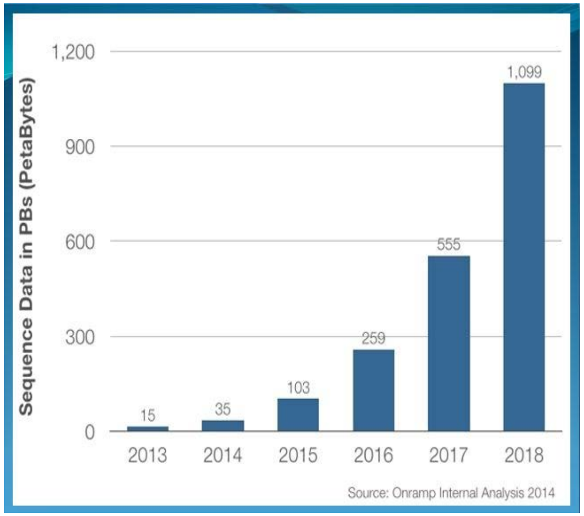
Toward Scalable Monitoring on Large-Scale Storage for Software Defined Cyberinfrastructure

Arnab K. Paul[†], Ryan Chard[‡], Kyle Chard[⋆], Steven Tuecke[⋆], Ali R. Butt[†], Ian Foster[‡ ⋆]

[†]Virginia Tech, [‡]Argonne National Laboratory,
[⋆]University of Chicago

Motivation



Data generation rates are exploding

Complex analysis processes

The data lifecycle often involves multiple organizations, machines, and people

Motivation

This creates a significant strain on researchers

- Best management practices (cataloguing, sharing, purging, etc.) can be overlooked.

- Useful data may be lost, siloed, and forgotten.

Software Defined Cyberinfrastructure (SDCI)

Accelerate discovery by automating research processes, such as data placement, feature extraction, and transformation.

Enhance reliability, security, and transparency by integrating secure auditing and access control mechanisms into workflows.

Enable data sharing and collaboration by streamlining processes to catalog, transfer, and replicate data.

Background: RIPPLE

RIPPLE: A prototype responsive storage solution

Transform static data graveyards into active, responsive storage devices

- Automate data management processes and enforce best practices
- Event-driven: actions are performed in response to data events
- Users define simple if-trigger-then-action recipes
- Combine recipes into flows that control end-to-end data transformations
- Passively waits for filesystem events (very little overhead)
- Filesystem agnostic – works on both edge and leadership platforms

