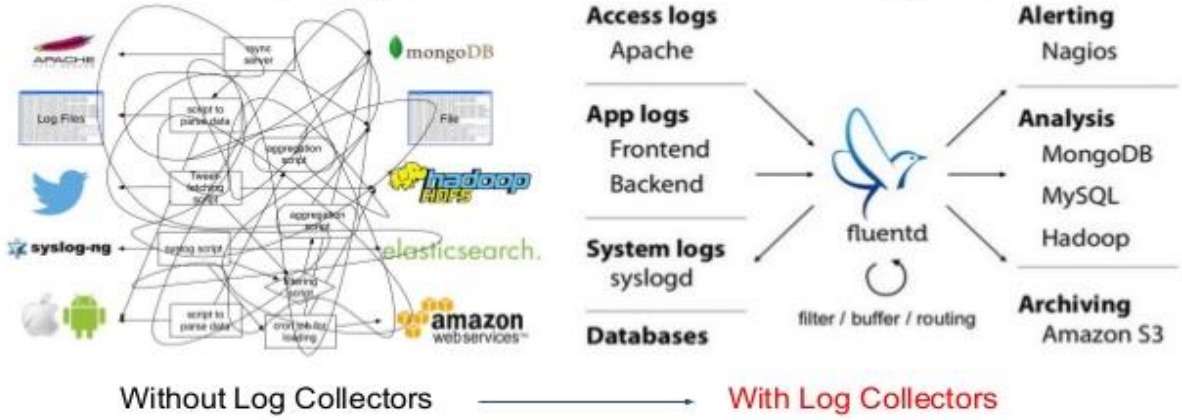
Fluentd- log collection tools

What are Log Collectors?

- Provide pluggable and unified logging layer



Without Log Collectors → With Log Collectors

Images from http://fluentd.org/

3

Two Popular Log Collectors

- Fluentd
  - Written in CRuby
  - Used in Kubernetes
  - Maintained by Treasure Data Inc.
- Logstash
  - Written in JRuby
  - Maintained by elastic.co
- They have similar features
- Which one is better for you?

5

fluentd

Structured logging

Reliable forwarding

http://fluentd.org/

Pluggable architecture

WHAT'S FLUENTD?

AN EXTENSIBLE & RELIABLE DATA COLLECTION TOOL

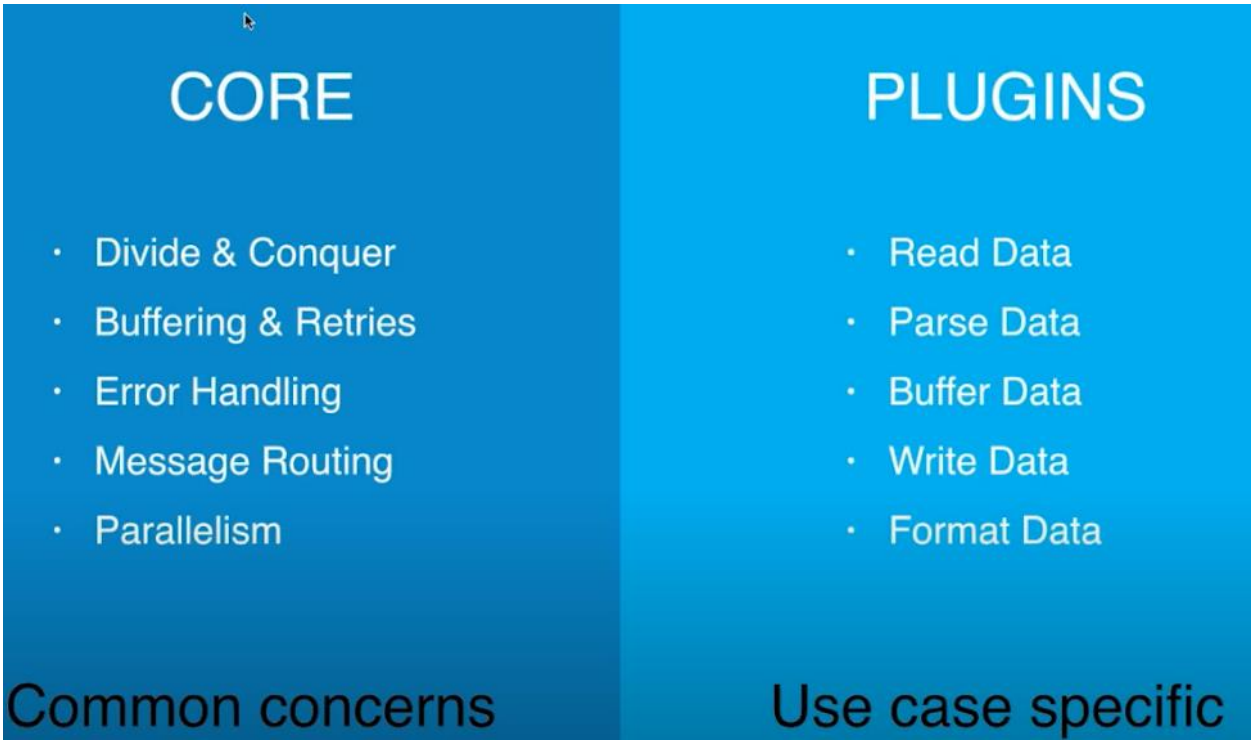
Buffering, HA (failover), load balancing, etc.

Simple core + plugins

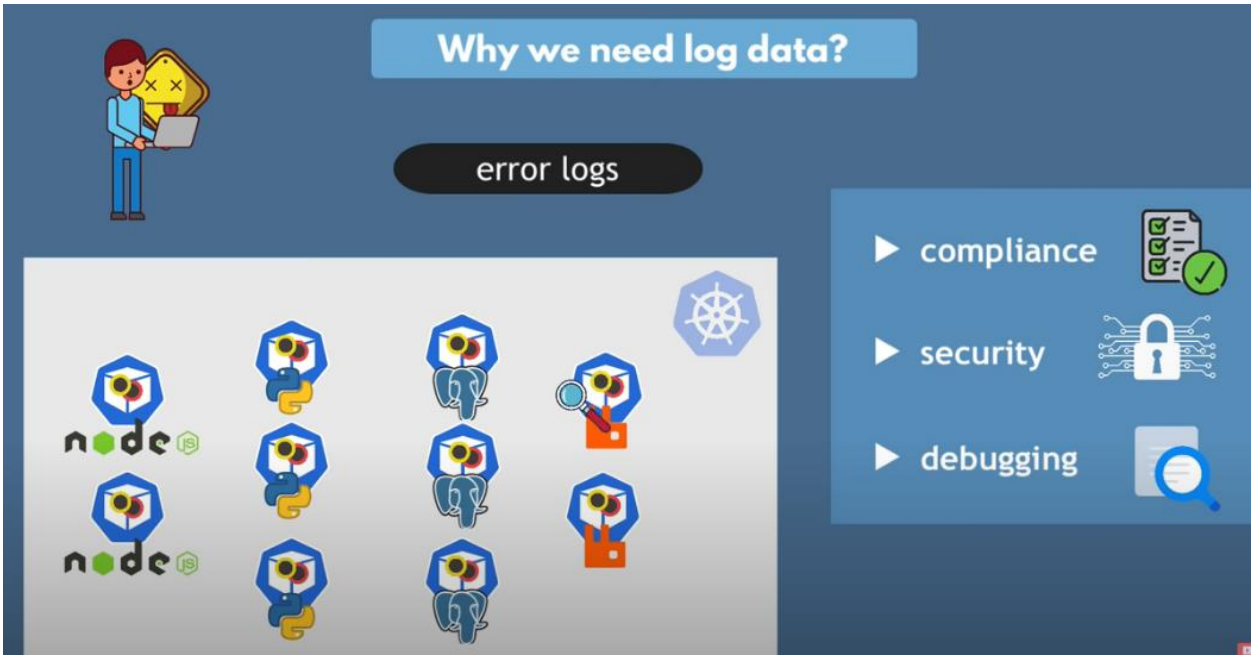
Like syslogd

Reliability (core + plugin)

