Look Inside™

# Lustre* consistency verification

Fan Yong (fan.yong@intel.com)
High Performance Data Division, Intel ® Corporation

**Breakthrough Storage Performance**
**LUG 2014**

Oct 14 2014
Beijing, China

*Other names and brands may be claimed as the property of others.

# Outline

## Lustre* consistency issues

- Dangling reference, orphan object, repeated reference, …

## Lustre consistency framework

- FID-in-LMA, linkEA, parent FID for OST-object

## Lustre consistency verification tools - LFSCK

- OI scrub, layout LFSCK, namespace LFSCK

*Other names and brands may be claimed as the property of others.

# Lustre* consistency issues

*Other names and brands may be claimed as the property of others.

# Some Lustre* consistency issues

- Dangling reference: where did my file/data go?
  - Name entry references non-exist or invalid MDT-object.
  - MDT-object references non-exist or invalid OST-object (via its LOV EA).
- Orphan object: who consumed my space?
  - No name entry references the MDT-object.
  - No MDT-object references the OST-object.
- Repeated reference: why has my data been overwritten?
  - Multiple MDT-objects reference the same OST-object.
  - Multiple objects references the same block.
    - Backend local consistency verification tools, such as e2fsck for ldiskfs/extN, focus on that. Lustre will use them and put more effort on other distributed consistency issues verification.

*Other names and brands may be claimed as the property of others.

# Lustre* special consistency issues

- Object Index (OI)
  - OI is used for mapping the object's global FID to server backend local identifier (such as <inode#, generation#> for ldiskfs).
  - Lost the OI mapping will cause the object to be invisible when locate the object by FID.
  - Corrupted OI mapping may misguide the application to access some unexpected object and cause unpredictable result.
- FID-in-dirent (directory entry)
  - To accelerate traversing directory, the FID of the object that is referenced by the dirent is appended after the name in the dirent.
  - Lost the FID-in-dirent will cause additional reading FID from the object (maybe load from disk) when traverse the directory (READDIR).
  - Corrupted FID-in-dirent may misguide the application to access some unexpected object and cause unpredictable result.

*Other names and brands may be claimed as the property of others.