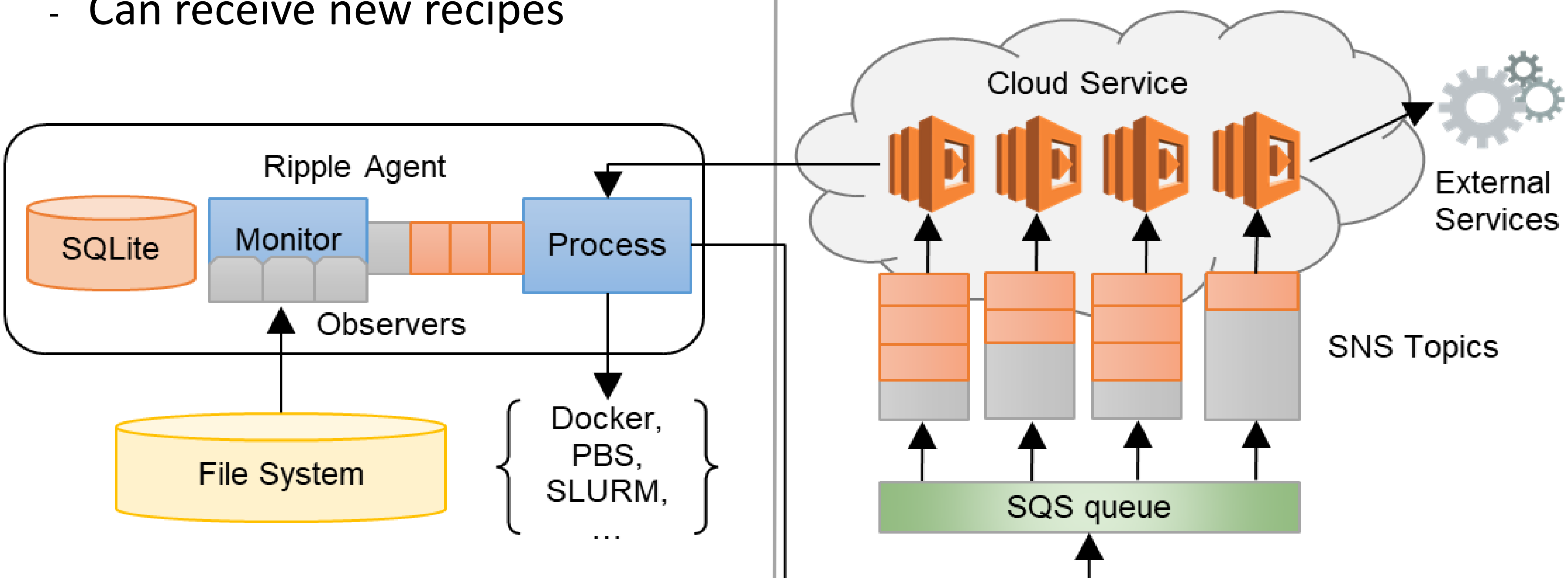
RIPPLE Architecture

Agent:

- Sits locally on the machine
- Detects & filters filesystem events
- Facilitates execution of actions
- Can receive new recipes

Service:

- Serverless architecture
- Lambda functions process events
- Orchestrates execution of actions



RIPPLE Recipes

IFTTT-inspired programming model:

Triggers describe where the event is coming from (filesystem create events) and the conditions to match (/path/to/monitor/.*.h5)

Actions describe what service to use (e.g., globus transfer) and arguments for processing (source/dest endpoints).

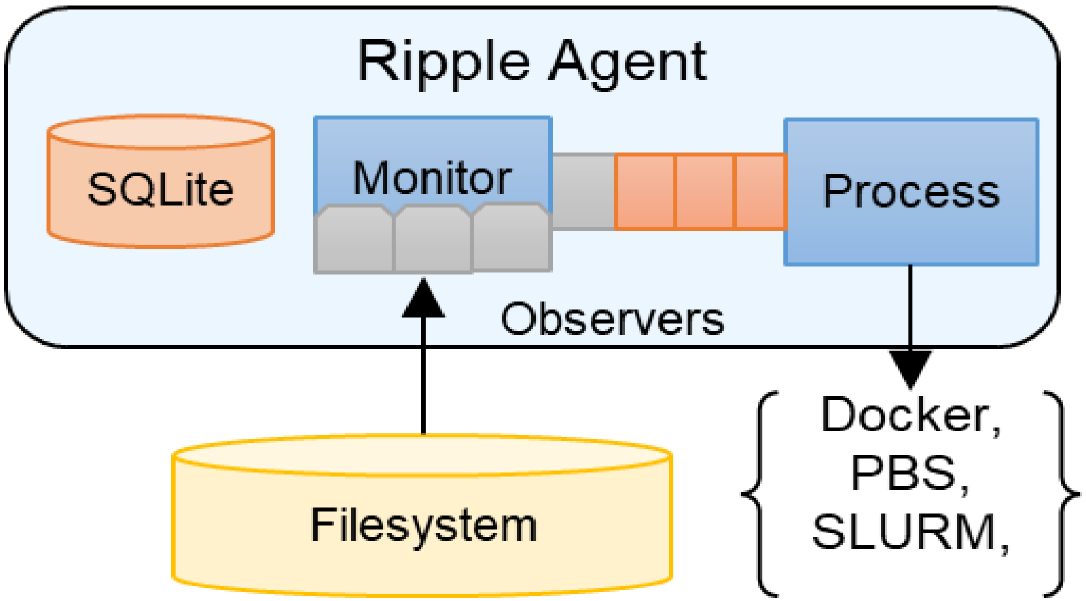
```
"recipe":{
  "trigger": {
    "username": "ryan",
    "monitor": "filesystem",
    "event": "FileCreatedEvent",
    "directory": "/path/to/monitor/",
    "filename": ".*.h5$"
  },
  "action": {
    "service": "globus",
    "type": "transfer"
    "source_ep": "endpoint1",
    "dest_ep": "endpoint2",
    "target_name": "$filename",
    "target_match": "",
    "target_replace": "",
    "target_path": "/~/$filename.h5",
    "task": "",
    "access_token": "<access token>"
  }
}
```

RIPPLE Agent

Python Watchdog observers listen for events

- *inotify*, polling, for filesystem events (create, delete, etc.)