- Buffering
  - Use file buffer for persistent data
  - buffer chunk has ID for idempotent
- Retrying
- Error handling
  - transaction, failover, etc on forward plugin
  - secondary

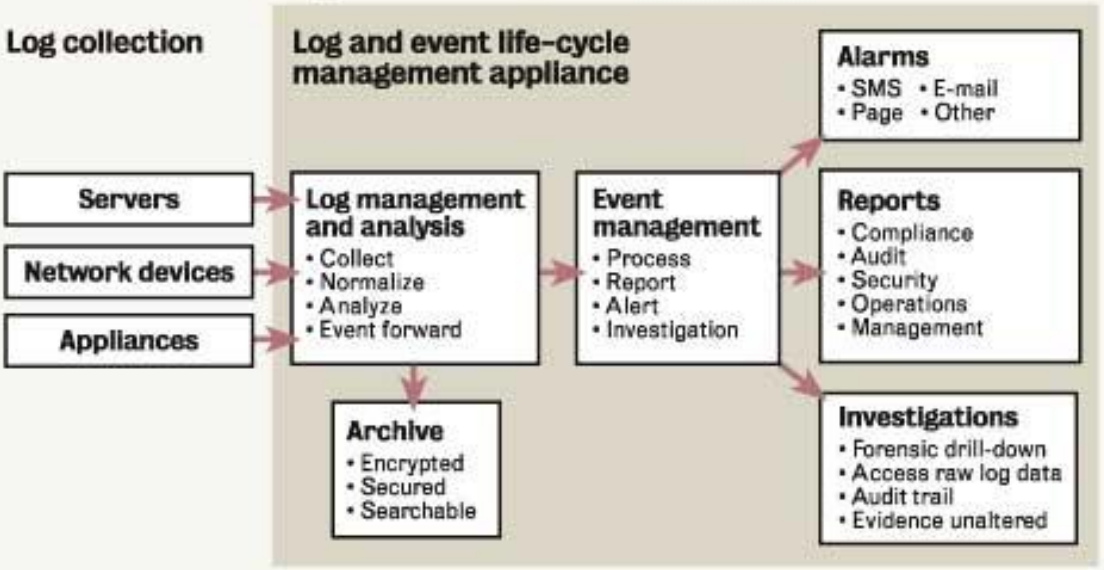


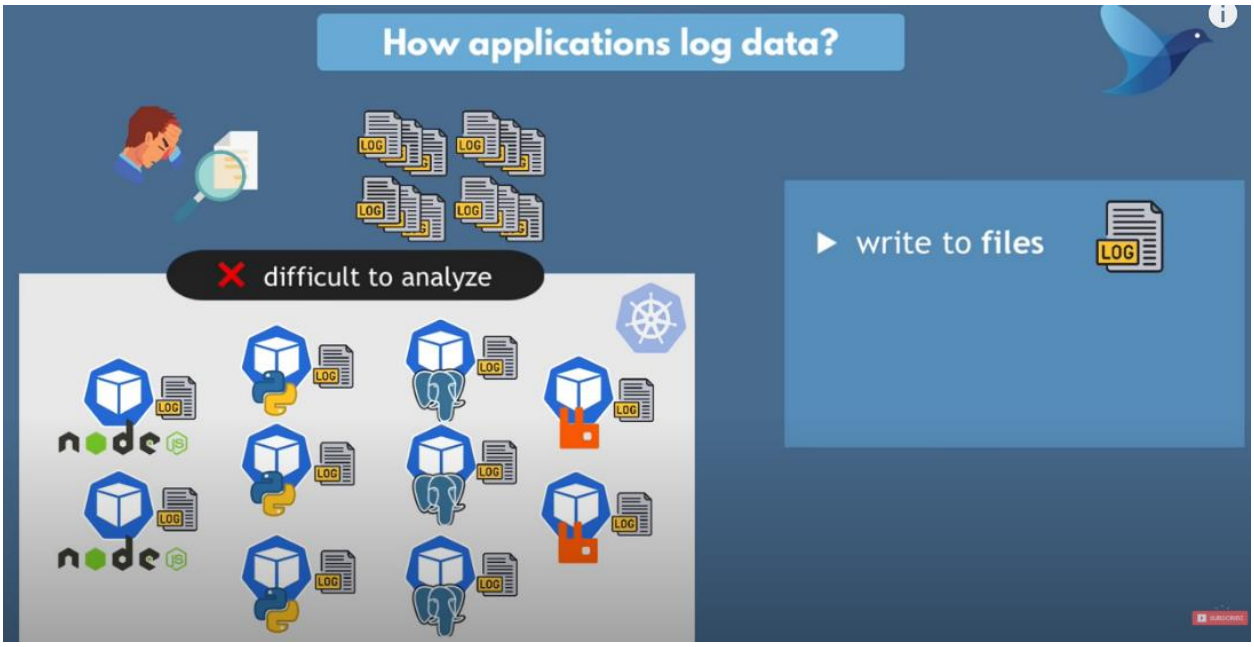
Why we actually need log data?

