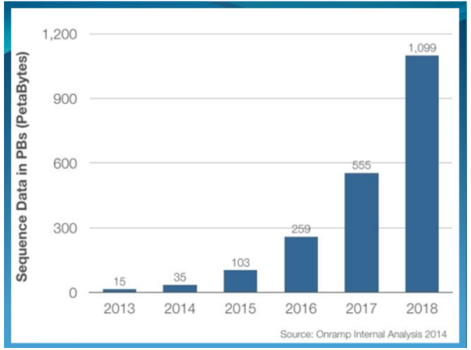
Responsive Storage:
Home Automation for
Research Data Management

Ryan Chard, Kyle Chard, Jason Alt, Dilworth Y. Parkinson, Steve Tuecke, and Ian Foster

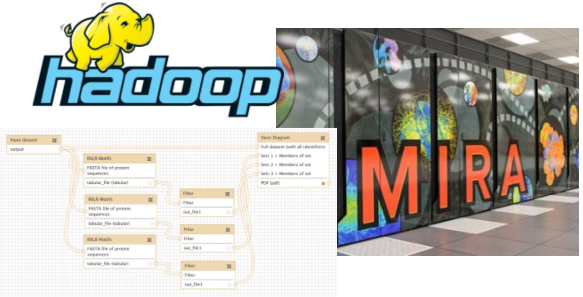
Argonne National Lab, University of Chicago, and Lawrence Berkeley National Lab

The Problem

- Data generation rates are exploding
- Complex analytics processes
- The data lifecycle often involves multiple organisations, machines, and people



Source: Onramp Internal Analysis 2014



This creates a significant strain on researchers

- Best management practices (cataloguing, sharing, purging, etc.) can be overlooked
- Useful data may be lost, siloed, or forgotten

RIPPLE: A prototype responsive storage solution

Transform static data graveyards into active, responsive storage devices

- Automate data management processes and enforce best practices
- Reliable event-driven execution
- Users focused: simple if-trigger-then-action rules
  - Accessible to all end users, not just admins and expert users
  - Users can set data management policies and then forget about them
- Combine rules into flows to control end-to-end data transformations
- Passively waits for filesystem events (very little overhead)
- Filesystem agnostic – works on both edge and leadership platforms

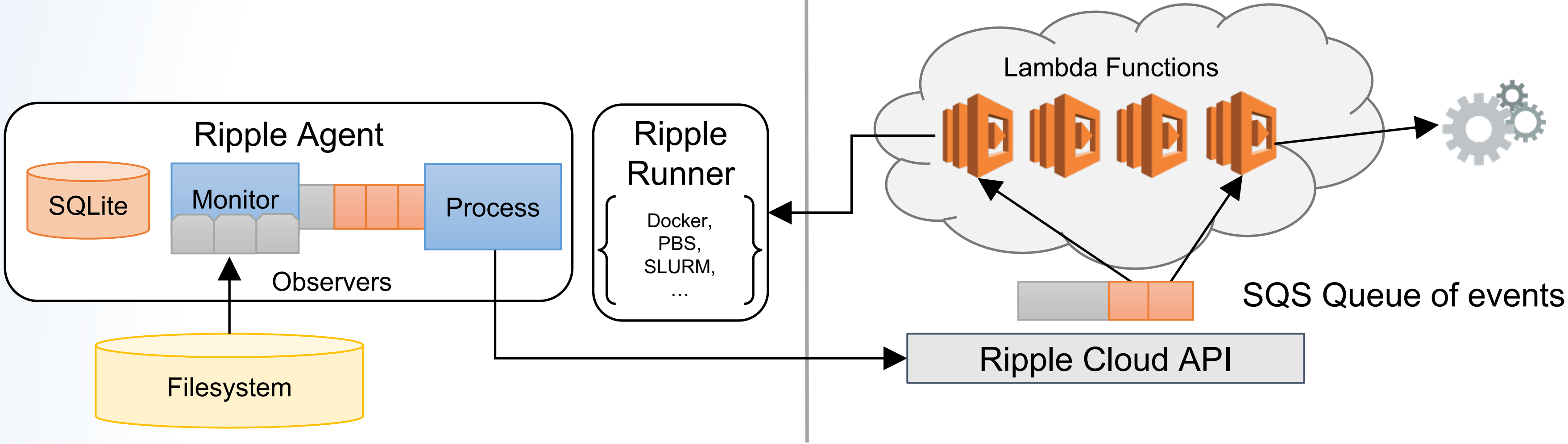
RIPPLE Architecture (updated)

