



**Lustre Integrated Policy Engine**

**Lu Jingjing**

DataDirect Networks

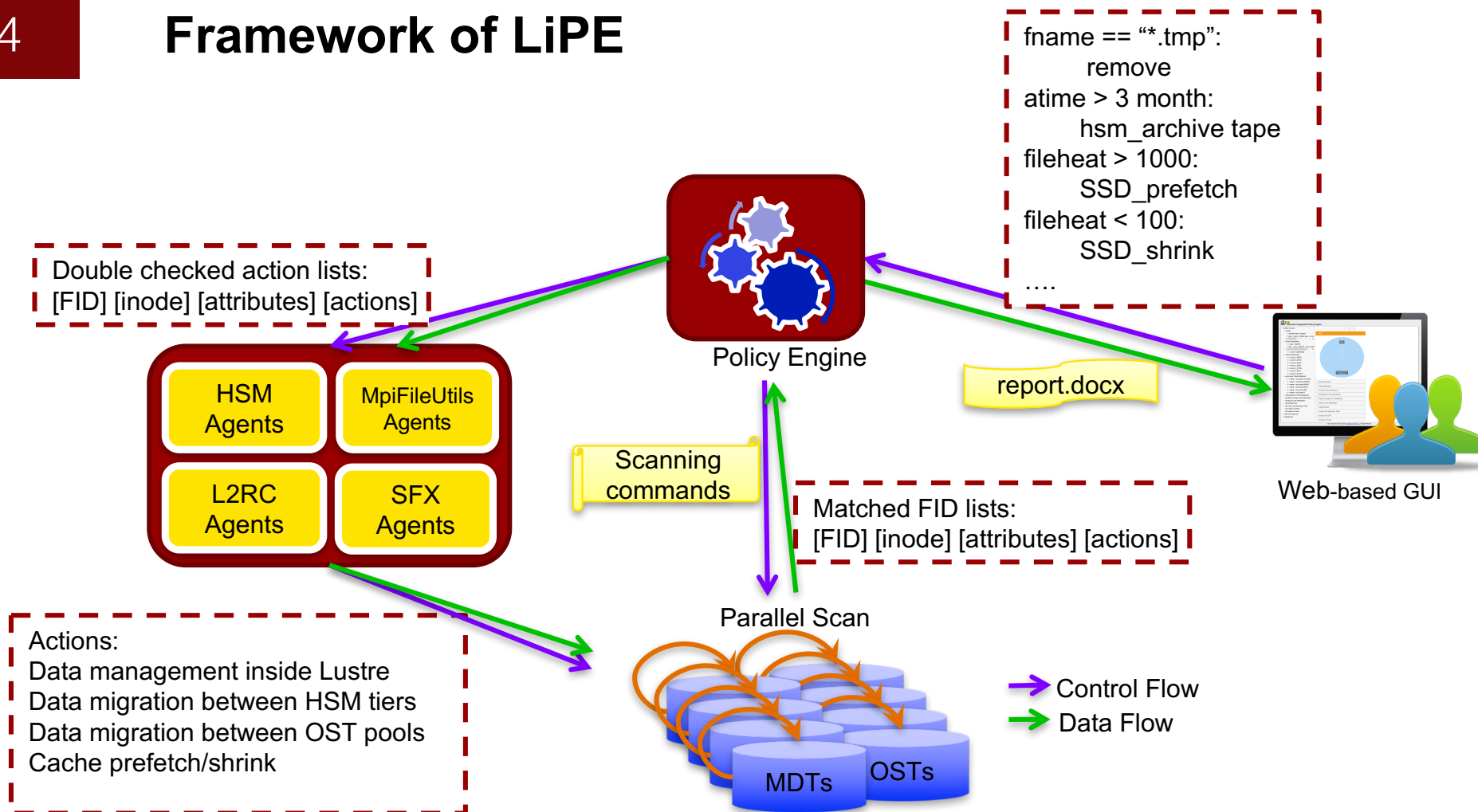
## Why LiPE?