Recipes are stored locally in a SQLite database



Limitations

- Inability to be applied at scale
- Approach primarily relies on targeted monitoring techniques
  - *inotify* has a large setup cost
  - time consuming and resource intensive
- Crawling and recording file system data is prohibitively expensive over large storage systems.

Scalable Monitoring

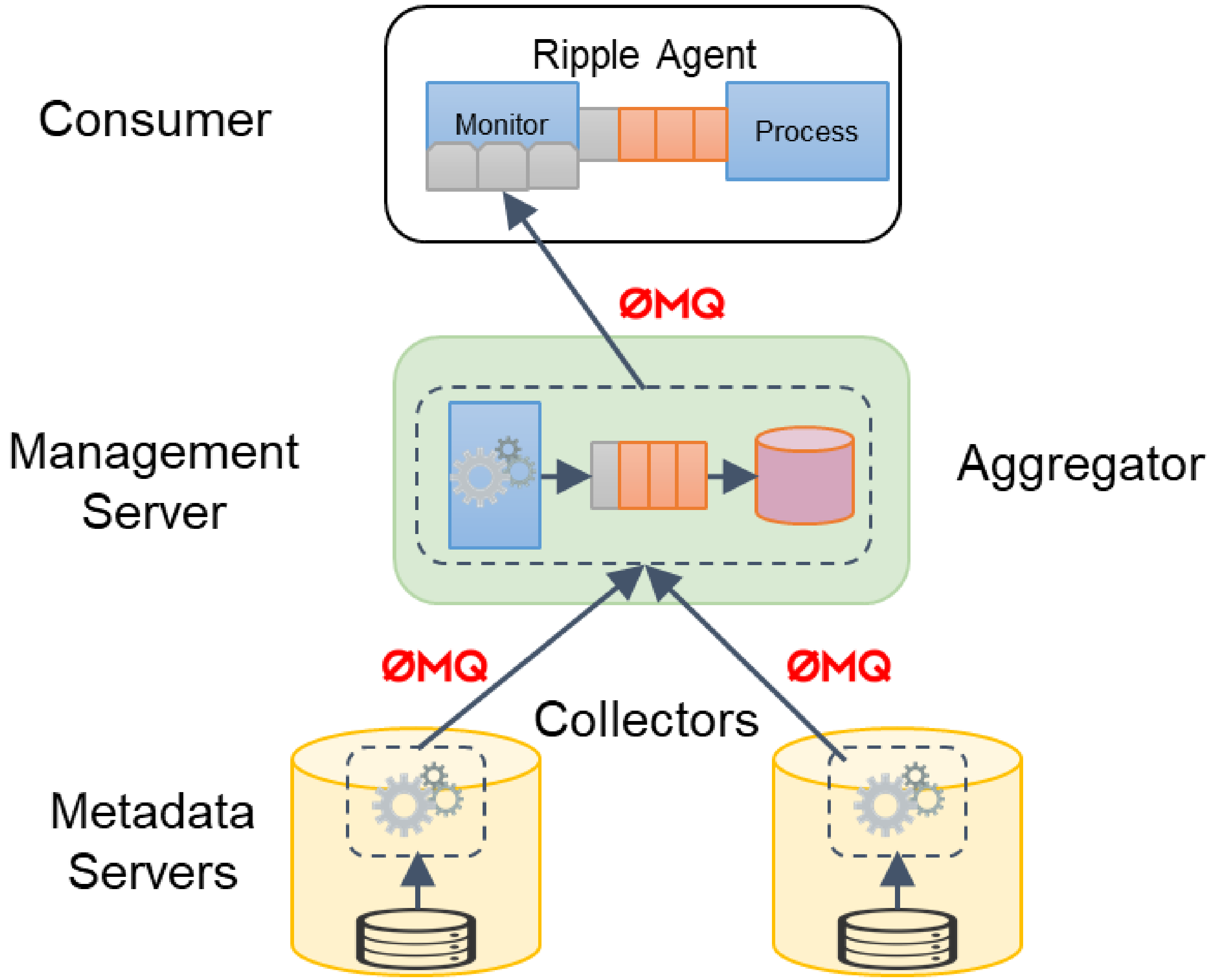
- Uses Lustre’s internal metadata catalog to detect events.
- Aggregate the events and stream those to any subscribed device.
- Provides fault tolerance.

