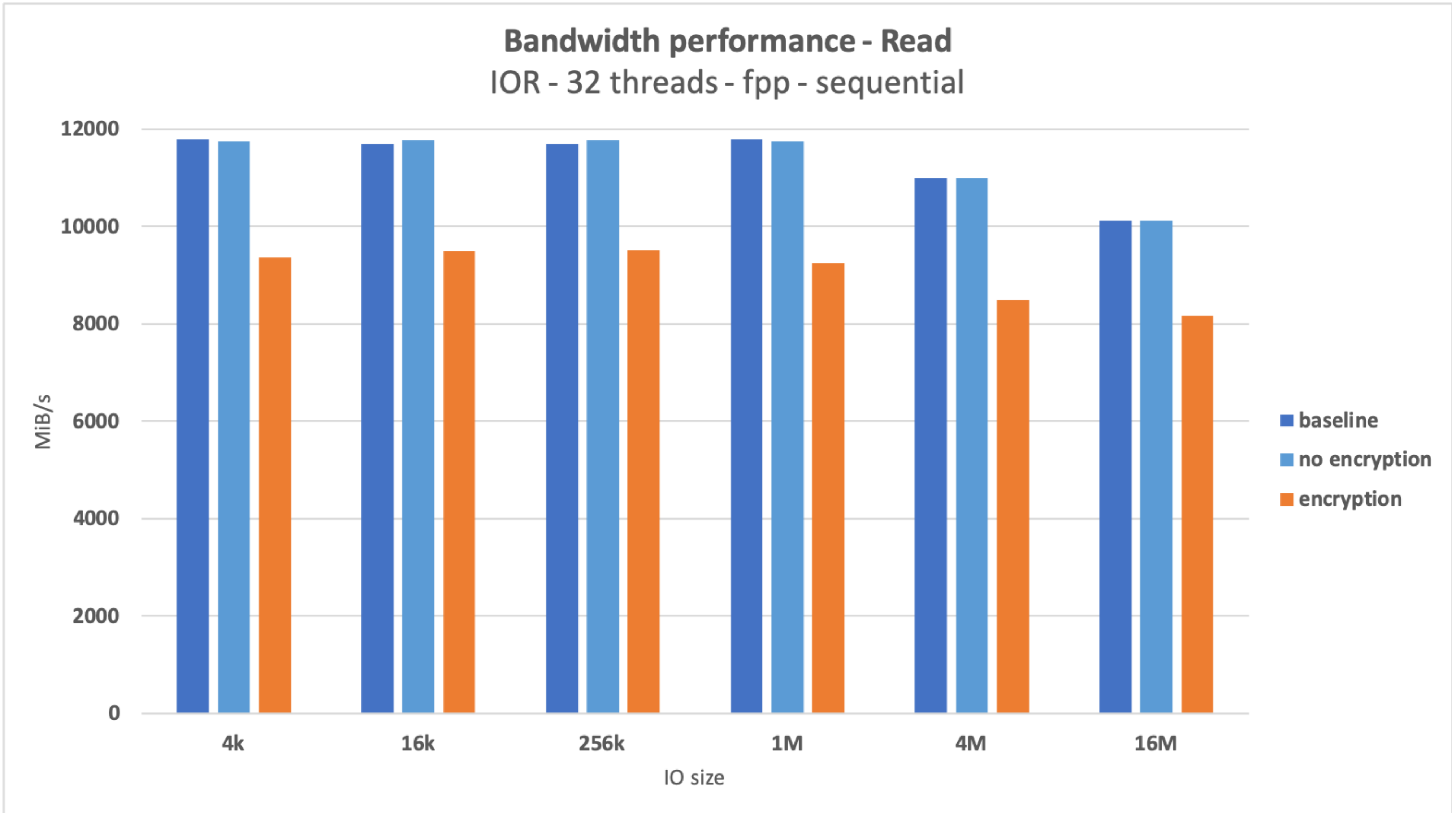
Lustre Client Encryption – bandwidth performance

- POC code on top of `master`, **dummy encryption mode** (AES-256-XTS)
- Testbed
  - Client
    - Skylake 48 cores, Intel(R) Xeon(R) Platinum 8160 CPU @ 2.10GHz
    - 96 GB RAM
    - ConnectX-4 Infiniband adapter, EDR network
  - Storage
    - DDN ES200NV, 20 x NVMe HGST 1,7TB, 1 DCR pool
    - 4 OSTs, each 1/10th of pool
- Methodology
  - IOR, file per process, sequential IO