- ▶ **Administrators are short of tools to do data management on Lustre/Storage**
- ▶ **Writing simple scripts without Lustre internal knowledge are far from enough**
  - Not easy to achieve high speed.
    - Scanning directory tree is not efficient
  - Not able to extract Lustre attributes
    - Stripe information
    - HSM status
    - Link EA
- ▶ **LiPE is a policy engine which knows Lustre well**
  - Optimized for Lustre: scanning Ldiskfs device directly
  - Understands Lustre: is able to extract all Lustre attributes saved on disk
  - Powerful & flexible

## Advantages of LiPE

- ▶ **Scans MDTs directly**
  - No need to add extra server or storage
  - No need to duplicate the metadata
  - No need to sync based on Lustre Changelog
  - No need to do initial scan for data injection
  - Quick scanning because MDTs are usually based on SSDs
  - Can scale up with DNE by adding more MDTs and MDS
- ▶ **Precise definition of rules**
  - Precise arithmetic expressions are used in rules to define the exact behavior of policies
    - Avoid vagueness that causes misunderstanding
    - Avoid subtle changes of semantic implication between versions
- ▶ **Easy to setup and configure**
  - Only one user-space RPM is needed
  - Web-based GUI is provided to help administrators to:
    - Configure policy rules with tips and correctness checking
    - Choose apply policies
    - Get graph-format reports
- ▶ **Multiple Lustre file systems for different purposes can be managed together in a single LiPE system**

# Framework of LiPE



## Design of LiPE (1)

### ► Expression of rule

- **Expression:** A arithmetic expression in the form of Polish notation that has a value of unsigned 64-bit integer
- **Operators**
  - Arithmetic operators: +, -, \*, /, %
  - Relational and logical operators (==, !=, >, >=, <, <=)
  - Bitwise operators: &, |, ^, <<, >>
- **Unsigned 64-bit integers** could be used in the rules.
- **Constants** could be used in the rules, e.g. the Lustre internal constants
- **System attributes** could be used in the rules, e.g. date time, free inode number, free disk space, etc.
- **Object attributes** could be used in the rules, e.g. atime, mtime, ctime, size, mode, uid, gid, blocks, type, flags, nlink, rdev, blksize, hsm stat, etc.
- **Functions**
  - fname(\$ARG)emarf: Whether the dentry name matches with regular expression \$ARG
  - ost(\$ARG)tso : Whether the file locates on OST(s) with regular expression \$ARG
  - pool(\$ARG)loop: Whether the file locates on OST pool(s) with regular expression \$ARG
  - ...

### ► Example

- The inode should be **regular file** that was **accessed** earlier than **one year ago**
- && == type S\_IFREG < atime - sys\_time 31536000000

## Design of LiPE (2)

### ► Action of rule

- Counter increase action: LAT\_COUNTER\_INC
- File removal action: LAT\_COUNTER\_REMOVE
- HSM actions : LAT\_HSMA\_\*
- Classification based on UID/GID/HSM state: LAT\_COUNTER\_CLASSIFY
- Project quota action: LAT\_SET\_PROJID
- Ladvise actions: LAT\_LADVISE\_\*
- Set the inode immutable: LAT\_IMMUTABLE