Lustre Changelog

Event ID	Type	Timestamp	Datestamp	Flags	Target FID	Parent FID	Target Name
13106	01CREAT	20:15:37.1138	2017.09.06	0x0	t=[0x200000402:0xa046:0x0]	p=[0x200000007:0x1:0x0]	data1.txt
13107	02MKDIR	20:15:37.5097	2017.09.06	0x0	t=[0x200000420:0x3:0x0]	p=[0x61b4:0xca2c7dde:0x0]	DataDir
13108	06UNLNK	20:15:37.8869	2017.09.06	0x1	t=[0x200000402:0xa048:0x0]	p=[0x200000007:0x1:0x0]	data1.txt

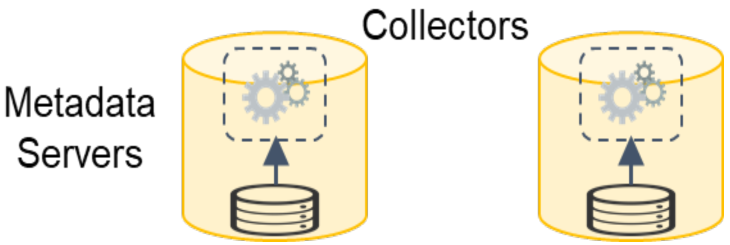
- Sample changelog entries
- Distributed across Metadata Servers (MDS)
- Monitor all MDSs

Monitoring Architecture



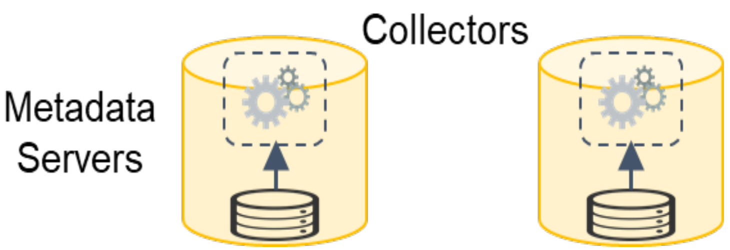
Monitoring Architecture (contd.)

- **Detection**
  - Collectors on every MDS
  - Events are extracted from the changelog.



Monitoring Architecture (contd.)

- **Detection**
  - Collectors on every MDS
  - Events are extracted from the changelog.

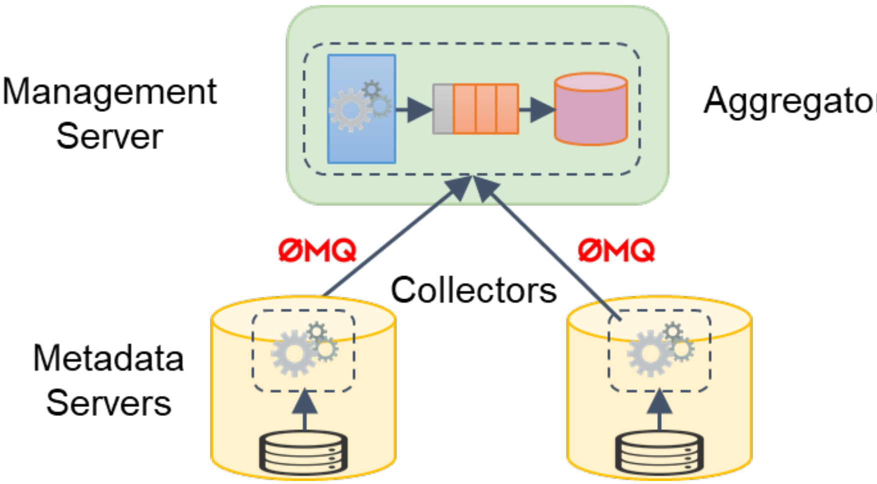


- **Processing**
  - Parent and target file identifiers (FIDs) are not useful to external services.
  - Collector uses Lustre *fid2path* tool to resolve FIDs and establish absolute path names.