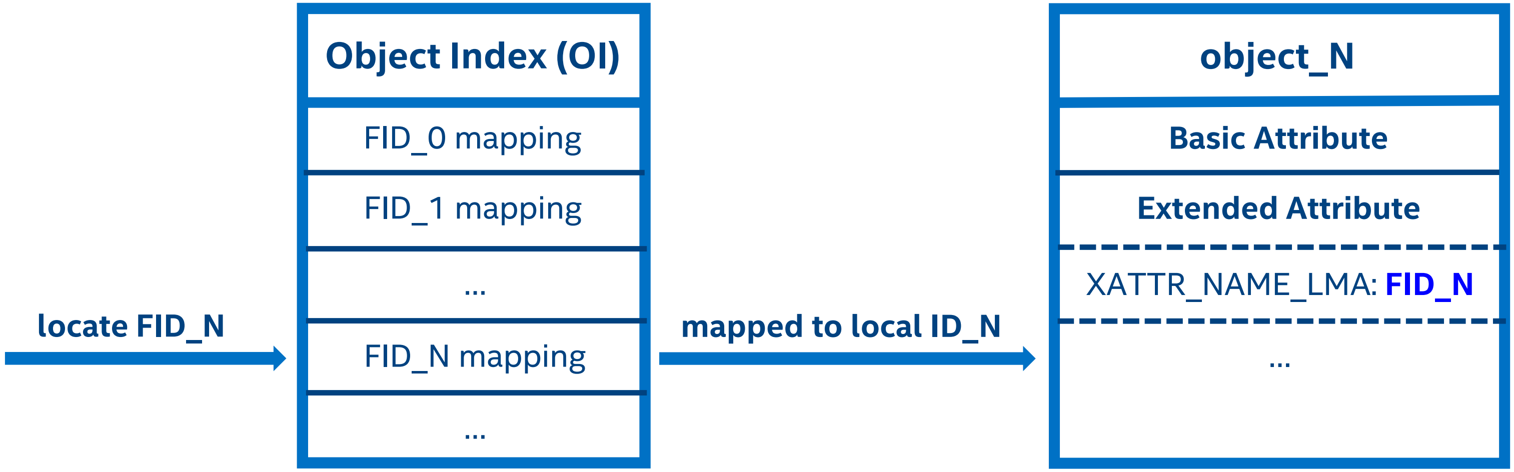
# Lustre* consistency framework

*Other names and brands may be claimed as the property of others.

# FID-in-LMA

Lustre* object stores its FID in the XATTR_NAME_LMA extended attribute (EA) for related OI mapping consistency self-verification.

- To check whether the object found by the FID is the expect one or not. If NOT, the application will get -**EREMCHG** (-78).
- The FID-in-LMA can be used to rebuild the Lustre OI.

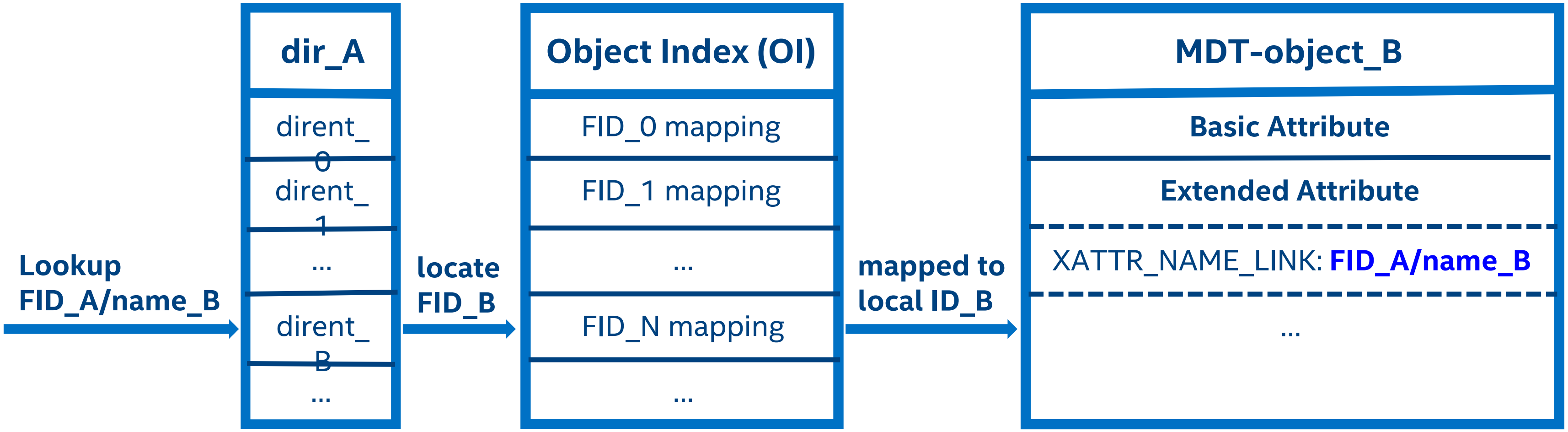


*Other names and brands may be claimed as the property of others.

# linkEA

The MDT-object stores its position (in namespace) information (the name and the parent FID) as XATTR_NAME_LINK EA.

- To knows where the given MDT-object resides in the (original) namespace.
- The linkEA can be used to rebuild the Lustre* namespace.