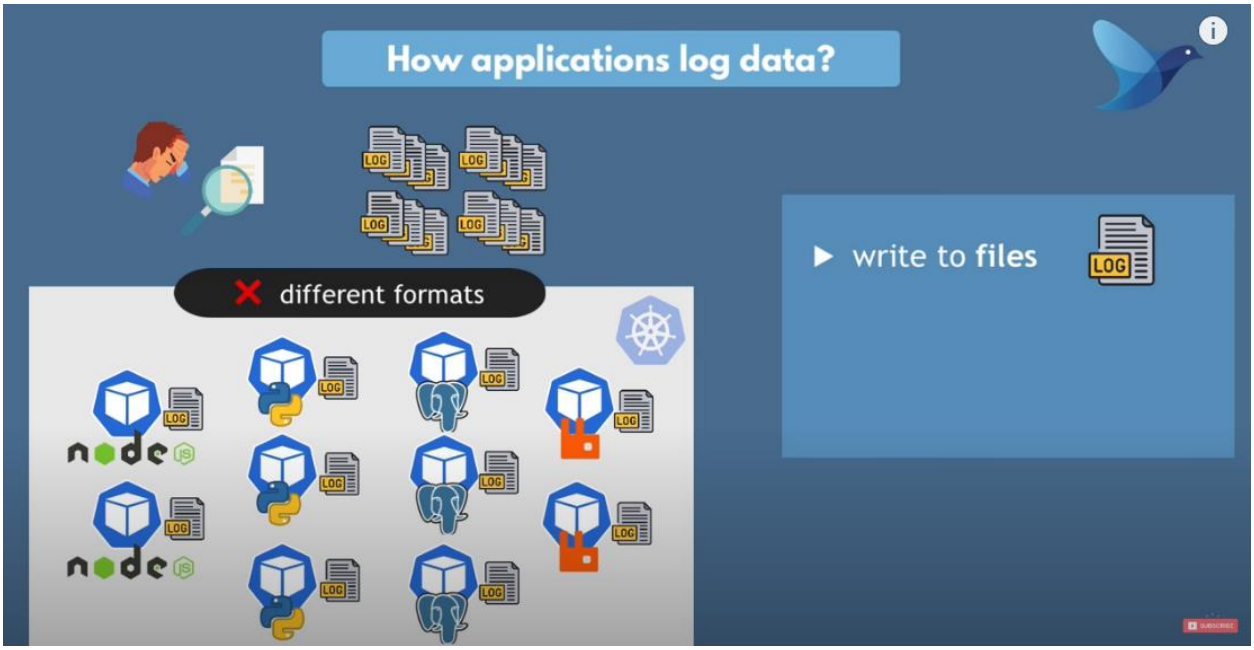


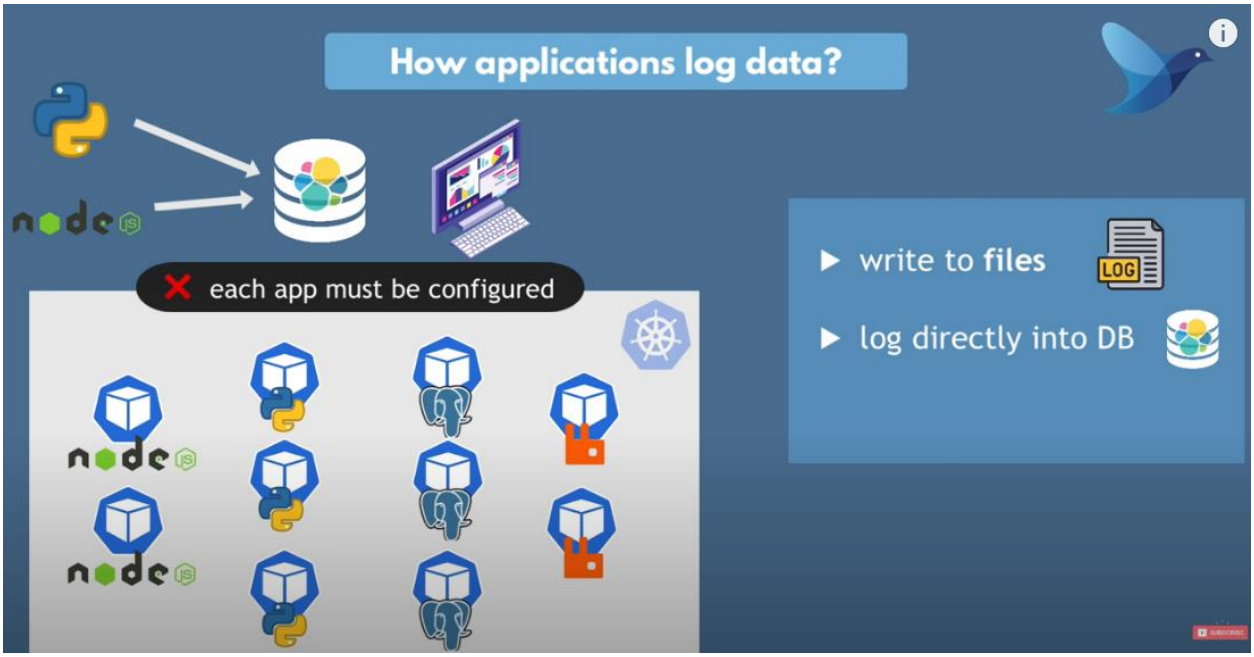
How it works

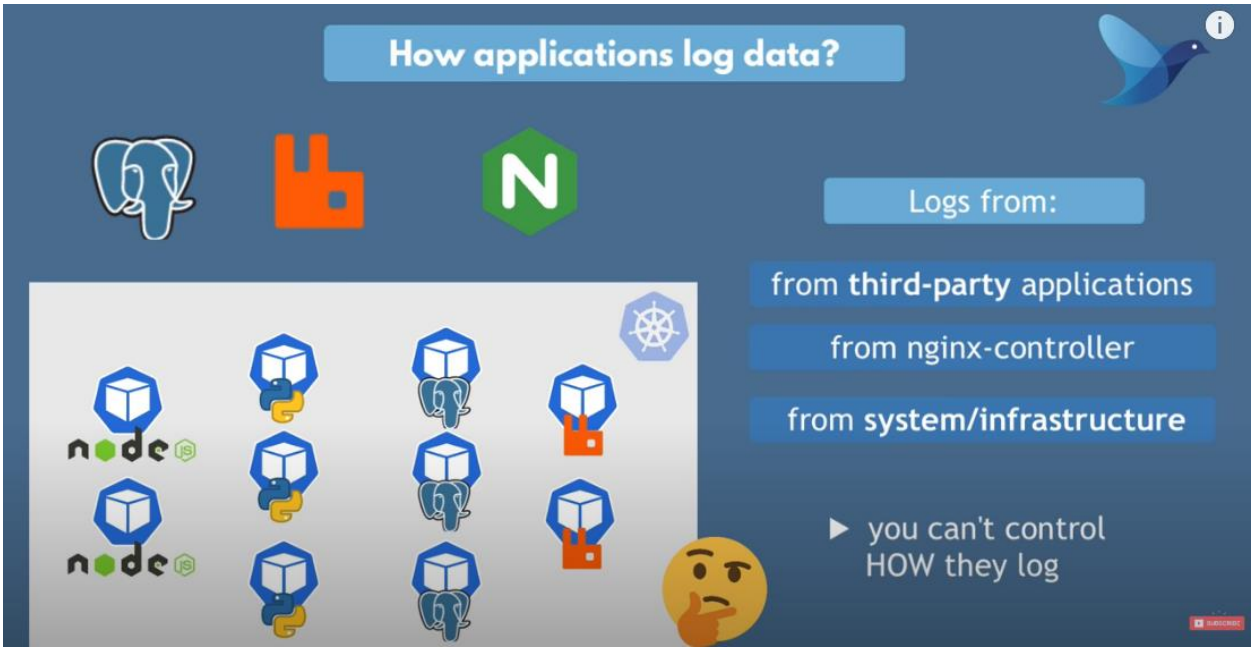
A log and event management appliance should be able to collect from multiple sources, normalize, time stamp and analyze the results, generate alerts and reports for audit and regulatory compliance, and support data mining for forensic and root-cause investigations.






How applications log data?
different formats
write to files


How applications log data?
each app must be configured
write to files
log directly into DB



problem 1: need ad-hoc parsing

the text-based logs have their own format, and analytics engineer need to write a dedicated parser for each format. but that's probably not the best use of your time. you should be analyzing data to make better business decisions instead of writing one parser after another.

problem 2: lacks freshness

the logs lag. the realtime analysis of user behavior makes feature iterations a lot faster. a nimbler a/b testing will help you differentiate your service from competitors.

this is where **fluentd** comes in. **we believe fluentd solves all issues of scalable log collection** by getting rid of files and turning logs into true semi-structured data streams.

what's fluentd?

the best way to describe fluentd is 'syslogd that understands json'. the notable features are:

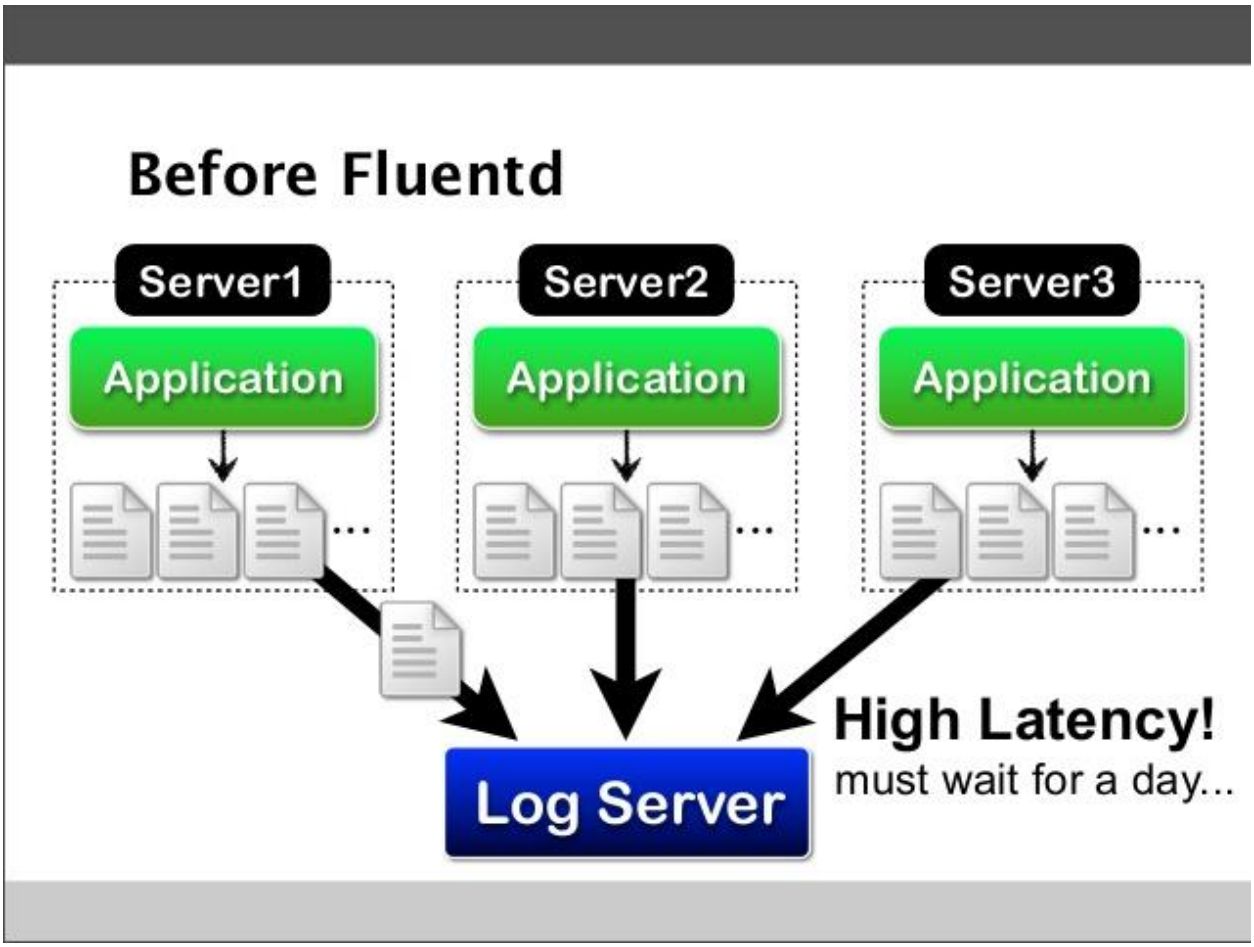
- **easy installation** by rpm/deb/gem
- **small footprint** with 3000 lines of ruby
- **semi-structured data** logging
- **easy start** with small configuration
- **fully pluggable** architecture, and plugin distribution by ruby gems

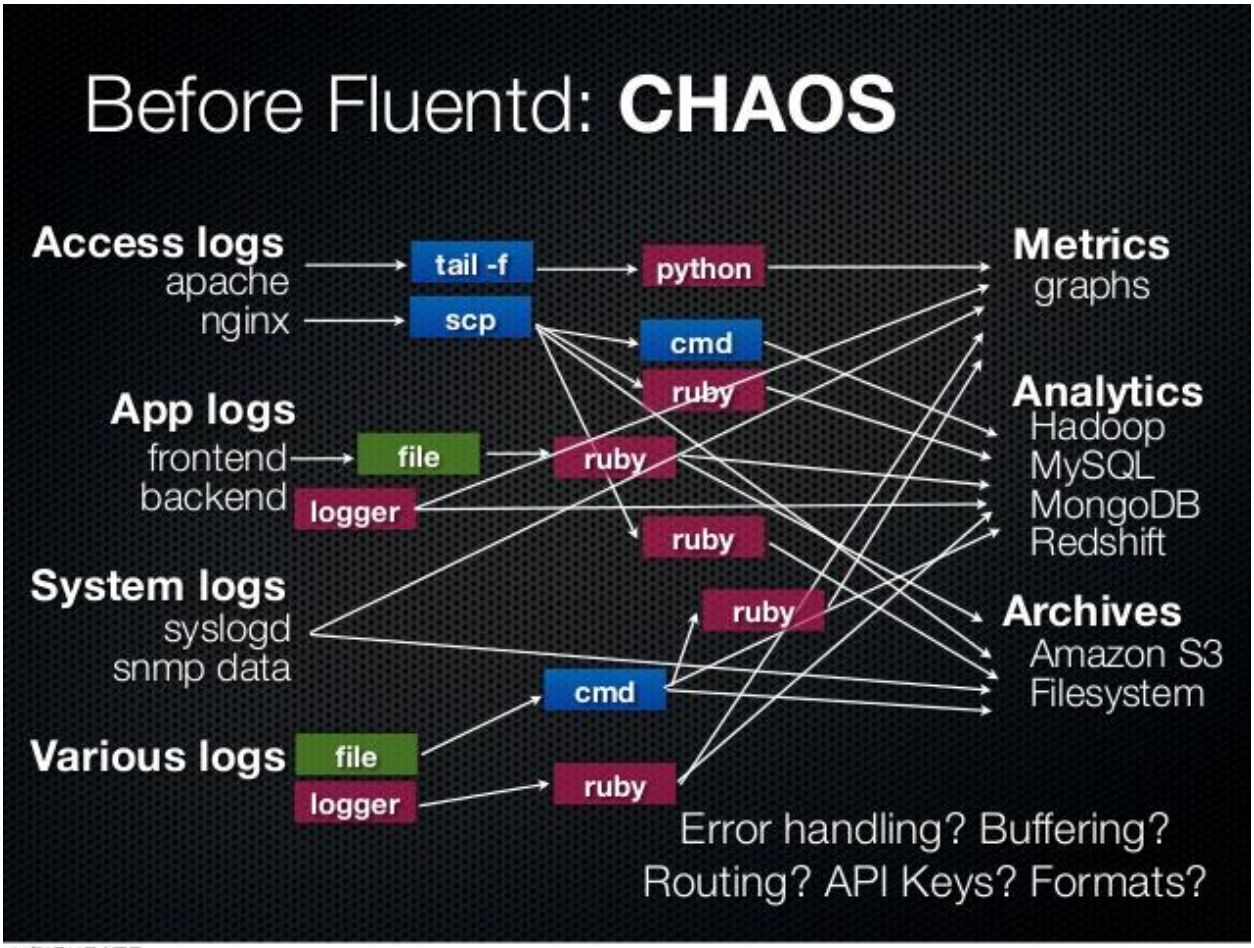
Fluentd

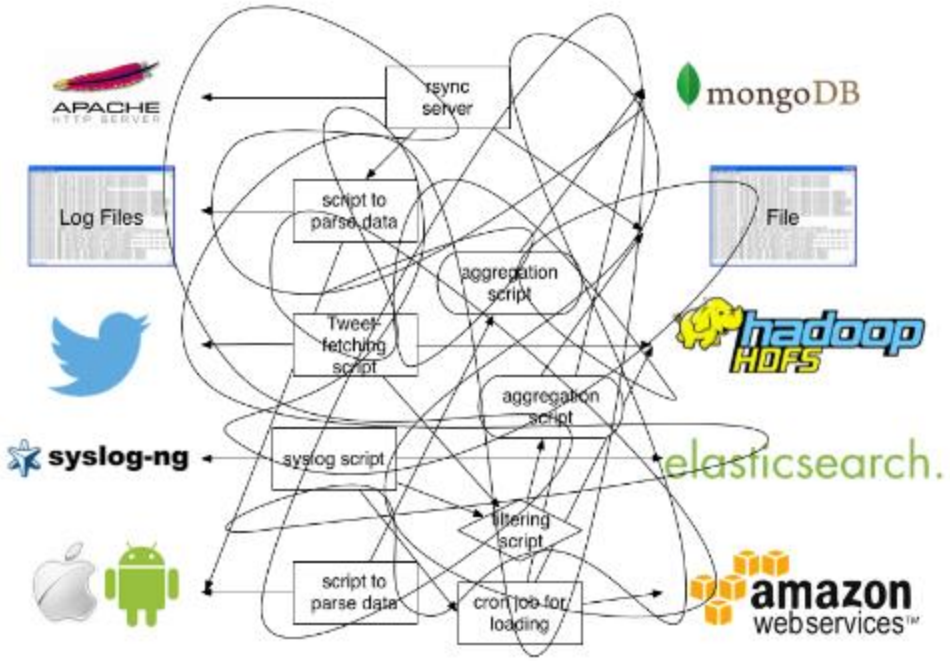
- Fluentd is an open source data collector for unified logging layer.
- Fluentd allows you to unify data collection and consumption for a better use and understanding of data.