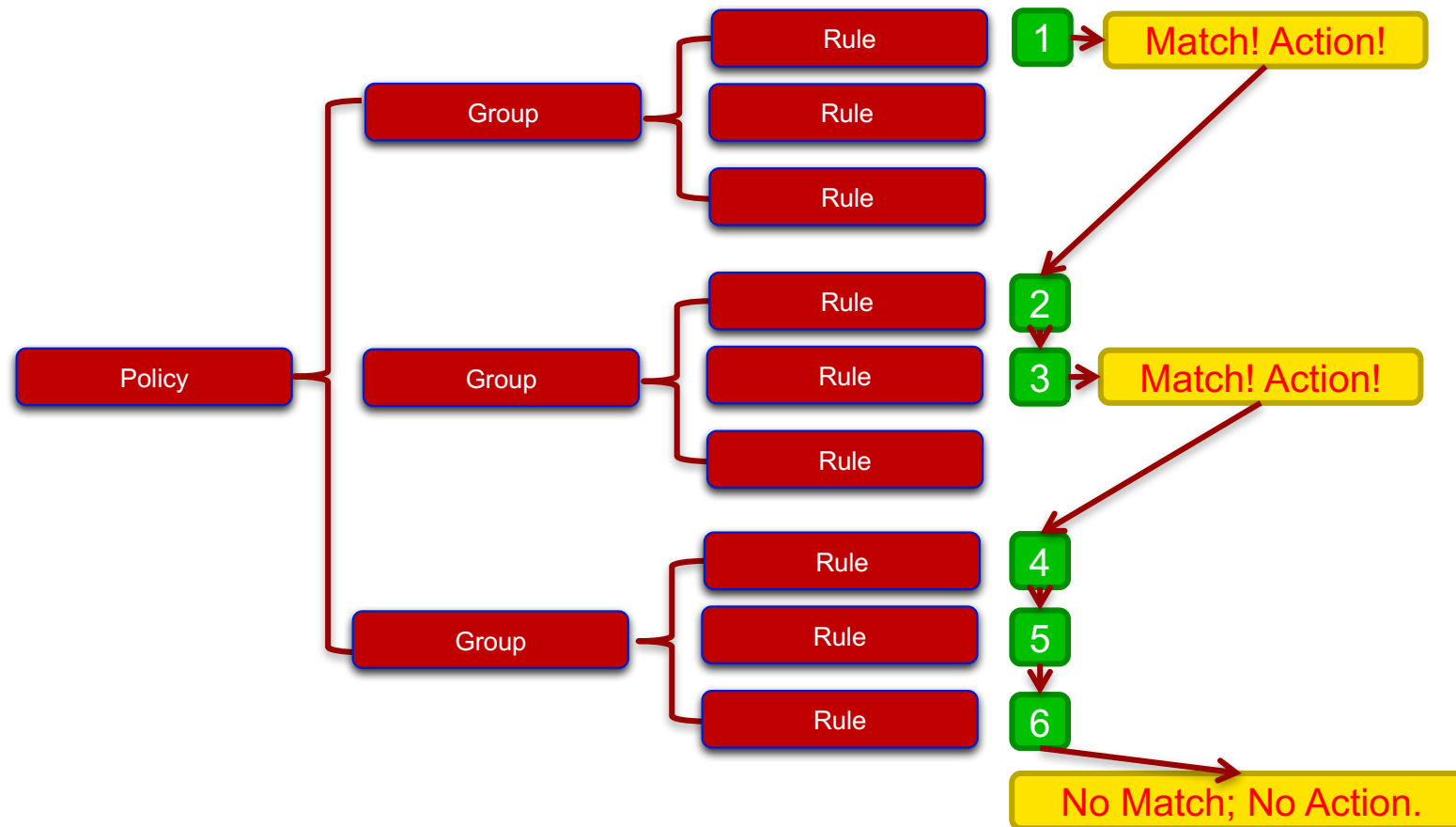
### ► Rule group

- One or more rules could be gathered as a rule group with order
- If an rule in a group is evaluated as matched, the rest rules in that group won't be matched

### ► Multiple rule groups could be defined

- A group to summarize size distribution
- A group to summarize access time distribution
- A group to summarize trigger HSM actions
- A group to summarize trigger backup actions
- ...