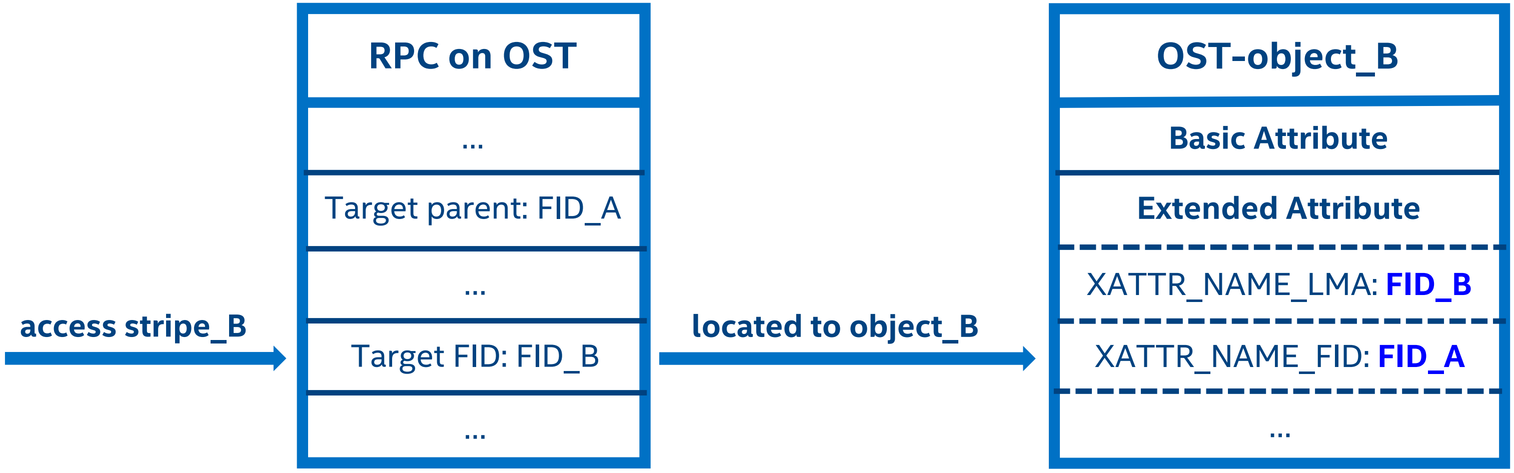


*Other names and brands may be claimed as the property of others.

# parent FID for OST-object

The OST-object stores the FID of its parent MDT-object that references the OST-object as XATTR_NAME_FID EA.

- To check whether the OST-object to be operated belongs to the given target (MDT-object) or not.
- The parent FID for OST-object can be used to rebuild the MDT-object LOV EA.



# Lustre* consistency verification tools - LFSCK

*Other names and brands may be claimed as the property of others.

# New LFSCK goals

- Online verification
  - LFSCK routine verification with normal Lustre* services non-stopped.
  - Speed is controllable to avoid affecting normal services too much.
- Robust
  - Allow servers (MDT/OST) to join/exit the LFSCK dynamically.
  - Resume the LFSCK from the latest checkpoint (breakpoint).
- Scalable
  - LFSCK on thousands of servers in parallel, the aggregate verification speed will increase as the servers count increasing.
  - Support DNE (Distributed NamespacE) mode consistency verification.

*Other names and brands may be claimed as the property of others.