Fluentd is an open-source log data collection software designed to separate data sources from the back-end system using a Unified Logging Layer (ULL). Here are some of the software's major highlights:

- **Unified Log Structuring With JSON:** Fluentd structures your log data in JSON, allowing you to collect, filter, buffer, and centralize logs from multiple sources (mobile devices, web servers, etc.).
- **Flexible Plug-In Architecture:** Using Fluentd plug-ins, IT teams can expand their functionalities and easily connect with several data sources and outputs.
- **Minimal Resource Requirements:** Fluentd is a lightweight log collection tool. It's written using C language and a Ruby framework, which is why it requires minimal system resources and memory allocation.
- **Data Loss Prevention:** Fluentd supports robust failover to avoid data loss. It's an extremely reliable tool in terms of memory allocation and file-based buffering, and it can store data in multiple systems.


Before Fluentd
Server1
Application
Server2
Application
Server3
Application
Log Server
High Latency!
must wait for a day...

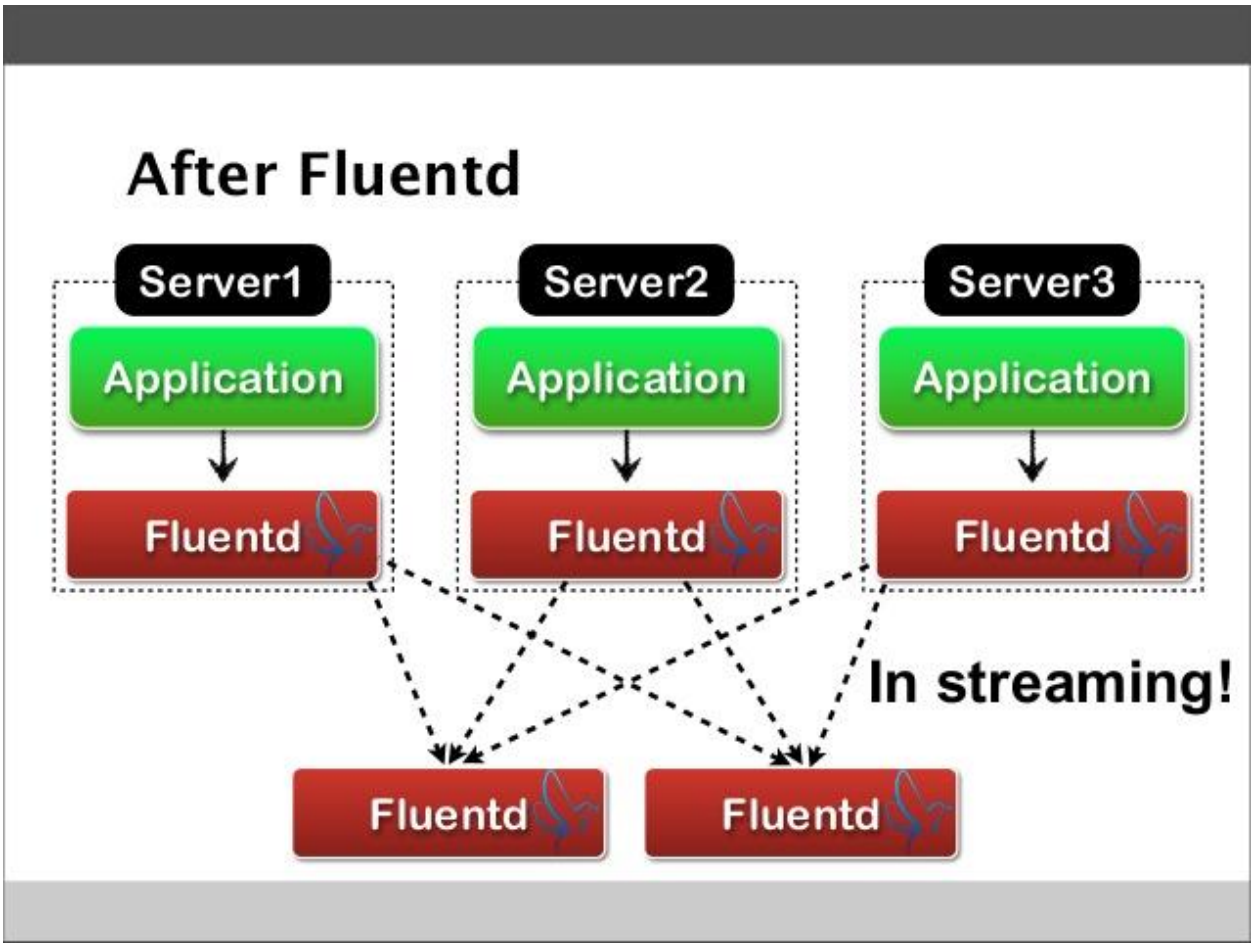

Before Fluentd: CHAOS
Access logs
apache
nginx
tail -f
scp
python
cmd
ruby
App logs
frontend
backend
file
logger
ruby
ruby
System logs
syslogd
snmp data
ruby
cmd
Various logs
file
logger
ruby
Metrics
graphs
Analytics
Hadoop
MySQL
MongoDB
Redshift
Archives
Amazon S3
Filesystem
Error handling? Buffering?
Routing? API Keys? Formats?



Before Fluentd

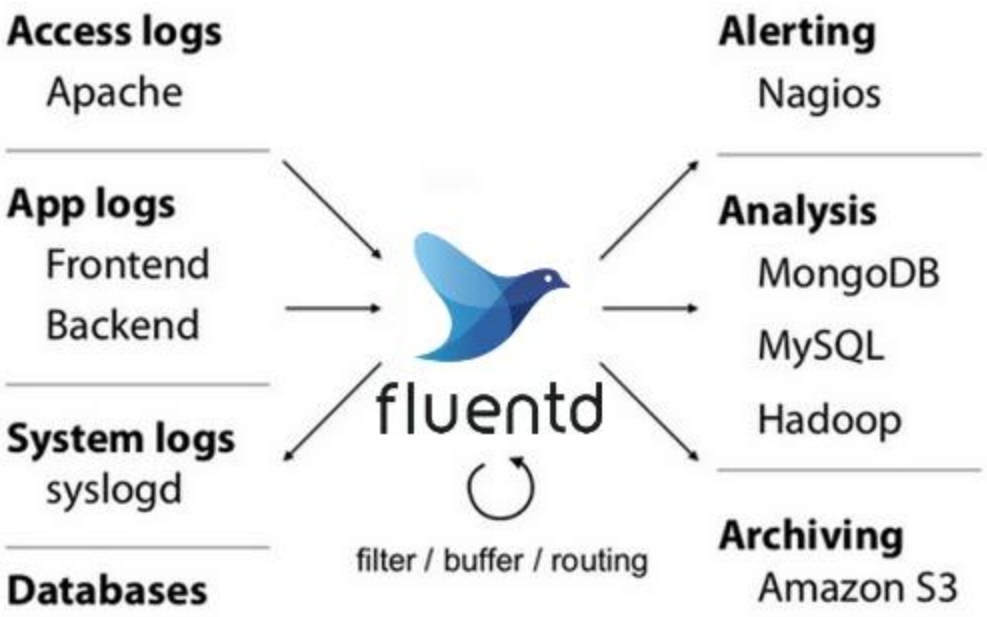
Before EFK

- Log files across the server
- Grep/shell scripting mess
- Production system access