Event ID	Type	Timestamp	Datestamp	Flags	Target FID	Parent FID	Target Name
13106	01CREAT	20:15:37.1138	2017.09.06	0x0	t=[0x200000402:0xa046:0x0]	p=[0x200000007:0x1:0x0]	data1.txt
13107	02MKDIR	20:15:37.5097	2017.09.06	0x0	t=[0x200000420:0x3:0x0]	p=[0x61b4:0xca2c7dde:0x0]	DataDir
13108	06UNLNK	20:15:37.8869	2017.09.06	0x1	t=[0x200000402:0xa048:0x0]	p=[0x200000007:0x1:0x0]	data1.txt

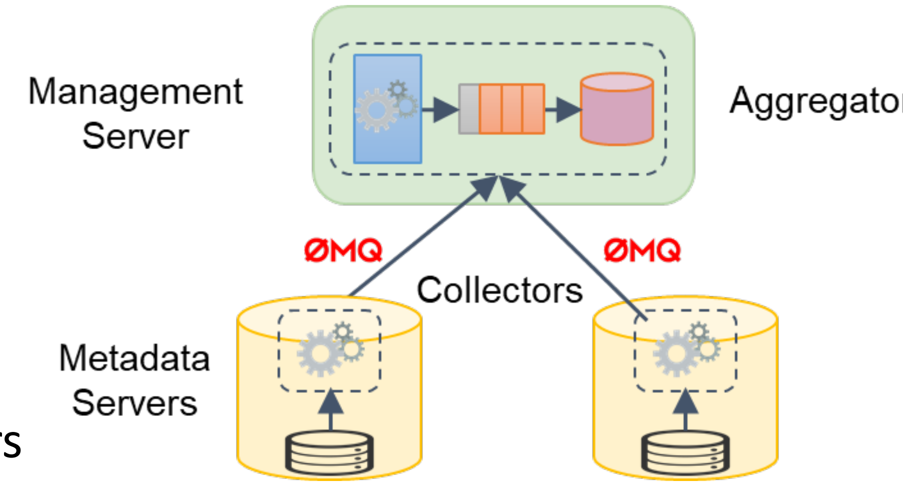
Monitoring Architecture (contd.)

- **Aggregation**
  - ZeroMQ used to pass messages.
  - Multi-threaded:
    - Publish events to consumers
    - Store events in local database for fault tolerance



Monitoring Architecture (contd.)

- **Aggregation**
  - ZeroMQ used to pass messages.
  - Multi-threaded:
    - Publish events to consumers
    - Store events in local database for fault tolerance



- **Purging Changelog**
  - Collectors purge already processed changelog events to lessen the burden in MDS.

Evaluation

Testbeds

- **AWS**
  - 5 Amazon AWS EC2 instance
  - 20 GB Lustre file system
  - Lustre Intel Cloud Edition 1.4
  - t2.micro instances
    - 2 compute nodes
    - 1 OSS, 1 MGS, and 1 MDS

Evaluation

Testbeds

- **IOTA**
  - Argonne National Laboratory's Iota cluster
  - 44 compute nodes
    - 72 cores
    - 128 GB memory
  - 897 TB Lustre Store ~ 150 PB for Aurora

Testbed Performance

	AWS	IOTA
Storage Size	20GB	897TB
Files Created (events/s)	352	1389
Files Modified (events/s)	534	2538
Files Deleted (events/s)	832	3442
Total Events (events/s)	1366	9593

Event Throughput

	AWS	IOTA
Storage Size	20GB	897TB
Files Created (events/s)	352	1389
Files Modified (events/s)	534	2538
Files Deleted (events/s)	832	3442
Total Events (events/s)	1366	9593

- **AWS**
  - Report 1053 events per second to the consumer.
- **IOTA**
  - Report 8162 events/s

Monitor Overhead

	CPU (%)	Memory (MB)
Collector	6.667	281.6
Aggregator	0.059	217.6
Consumer	0.02	12.8

Maximum Monitor Resource Utilization

Scaling Performance

- Analyzed NERSC's production 7.1PB GPFS file system
  - Over 16000 users and 850 million files
- 36-day file system dumps.
- Peak of 3.6 million differences between two days
  - ~ 127 events/s
- Extrapolate to 150PB store for Aurora
  - ~ 3178 events/s

Conclusion

- SDCI can resolve many of the challenges associated with routine data management processes.

- RIPPLE enabled such automation but was not often available on large-scale storage systems.

- Scalable Lustre monitor addresses this shortcoming.

- Lustre monitor is able to detect, process, and report events at a rate sufficient for Aurora.

Thank you! Q&A

akpaul@vt.edu

http://research.cs.vt.edu/dssl/