# LFSCK engines

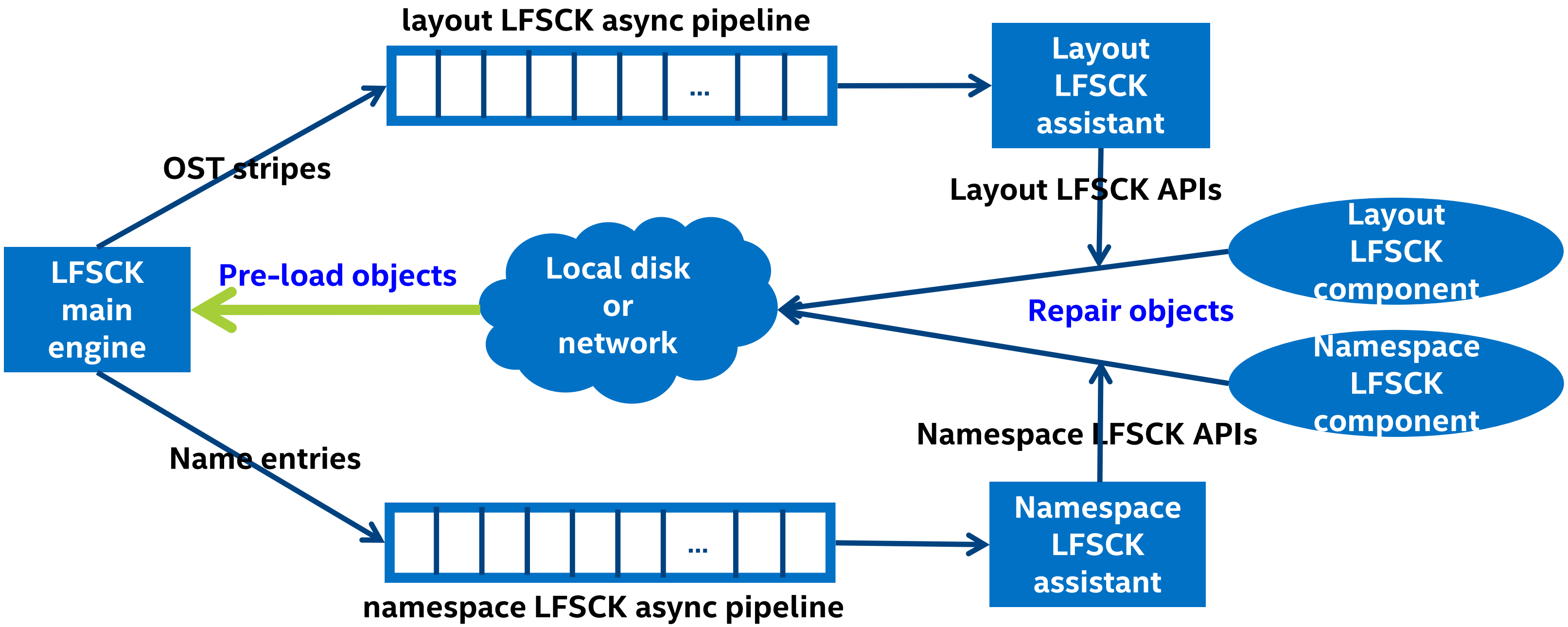
LFSCK is driven by the LFSCK engines to verify the objects in the whole/partial system.

- Each Lustre* MDT/OST has a main engine.
  - All the main engines are equal, no central control-point.
  - All the main engines are relative independent.
    - Each main engine only verifies the objects in its own scope.
- Each main engine on MDT may has some assistant engine(s).
  - The main engine and the assistant engine(s) compose some asynchronous pipeline(s).
    - The main engine loads objects (from disk or network) and input the pipeline.
    - The assistant engine verifies the objects consistency from the pipeline output.

*Other names and brands may be claimed as the property of others.

# LFSCK components

Every LFSCK component corresponds to one of the LFSCK verification types (OI scrub/layout LFSCK/namespace LFSCK). LFSCK uses the LFSCK component's APIs to verify the object.



# LFSCK component – OI scrub

## Special for ldiskfs-based backend to verify OI files.

- Basic principle
  - Trust FID-in-LMA if LMV EA is there.
  - Linearly scan all objects on the local device.
- Use cases
  - Re-create OI files totally
    - Some OI files are lost.
    - Split OI files to improve OI efficiency.
    - Shrink OI size to release the disk space occupied by empty FID mappings.
  - Re-build FID mappings after MDT file-level backup/restore
    - Backend local identifier (inode#/generation#) cannot be preserved when MDT file-level backup, but the FID mappings in OI are kept after the restoring.
  - Recover backend orphans on OST from /lost+found to Lustre* OI

*Other names and brands may be claimed as the property of others.

# LFSCK component – layout LFSCK

For regular striped file layout consistency between MDT and OST.