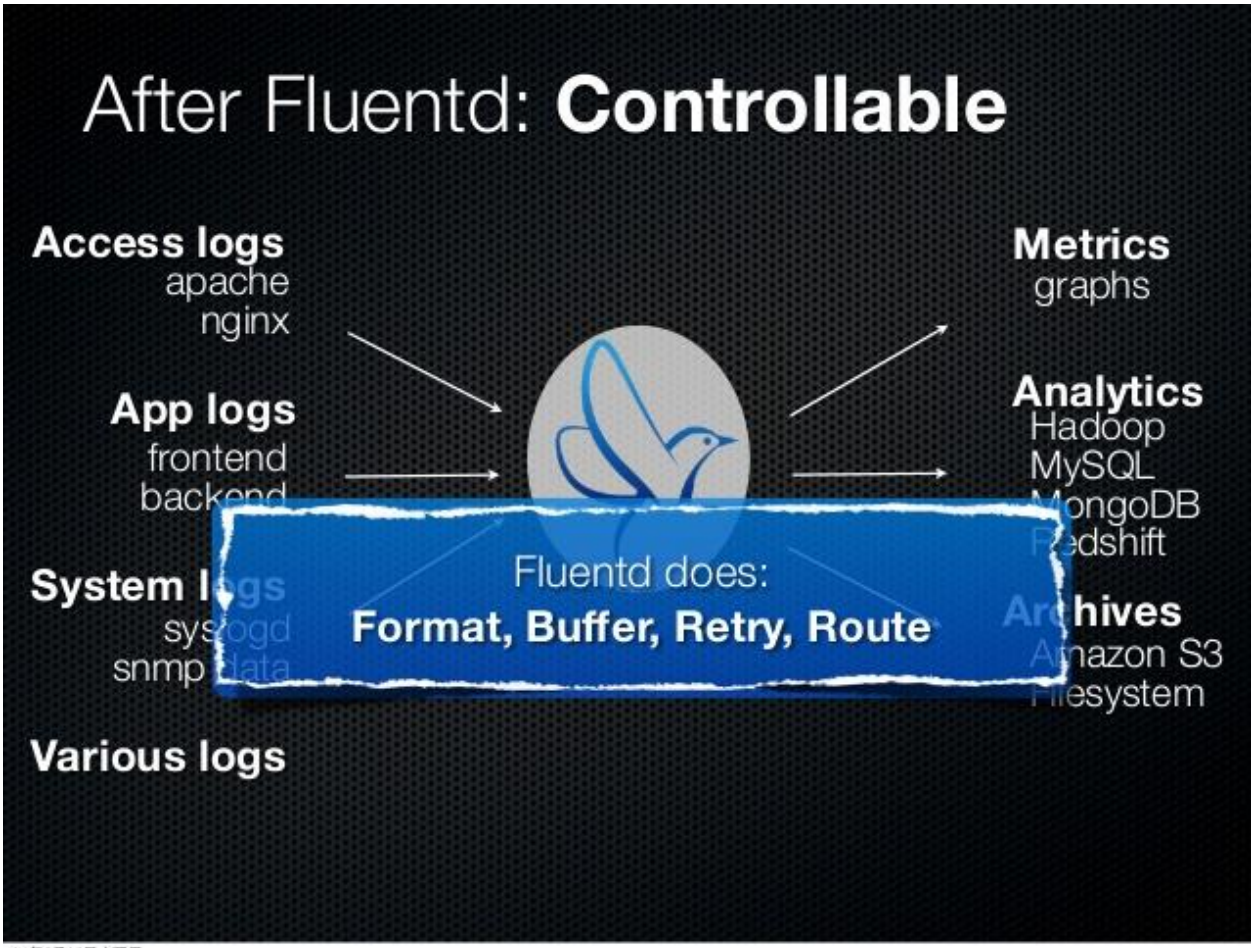


EFK:

- Detecting log patterns
- Easily Searchable
- Trouble-shooting issues



After Fluentd

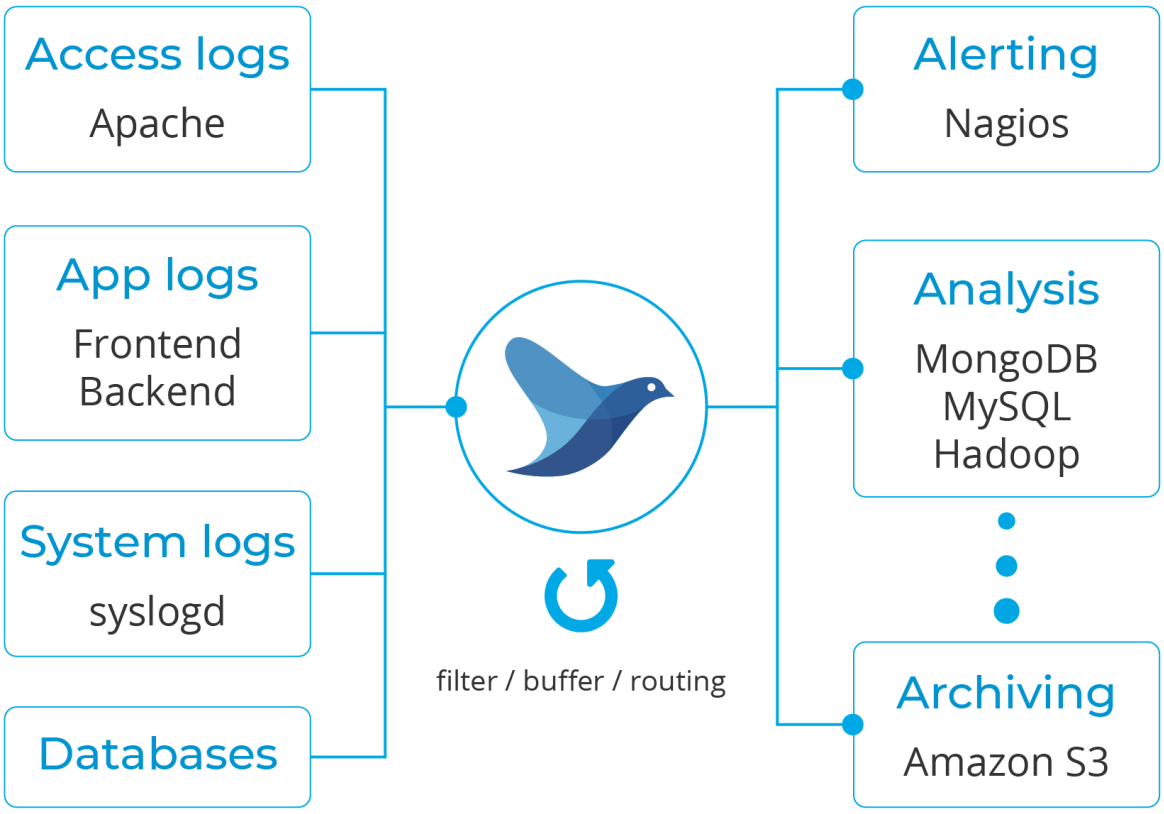

After Fluentd: Controllable
Access logs
apache
nginx
App logs
frontend
backend
System logs
syslogd
snmp
Various logs
Metrics
graphs
Analytics
Hadoop
MySQL
MongoDB
Redshift
Archives
Amazon S3
Filesystem
Fluentd does:
Format, Buffer, Retry, Route