Agent:

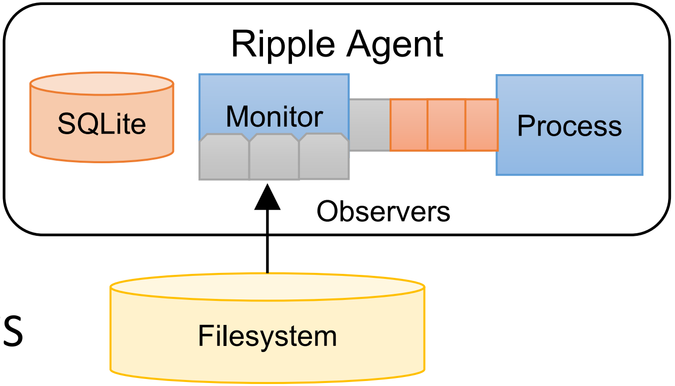
- Sits locally on the machine
- Detects & filters filesystem events
- Facilitates execution of actions
- Can receive new recipes

Service:

- Serverless architecture
- Lambda functions process events
- Orchestrates execution of actions
- Records and manages execution of flows



RIPPLE Agent



Responsible for detecting and reporting events of interest

Filesystem agnostic – uses an appropriate monitor for the FS

- Leverages Python Watchdog observers
  - inotify, polling, kqueue, etc.
- Globus Transfer API detects globus events (transfer, create, delete)

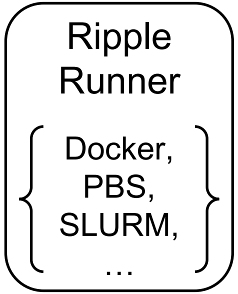
Rules are retrieved from the cloud service and stored in an SQLite database

Hybrid filtering model:

- Local monitor checks events against active rules
- If they match, they are reported to the cloud for processing

RIPPLE Runner

Abstracts execution environments and allows job submission/status checks via API



Ripple Runner

Docker, PBS, SLURM, …

Has a UUID and polls for actions – rules can invoke actions on any runner

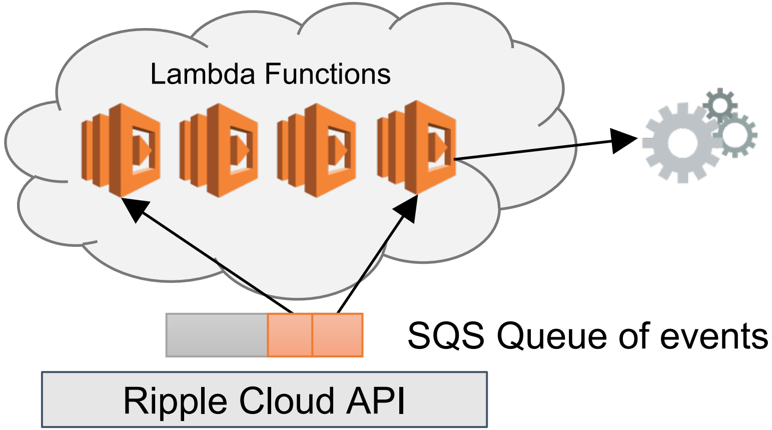
Can be deployed almost anywhere:

- Locally initiate Docker containers, singularity exec commands, and subprocesses to act on local files (metadata extraction, dispatch jobs, etc.)