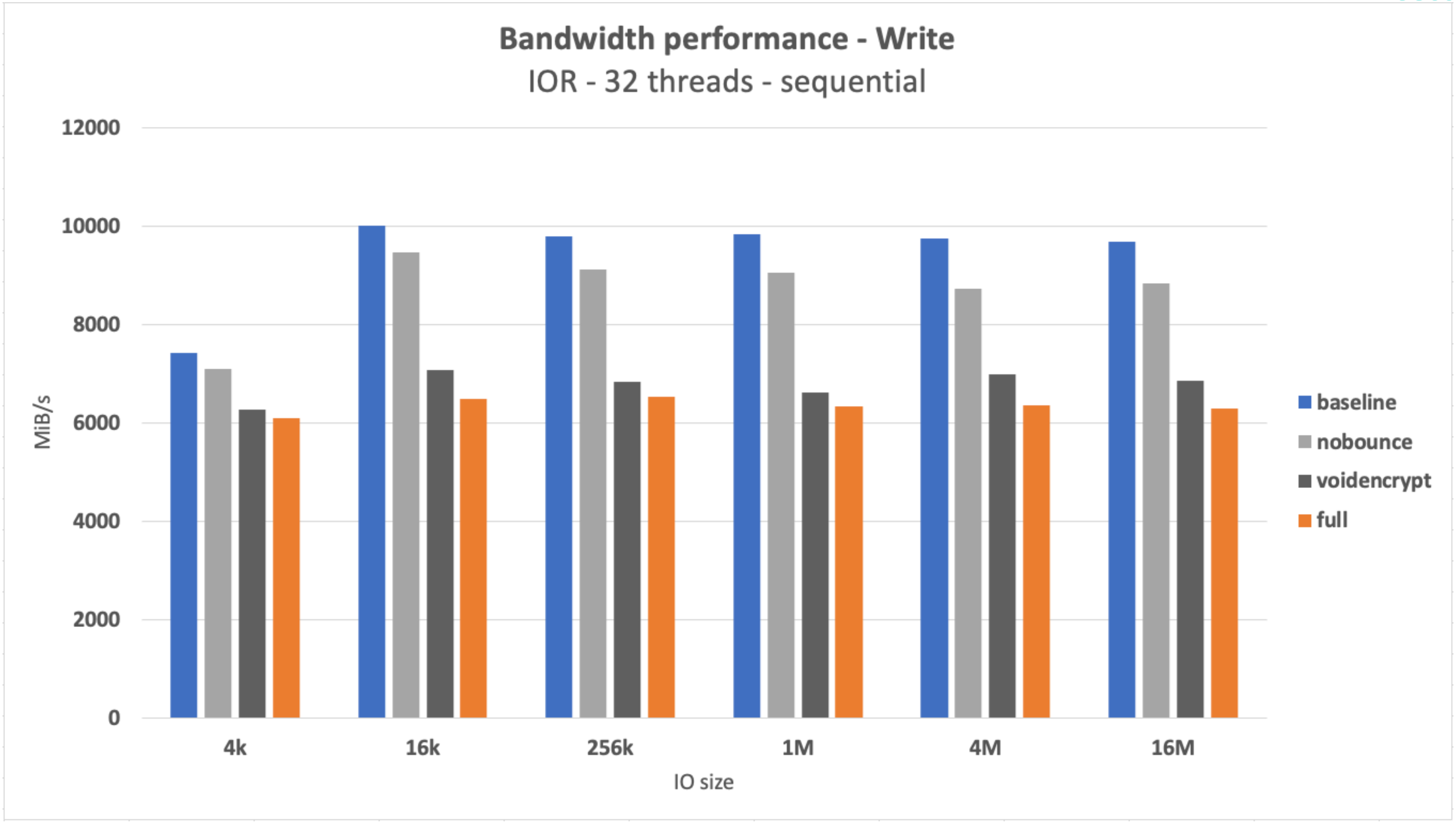
Lustre Client Encryption – bandwidth performance



Lustre Client Encryption – bandwidth performance



Lustre Client Encryption – performance investigations



Lustre Client Encryption – performance investigations

- Compare `nobounce` and `voidencrypt`
  - `nobounce`: encryption but no bounce page allocation: 10% drop
  - `voidencrypt`: no encryption but bounce page allocation: 30% drop

⇒ bounce page allocation hurts

- Possible optimization path
  - Leverage Lustre's enc_pool mechanism
    - Take bounce pages from this pool
    - Do not allocate bounce page for every call to encryption primitive

Lustre Client Encryption – name encryption

- POC just started: LU-13717
- Wire up llcrypt API in llite to encrypt/decrypt names
- Convert between plain text and cipher text names
  - From plain to cipher before sending request to MDT
  - From cipher to plain upon reply
  - 2 cases to support
    - Access with the key: present actual names
    - Access without the key: base64 encoding of cipher text names

Lustre Client Encryption – name encryption challenges

- ‘name’ is no longer a valid path name, not even a well-formed string
  - Binary ciphertext names just cannot be encoded (base64 or similar)
    - fscrypt API supports plain text file names of up to NAME_MAX length
    - NAME_MAX limit would be exceeded when encoding

- Hopefully, ldiskfs and ZFS backend file systems are capable of handling binary names
  - Send binary names to server side
  - Make server side handle binary names

Lustre Client Encryption – name encryption preliminary testing

- Works fine with DNE
  - Thanks to FID mapping, no particular aspect to take care of
- fid2path broken
  - Because path is built on server side…
  - … and Lustre server side is not encryption aware
- Metadata performance benchmarks to be done

Lustre Client Encryption – releases compatibility

- **Compatibility with future versions**
  - Lustre 2.14 will have content encryption only
  - Future versions will add name encryption
  - But fscrypt policies designed to handle both content and name encryption

- **Problem when upgrading from 2.14**
  - New code will try to decrypt clear text names created with 2.14

- **Solution proposal**
  - Add Lustre specific LLCRYPT_MODE_NULL
  - Enforce in 2.14 for name encryption == no name encryption

Lustre Client Encryption

Whamcloud

- Projected roadmap
  - Content encryption
  - fscrypt inclusion
  - Encryption policies support

land in 2.14

  - Metadata encryption
  - Performance optimizations

target 2.15

Whamcloud

Thank you!

sbuisson@whamcloud.com

ddn

https://cnb.cx/2EdCoGr