## Evaluation of rules in LiPE



## LiPE components

### ▶ **lipe\_web**

- “lipe\_web” is a web-base GUI for administrators to configure rules, run policies and get reports

### ▶ **lipe**

- “lipe” is a tool that scans the MDT and check whether the objects match rule groups
- FID lists of matched files will be printed for rule groups

### ▶ **lipe\_flist\_handle**

- “lipe\_flist\_handle” is a tool that carries out the actions on the FID lists printed by “lipe” command

### ▶ **lipe\_launch**

- “lipe\_launch” is a tool that launches the whole process of scanning of MDTs, applying of actions on the file lists, and generating reports
- “lipe\_launch” can be configured in “crond” to schedule repetitive LiPE tasks

### ▶ **lipe\_agent\_manager**

- “lipe\_agent\_manager” is a tool that manages the HSM copytools

### ▶ **HSM tools: “lipe\_hsm\_remover”, “lipe\_hsm\_check”, etc.**



## GUI of LiPE (1)

The screenshot displays the LiPE (Lustre In-Place Editor) GUI. On the left, a tree view under 'Rule Groups' shows the 'HSM' group expanded, listing various file attributes like 'Size\_Distribution', 'Type\_Distribution', and 'HSM\_State'. On the right, a table lists rules. Two rules are visible, both using the 'LAT\_HSMA\_ARCHIVE' action. Callout boxes identify the 'Rule Group' (HSM), the 'Rule' (the condition part of the rule), and the 'Action' (LAT\_HSMA\_ARCHIVE).

Rule	Action	Count
fname(old_file.+)emanf	LAT_HSMA_ARCHIVE	1
&& == type S_IFREG < atime - sys_time 31536000000	LAT_HSMA_ARCHIVE	2

## GUI of LiPE (2)

The screenshot shows the LiPE GUI interface. On the left is a **Navigation** pane with a tree view of Rule Groups. The 'HSM' group is expanded, showing various sub-items like 'fname(old\_file.+).emarf', 'Size\_Distribution', 'Type\_Distribution', etc. The main area on the right contains a **Control Table** with buttons for navigation and editing, and an **Expression Editor** table. The Expression Editor table has columns for 'Dentry Name', 'Operator', and 'Number'. It currently displays 'old\_file.+' under Dentry Name, 'None' under Operator, and '0' under Number. Below the table are buttons for 'Edit Left Expression', 'Upper', and 'Edit Right Expression'. A large orange bar at the bottom of the main area displays the current expression: 'fname(old\_file.+).emarf'. A **Tooltip** points to this bar, explaining that the expression is in Polish notation and can be edited directly or via its sub-values and operator.

**LiPE** Lustre Integrated Policy Engine

**Navigation**

- Rule Groups
  - HSM
    - fname(old\_file.+).emarf**
      - && == type S\_IFREG < atime - sys
    - Size\_Distribution
    - Type\_Distribution
    - Access\_Time\_Distribution
    - Modification\_Time\_Distribution
    - Status\_Change\_Time\_Distribution
    - Stripe\_Count\_Distribution
    - Hardlink\_Files
    - Looks\_Like\_Temporary\_Files
    - Location\_on\_OSTs
    - Locate\_on\_Pool0
    - User\_Distribution\_of\_Old\_Files
    - Group\_Distribution\_of\_Old\_Files
    - All\_Inodes
    - HSM\_State
  - Hosts
  - Devices
  - Results

**Control Table**

Dentry Name	Operator	Number
old_file.+	None	0

Edit Left Expression    Upper    Edit Right Expression

**Expression Editor**

fname(old\_file.+).emarf

The current editing expression in the form of Polish notation. It is composed by the left sub-value, right sub-value and the operator between them. You can edit it directly, or change it via updating its sub-values and operator.

**Tooltip**

## GUI of LiPE (3)

The screenshot displays the LiPE (Lustre Integrated Policy Engine) GUI. The interface is divided into a left sidebar and a main content area.