- Basic principle
  - For a regular file, the MDT-object references the stripes (OST-objects) via LOV EA; the OST-object back references the MDT-object via PFID EA.
  - The LFSCK on the MDT verifies the stripes in all MDT-objects' LOV EA.
  - The LFSCK on the OST records non-verified OST-objects that are orphans.
  - Share the same linear iterator as OI scrub used for scanning.
- Use cases
  - Guarantee that your data is written to the right OST-object(s).
  - Find the lost data via re-generating the lost or corrupted LOV EA.
  - Retrieve the lost space (occupied by the stale orphan OST-objects).

*Other names and brands may be claimed as the property of others.

# LFSCK component – namespace LFSCK

For local/global namespace consistency inside/among MDT(s).

- Basic principle
  - Traverse the namespace on MDT, for each name entry, check whether the referenced MDT-object has linkEA to back references the name entry.
  - Statistics the name entries that reference the same MDT-object to verify the MDT-object's nlink attribute.
  - Share the linear iterator, and plus namespace-based directory traversing.
- Use cases
  - Guarantee that the name entry references the right MDT-object.
  - Find the lost file/MDT-object via re-generating the name entry.
  - Retrieve the lost space (occupied by the stale orphan MDT-objects).
  - Guarantee that the nlink attribute matches the real name entries.
  - Verify FID-in-dirent, name hash for striped directory, and so on.

# User Interfaces – start LFSCK

```
lfsck_start <-M | --device {MDT,OST}_device> [-A | --all] [-c | --create_ostobj [on | off]]
            [-C | --create_mdtobj [on | off]] [-e | --error {continue | abort}] [-h | --help]
            [-n | --dryrun [on | off]] [-o | --orphan] [-r | --reset] [-s | --speed ops_per_sec_limit]
            [-t | --type check_type[,check_type...]] [-w | --window_size size]
```

options:

-M: device to start LFSCK/scrub on

-A: start LFSCK on all MDT devices

-c: create the lost OST-object for dangling LOV EA (default 'off', or 'on')

-C: create the lost MDT-object for dangling name entry (default 'off', or 'on')

-e: error handle mode (default 'continue', or 'abort')

-h: this help message

-n: check with no modification (default 'off', or 'on')

-o: repair orphan OST-objects

-r: reset scanning to the start of the device

-s: maximum items to be scanned per second (default '0' = no limit)

-t: check type(s) to be performed (default all)

-w: window size for async requests pipeline

# User Interfaces – stop LFSCK

```
lfsck_stop <-M | --device {MDT,OST}_device>
           [-A | --all] [-h | --help]
```

options:

-M: device to stop LFSCK/scrub on

-A: stop LFSCK on all MDT devices

-h: this help message

# User Interfaces – query LFSCK

- Query OI scrub
  - /proc/fs/lustre/osd-ldiskfs/${FSNAME}-MDTxxxx/oi_scrub
  - /proc/fs/lustre/osd-ldiskfs/${FSNAME}-OSTxxxx/oi_scrub
- Query layout LFSCK
  - /proc/fs/lustre/mdd/${FSNAME}-MDTxxxx/lfsck_layout
  - /proc/fs/lustre/obdfilter/${FSNAME}-OSTxxxx/lfsck_layout
- Query namespace LFSCK
  - /proc/fs/lustre/mdd/${FSNAME}-MDTxxxx/lfsck_namespace

# LFSCK project processing

Congratulations if you are using Lustre* 2.3 or newer!

- LFSCK 1 – OI scrub & object-table based linear iteration
  - Released in Lustre 2.3
- LFSCK 1.5 – FID-in-dirent & linkEA for local MDT
  - namespace LFSCK part1
  - Released in Lustre 2.4
- LFSCK 2 – layout LFSCK
  - Released in Lustre 2.6
- LFSCK 3 – LFSCK for DNE
  - Namespace LFSCK part2
  - To be released in Lustre 2.7
- ...