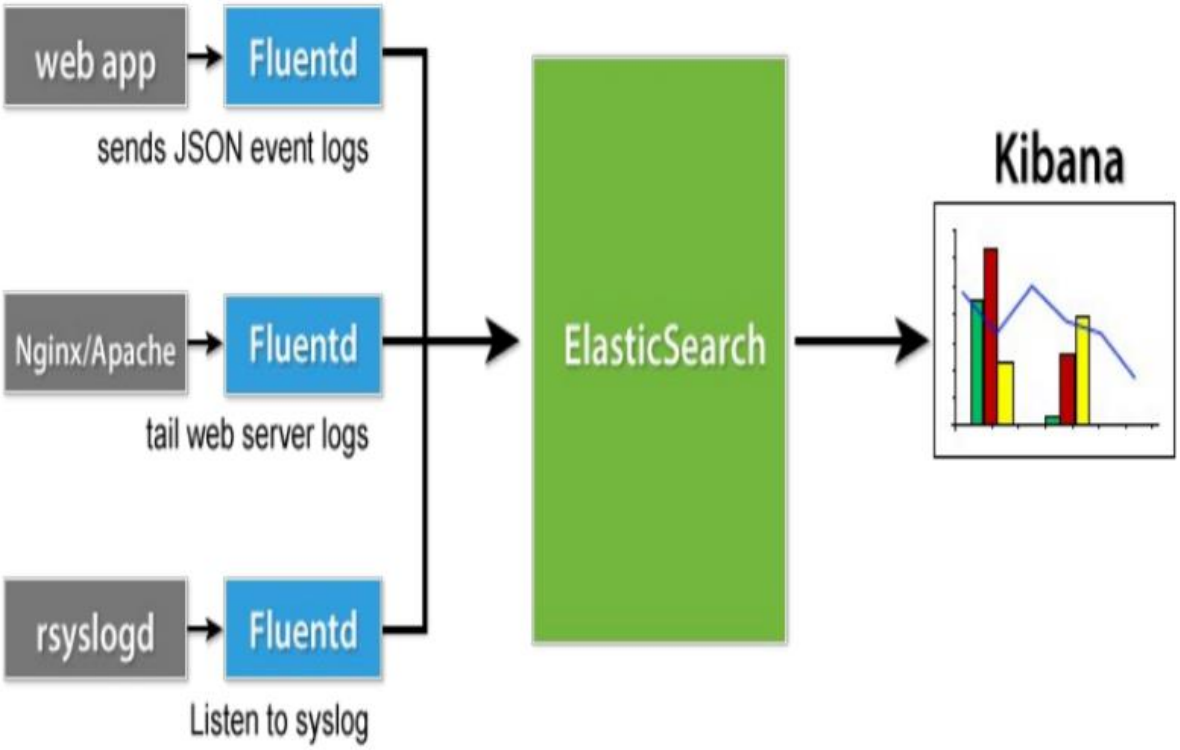
Need for Log Management

- Logs are usually in proprietary format and difficult to manage
- Routine log reviews and analysis are beneficial for identifying security incidents, policy violations, fraudulent activity, and operational problems
- Logs can also be useful for performing auditing and forensic analysis, supporting the organization's internal investigations, establishing baselines, and identifying operational trends
- Legal compliance. For critical applications like, health, public financial records, bank accounts, Government requires the organizations to maintain logs
- Protecting the trustworthiness of the log sources and also, the logs themselves need to be protected from malicious activities

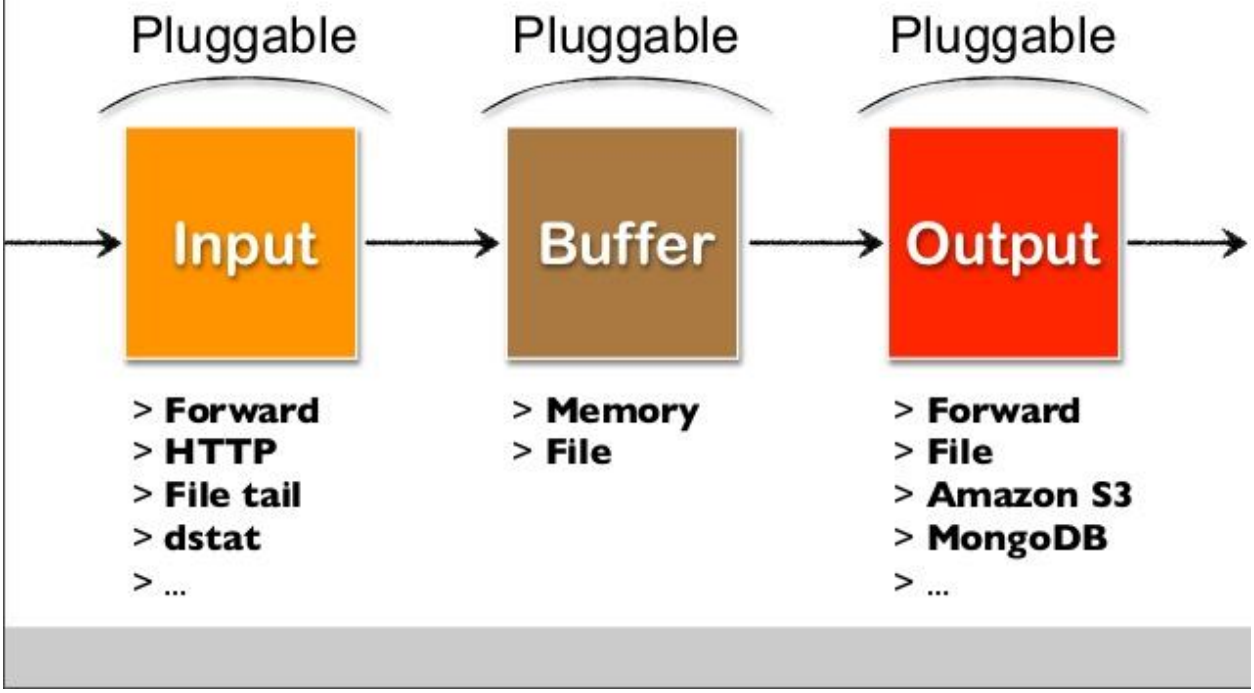
Log Management Infrastructure

- A three-tier Architecture
  - Log generation : Synchronized hosts generate
  - Logs analysis and storage : One or more log servers that receive the logged data. This transfer is either real-time or periodic. Such servers are called collectors or aggregators
  - Log monitoring : analyze and monitor the logged data using application consoles

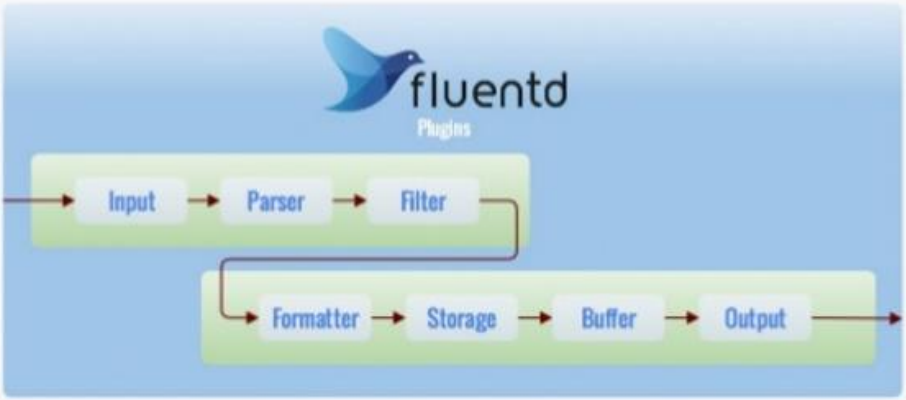

Access logs
Apache
App logs
Frontend
Backend
System logs
syslogd
Databases
filter / buffer / routing
Alerting
Nagios
Analysis
MongoDB
MySQL
Hadoop
Archiving
Amazon S3


web app
Fluentd
sends JSON event logs
Nginx/Apache
Fluentd
tail web server logs
rsyslogd
Fluentd
Listen to syslog
ElasticSearch
Kibana