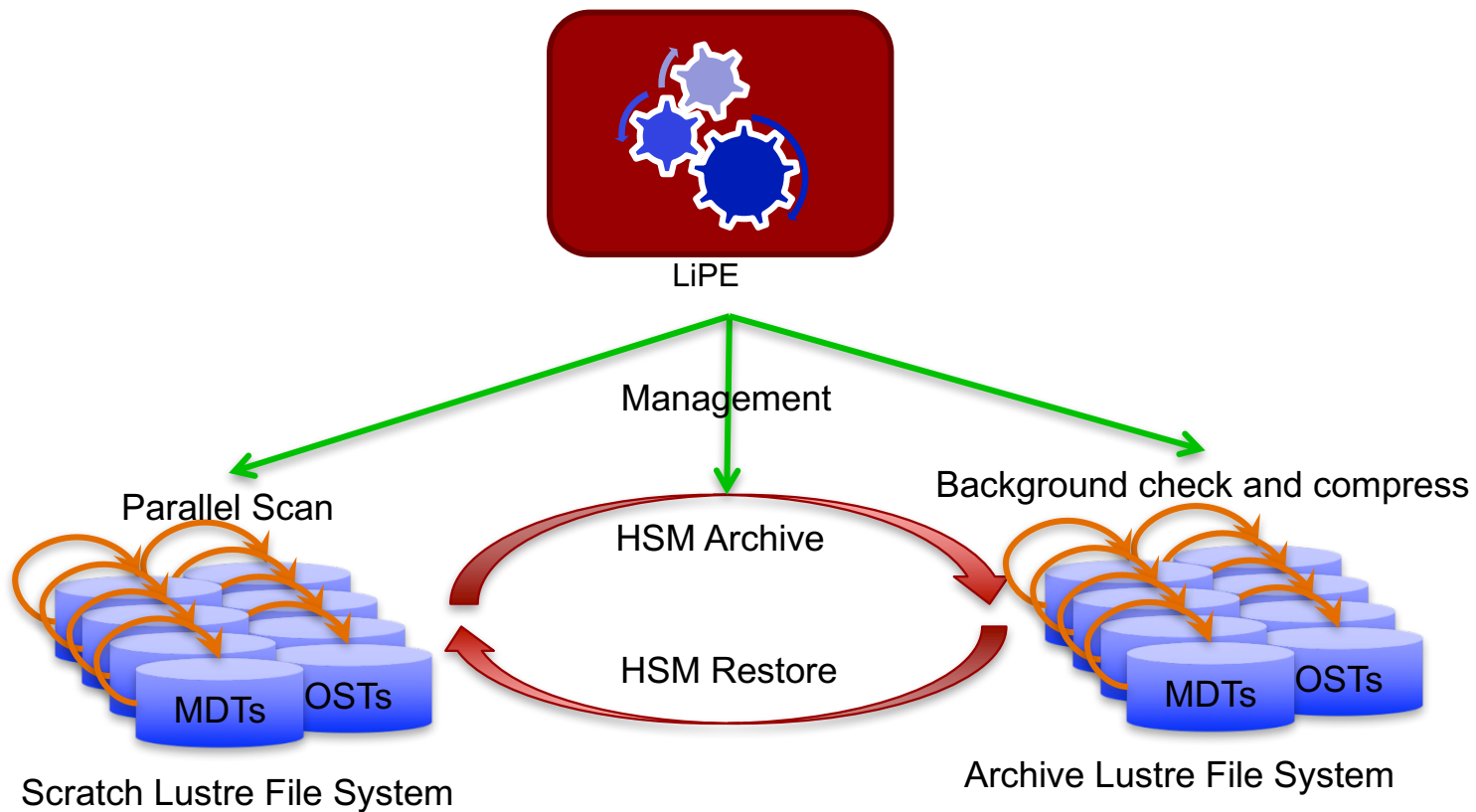
**Left Sidebar:**

- Rule Groups
- Hosts
- Devices
- Results
  - 2017-05-17-17\_43\_33
  - 2017-05-17-17\_44\_06
  - 2017-05-17-19\_18\_56
  - 2017-05-18-10\_52\_28
  - 2017-05-18-10\_52\_45
  - 2017-05-18-12\_10\_40
  - 2017-05-18-12\_23\_18
    - Console Output
    - Config File
    - Result Devices
      - server17\_sda1
        - Result Statistics** (highlighted)
      - server1\_sdb