- Cloud runner (backed by Lambda functions) performs cloud functions: Globus transfers, create shared endpoints, send emails, invoke other Lambda functions etc. This functions an API gateway exposing the runner API and proxying requests through to Lambdas

- HPC systems employ a runner for exposed batch submission (currently just SLURM)

RIPPLE Cloud Service



- Gateway API exposes Ripple service
  - Get rules
  - Report events
  - Update event status
- API either proxies Lambda functions (get rules) or inserts payload into SQS queue.
- Once an event reaches the SQS, it should not be lost
- SQS queue reports to SNS topic, triggering Lambda functions to pull from the queue
  - Dead letter queue after 3 processing failures
- CloudWatch timer triggers “cleanup” and “checkup” functions to process events still on the queue and outstanding jobs

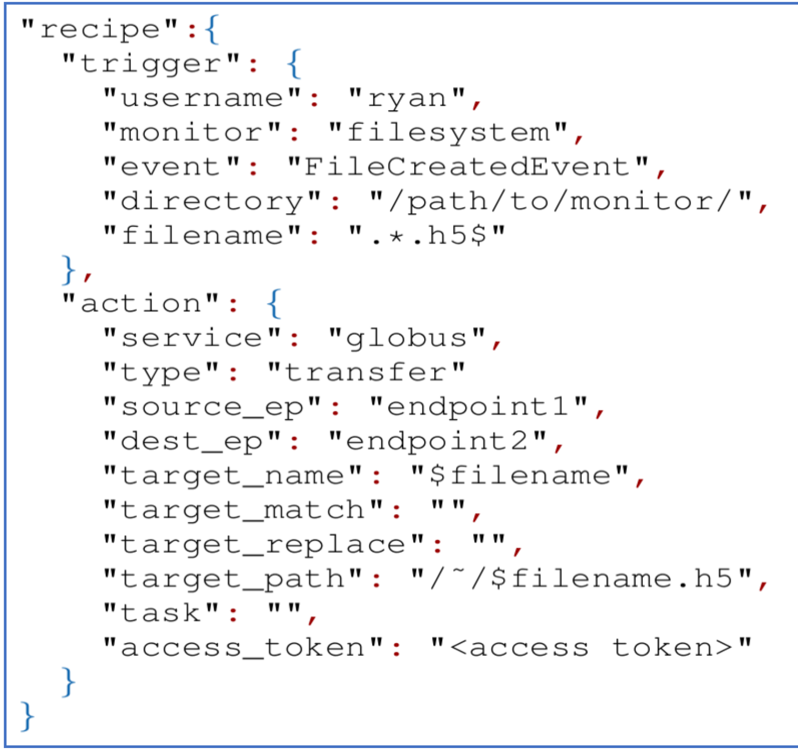
RIPPLE Rules

IFTTT-inspired programming model:

Turn on your lights when you're near home

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>"
  }
}
```

Event Detection

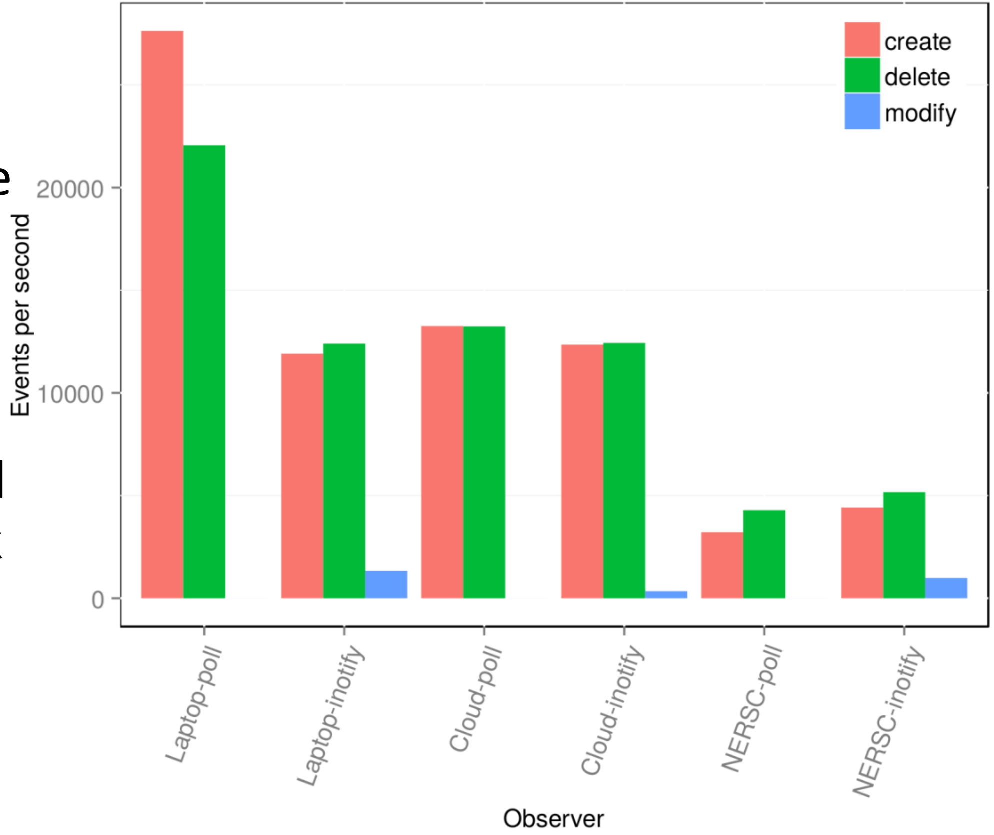
Goal: be able to monitor HPC storage workload (>3 mil events/day)

Inotify vs polling

Create/touch/delete 10,000 files and record event reporting duration (20k total)

Machines:

- Laptop
- c4.xlarge instance
- Edison login node (gpfs)



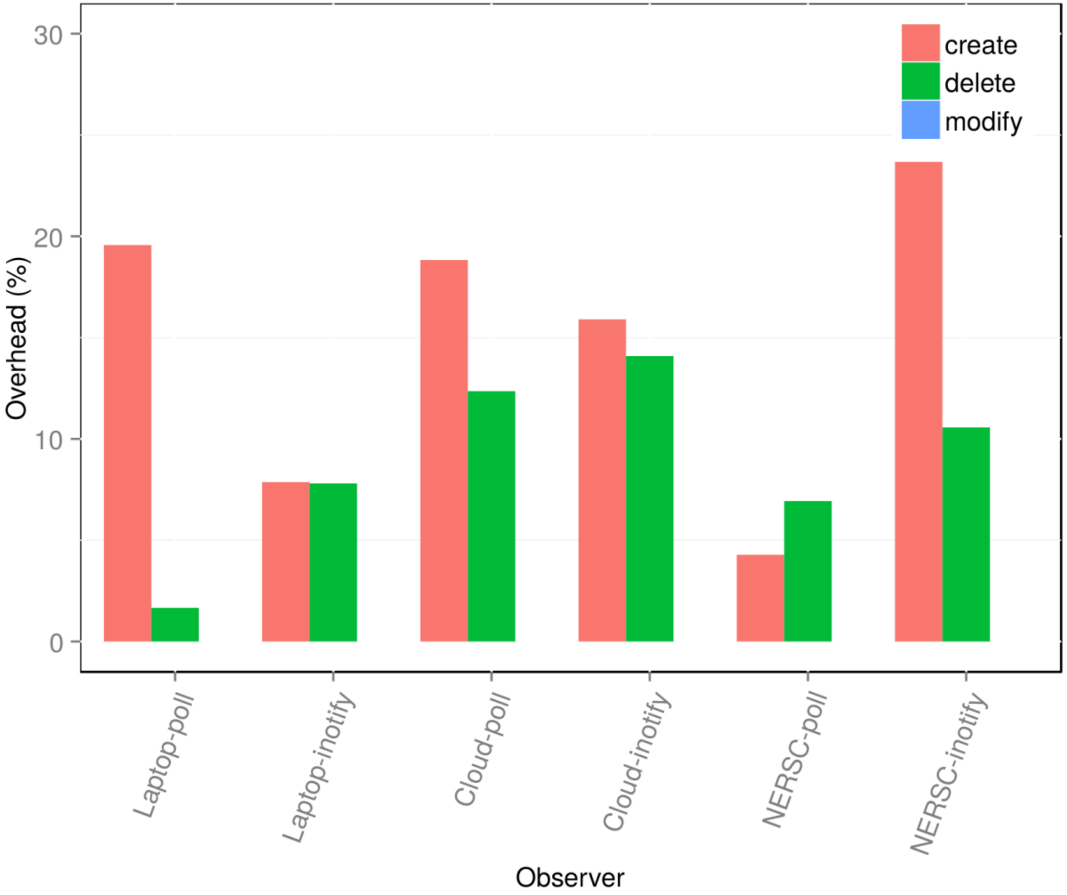
Filtering overhead

Goal: Determine overhead caused by filtering events locally

Measure differences in event/second detection

Filtering requires matching directory path and file extension

Polling is odd as it only polls once every second



Lambda Performance

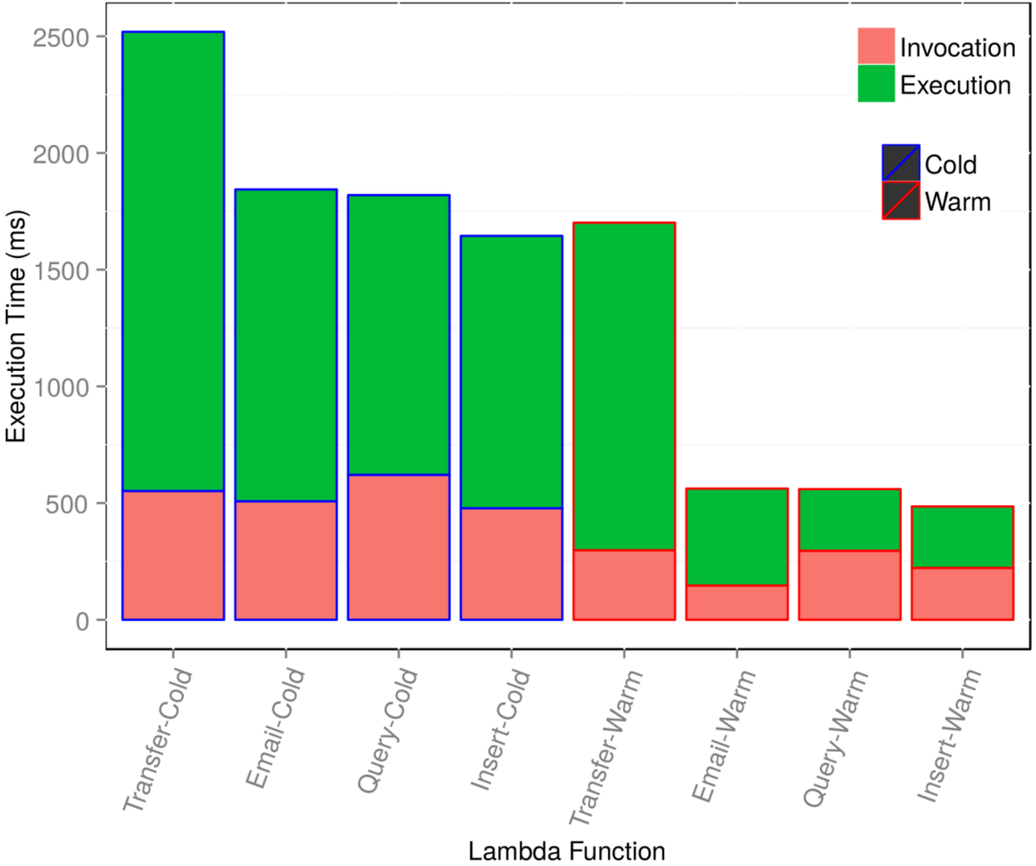
Goal: Understand lambda performance for different tasks

Cold vs Warmed functions

Actions:

- Globus transfer
- SMS email