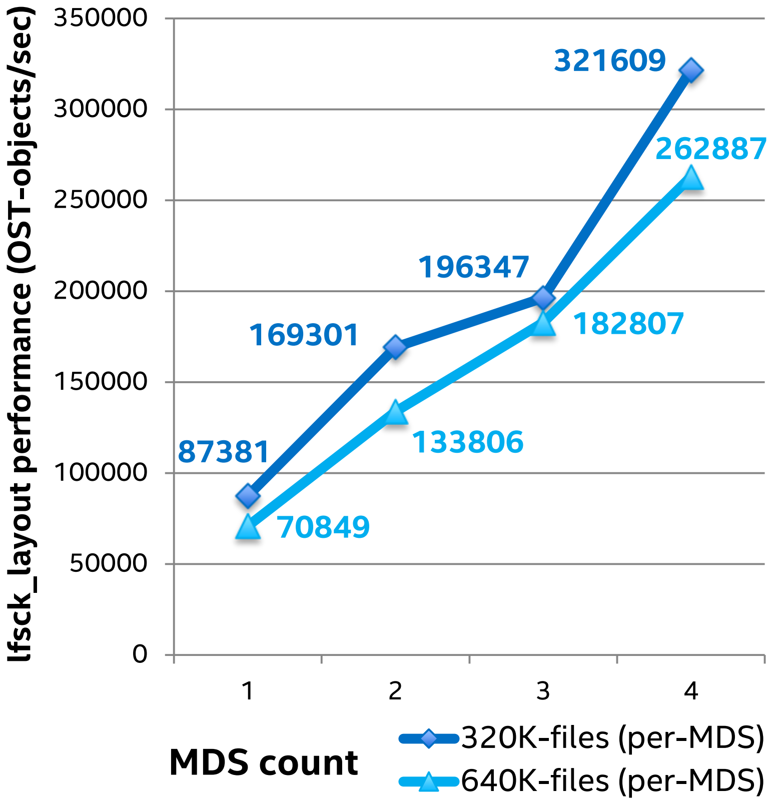
*Other names and brands may be claimed as the property of others.

# LFSCK performance test environment

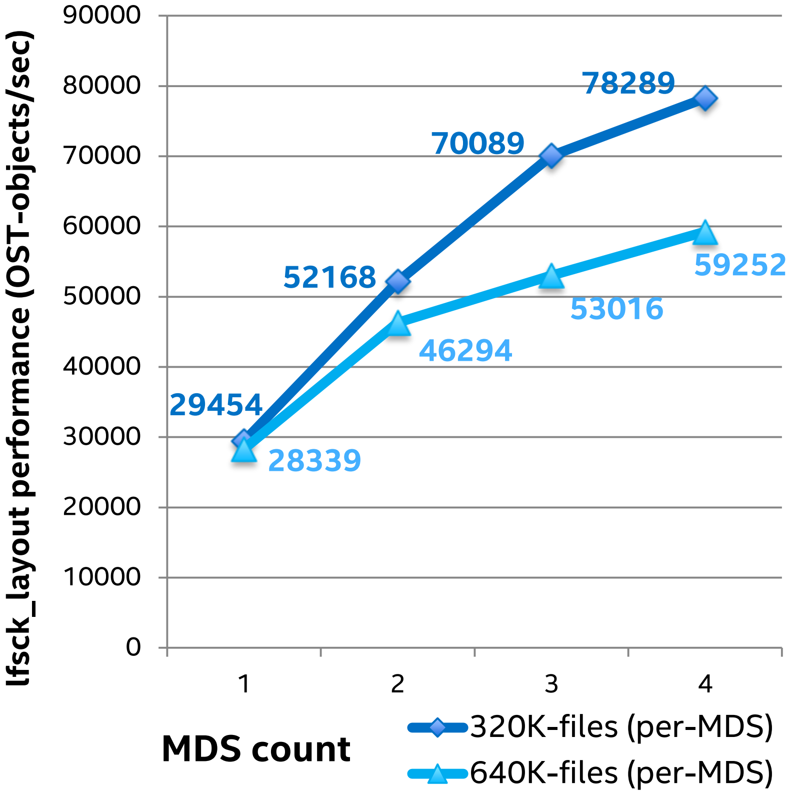
- CPU
  - 1 * Intel® Xeon® CPU E5620 @ 2.40GHz, 8 logic processors
- RAM
  - 32GB DDR3 RAM on each server (MDS/OSS) node
- Storage
  - 500GB 7200 rpm SATA disk on each server node
- Network
  - InfiniBand QDR
- Logic servers
  - 4 MDS nodes, 1 MDT per MDS
  - 4 OSS nodes, 2 OSTs per OSS
  - 1 client node, multiple mount points