**Main Content Area:**

- Buttons: Back, Forward, Add, Subtract, Filter, Download, Save, Share, Play.
- Section: Type\_Distribution
- Section: Access\_Time\_Distribution
- Pie Chart showing data distribution:
  - Other: 1(0.02%)
  - Accessed\_more\_than\_1\_year\_ago: 0(0%)
  - Accessed\_more\_than\_1\_month\_less\_than\_1\_year\_ago: 15(0.23%)
  - Accessed\_more\_than\_1\_week\_less\_than\_1\_month\_ag
  - Accessed\_more\_than\_1\_day\_less\_than\_1\_week\_a
  - Accessed\_more\_than\_1\_hour\_less\_than\_1\_day\_
- Dialog Box: Download flist of "Accessed\_more\_than\_1\_minute\_less\_th..."
  - Get file paths (checked)
  - Download
  - Cancel

## Use case (1): Use LiPE to manage HSM



## Use case (2): Use LiPE for file system report

- ▶ A report includes statistical charts of percentages based on either disk usages (implementing) or inode numbers
- ▶ All file lists can be downloaded for further check
- ▶ File lists can be pre-sorted based on UID/GID or size or any other attributes (implementing)
- ▶ A .docx format report can be downloaded (implementing)

## Benchmark results of LiPE

### ► Test environment

- A single SSD based MDT with read speed of about 549 MB/s: Samsung SSD 850
- Intel(R) Xeon(R) CPU E5-2630 v3 @ 2.40GHz

### ► Fully cached scanning speed

- 59 million inode/s
- CPU is the limitation
- 0.76KB memory cache is needed for each inode

### ► None cached scanning speed

- 1.8 million inode/s
- Disk speed is the main limitation of scanning speed
- CPU usage is high but not the bottle neck of performance
- On production system, LiPE should limit its CPU and disk bandwidth usage to avoid impacting Lustre service

## Potentials of LiPE

- ▶ **LiPE + ZFS (Btrfs, etc.)**
  - Efficient scanning tools needs to be developed for new types of OSD
- ▶ **LiPE + ladvice**
  - Ladvice is a tool that can give file access advices or hints to Lustre servers
  - LiPE can automatically generate advices to be sent according to pre-defined policies
- ▶ **LiPE + Loris**
  - Loris: Lustre Online Reliability Improvement System
  - Loris backups MDTs to external storage for disaster recovery
  - LiPE can scan the MDT mirrors of Loris instead of MDTs to avoid metadata performance impact
  - Performance of scanning MDT mirrors is almost the same with scanning MDTs if storage is the same type
- ▶ **LiPE + L2RC**
  - L2RC: Lustre Level 2 Read Cache, a OSD level read cache for Lustre
  - Use the LiPE to manage the cache readahead of L2RC
- ▶ **LiPE + File Heat**
  - File heat: a value that reflects the access frequency of the objects
  - Scanning OSTs would be more useful if LiPE can apply rules based on the file heat values of OST objects
- ▶ **LiPE + MpiFileutils**
  - MpiFileutils: a suite of MPI-based tools to manage large datasets
  - LiPE could use MpiFileutils to scan all kinds of Posix file system

## Conclusion

- ▶ **Implemented a new policy engine for Lustre: LiPE**
- ▶ **LiPE scans MDTs/OSTs directly and requires no external storage**
- ▶ **LiPE has quick scanning speed**
- ▶ **LiPE has a lot of use case potentials, including HSM**



17

**Thank you!**