- DynamoDB insert/query

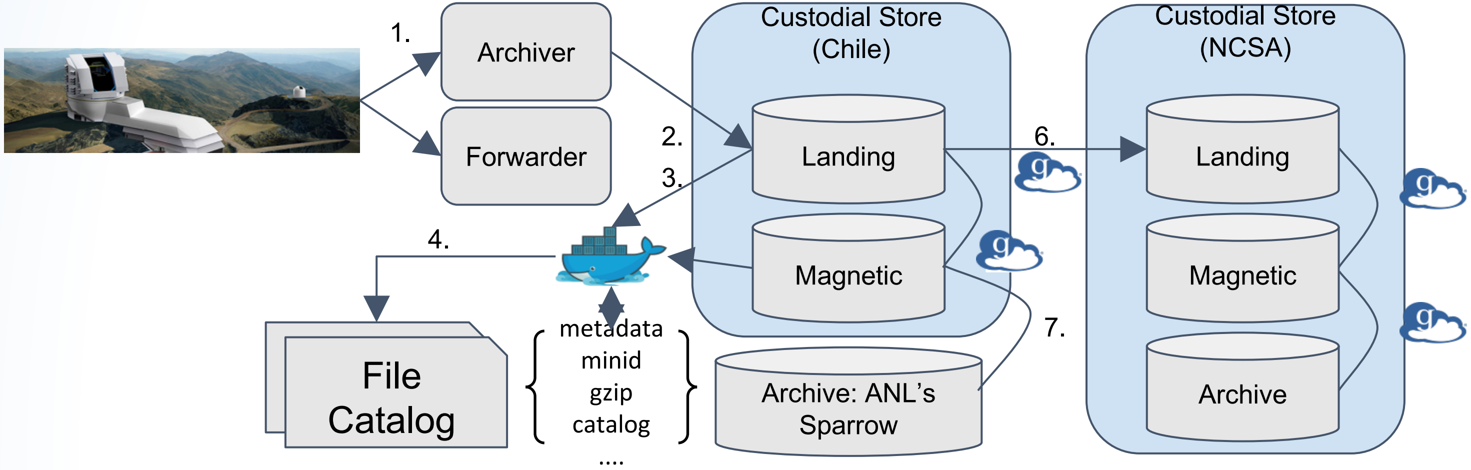
Transfers require a handshake with the Globus service, which also communicates with the endpoints



Use Case: Large Synoptic Survey Telescope

Developed a representative testbed of the LSST storage requirements

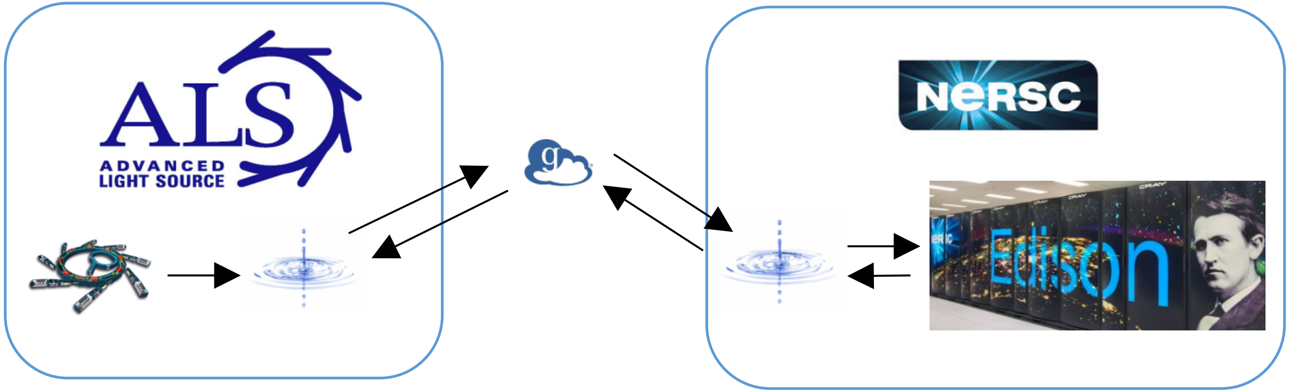
- Automatically propagate data between storage tiers and facilities
- Invoke Docker containers to extract metadata and maintain a file catalog
- Compress and archive files
- Recover deleted/corrupted files when delete and modification events occur



Use Case: Advanced Light Source

Deployed Ripple on an ALS and NERSC machine to automate data analysis

- **At ALS:** Detect new heartbeat beamline data and initiate transfer to NERSC
- **At NERSC:** Extract metadata, create sbatch file, dispatch analysis job to Edison queue, detect result and transfer back to ALS
- **At ALS:** create a shared endpoint, notify collaborators of result via email



New Use Cases

- Automated metadata extraction and ingestion into Globus Search
  - Uses singularity and Apache Tika to extract metadata
  - Metadata is wrapped into gmeta (json) documents and ingested into search
- Offline feedback mechanism for workflows
  - Researchers want a human quality control component
  - Have Ripple send subsets of data to researchers via email to check it
  - Trigger actions based on content of reply messages

Summary

Event-driven automation of data management practices

User focused

Monitoring agent agnostic to underlying filesystem

Serverless event processing and action orchestration

Future Work

More use cases!

More runners

Scalable & high performance event monitors for leadership resources

Programming model for event-based data management

Integration with Globus