# Layout LFSCK performance

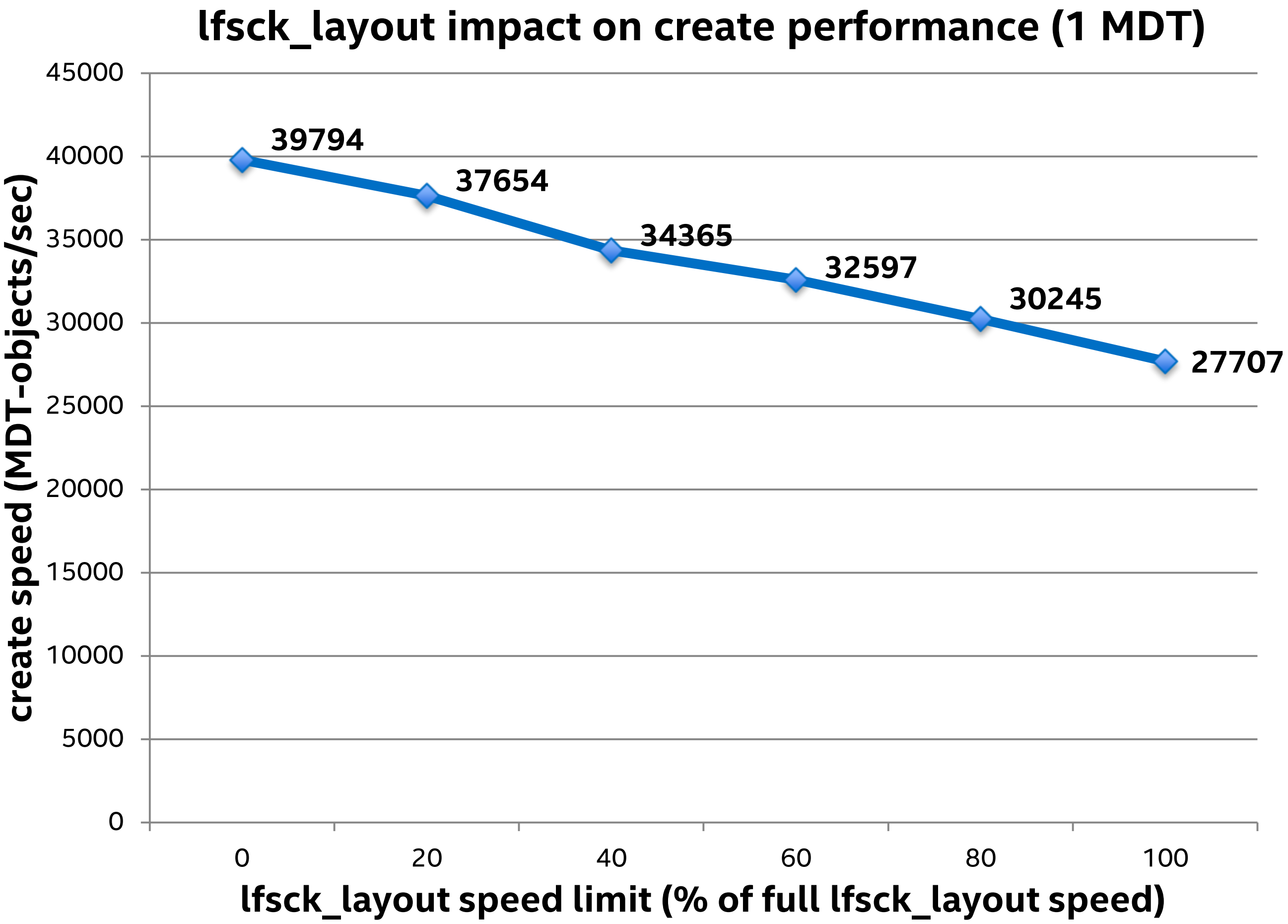
lfsck_layout routine check (bundle) performance under DNE



lfsck_layout repair dangling (bundle) performance under DNE



# Layout LFSCK impact on others



intel®

# Legal Disclaimer

INFORMATION IN THIS DOCUMENT IS PROVIDED IN CONNECTION WITH INTEL PRODUCTS.  NO LICENSE, EXPRESS OR IMPLIED, BY ESTOPPEL OR OTHERWISE, TO ANY INTELLECTUAL PROPERTY RIGHTS IS GRANTED BY THIS DOCUMENT.  EXCEPT AS PROVIDED IN INTEL'S TERMS AND CONDITIONS OF SALE FOR SUCH PRODUCTS, INTEL ASSUMES NO LIABILITY WHATSOEVER AND INTEL DISCLAIMS ANY EXPRESS OR IMPLIED WARRANTY, RELATING TO SALE AND/OR USE OF INTEL PRODUCTS INCLUDING LIABILITY OR WARRANTIES RELATING TO FITNESS FOR A PARTICULAR PURPOSE, MERCHANTABILITY, OR INFRINGEMENT OF ANY PATENT, COPYRIGHT OR OTHER INTELLECTUAL PROPERTY RIGHT.
A "Mission Critical Application" is any application in which failure of the Intel Product could result, directly or indirectly, in personal injury or death.  SHOULD YOU PURCHASE OR USE INTEL'S PRODUCTS FOR ANY SUCH MISSION CRITICAL APPLICATION, YOU SHALL INDEMNIFY AND HOLD INTEL AND ITS SUBSIDIARIES, SUBCONTRACTORS AND AFFILIATES, AND THE DIRECTORS, OFFICERS, AND EMPLOYEES OF EACH, HARMLESS AGAINST ALL CLAIMS COSTS, DAMAGES, AND EXPENSES AND REASONABLE ATTORNEYS' FEES ARISING OUT OF, DIRECTLY OR INDIRECTLY, ANY CLAIM OF PRODUCT LIABILITY, PERSONAL INJURY, OR DEATH ARISING IN ANY WAY OUT OF SUCH MISSION CRITICAL APPLICATION, WHETHER OR NOT INTEL OR ITS SUBCONTRACTOR WAS NEGLIGENT IN THE DESIGN, MANUFACTURE, OR WARNING OF THE INTEL PRODUCT OR ANY OF ITS PARTS.

Intel may make changes to specifications and product descriptions at any time, without notice.  Designers must not rely on the absence or characteristics of any features or instructions marked "reserved" or "undefined".  Intel reserves these for future definition and shall have no responsibility whatsoever for conflicts or incompatibilities arising from future changes to them.  The information here is subject to change without notice.  Do not finalize a design with this information.

The products described in this document may contain design defects or errors known as errata which may cause the product to deviate from published specifications.  Current characterized errata are available on request.

Contact your local Intel sales office or your distributor to obtain the latest specifications and before placing your product order.
Copies of documents which have an order number and are referenced in this document, or other Intel literature, may be obtained by calling 1-800-548-4725, or go to:  http://www.intel.com/design/literature.htm

Intel, Look Inside and the Intel logo are trademarks of Intel Corporation in the United States and other countries.

*Other names and brands may be claimed as the property of others.