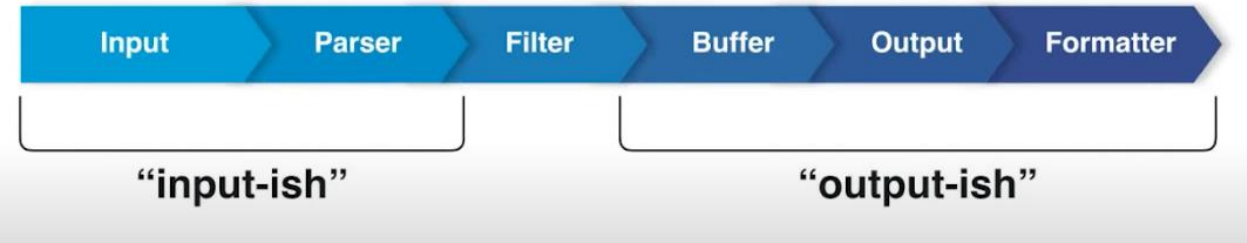
Architecture



Available Fluentd Plugin Types



INTERNAL ARCHITECTURE



input

input is the place where the log comes in. the user can extend it to feed the events from various sources. the example input supported officially includes: http+json, tailing files (apache log parser is supported), syslog. of course you can add input plugin by writing a ruby plugin.

buffer

buffer exists for reliability. when the output fails, the events are kept by buffer and automatically retried. memory or file buffer is supported now.

output

buffer creates chunks of logs, and passes them to the output. output stores or forwards chunks. the buffer waits several seconds to 1 minute, to create chunks. this is really efficient for writing into the storage which supports batch-style importing.

Fluentd Plugins Used

- in_forward: capture logs securely on port 24224 and unsecurely on port 24223
- parser_multi_format: parse logs where the log stream has more than one format e.g Redis
- filter_record_transformer: used to add a 'source' key value pair
- out_elasticsearch: forward logs to Elasticsearch targetting different indices as appropriate
- out_copy: copies logs to more than one output source e.g. Elasticsearch AND stdout
- out_rewrite_tag_filter: used to rewrite the tags from k8s and re-emit logs to process

Fluentd

- Fluentd has 5 types of plugins: Input, Parser, Output, Formatter and Buffer

Input Plugins:

- in_forward
- in_http
- in_tail
- in_exec
- in_syslog

Output Plugins:

- out_forward
- out_mongo
- out_file
- out_s3

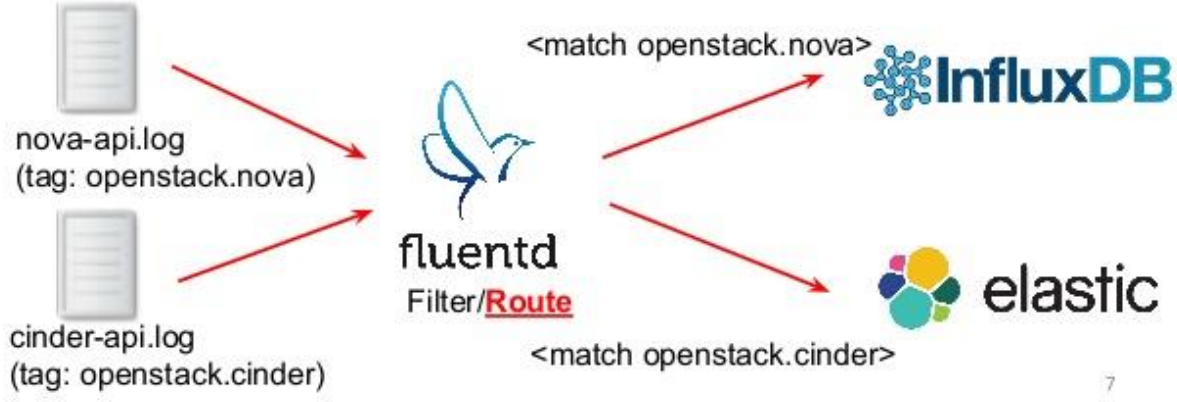
Buffer Plugins:

Buffer plugins are used by buffered output plugins, such as out_file, out_forward, etc

- buf_memory
- buf_file

Configuration: Fluentd

- Every inputs are tagged
- Logs will be routed by tag



Fluentd Configuration: Input

- Every inputs will be tagged

```
<source>
  @type tail
  path /var/log/nova/nova-api.log
  tag openstack.nova
</source>
```

Example of tailing nova-api log

8

Fluentd Configuration: Output

```
<match openstack.nova> # nova related logs
  @type elasticsearch
  host example.com
</match>

<match openstack.*> # all other OpenStack related logs
  @type influxdb
  # ...
</match>
```

Routed by tag
(First match is priority)

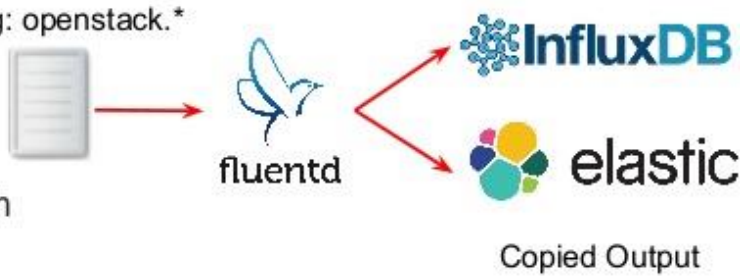
Wildcards can be used

9

Fluentd Configuration: Copy

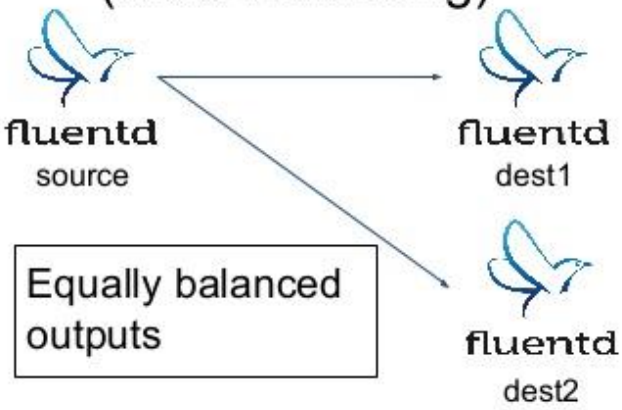
```
<match openstack.*>
  @type copy
  <store>
    @type influxdb
  </store>
  <store>
    @type elasticsearch
  </store>
</match>
```

Copy plugin enables multiple outputs for a tag



Fluentd Transport: forward

- Active-Active (Load Balancing)

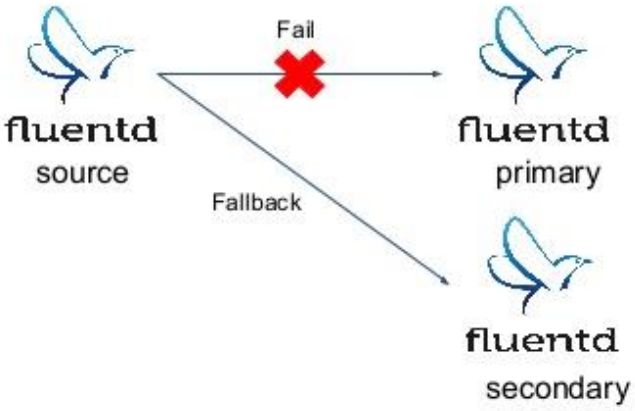


Equally balanced outputs

```
<match openstack.*>
  type forward
  <server>
    host dest1
  </server>
  <server>
    host dest2
  </server>
</match>
```

Fluentd Transport: forward

- Active-Standby

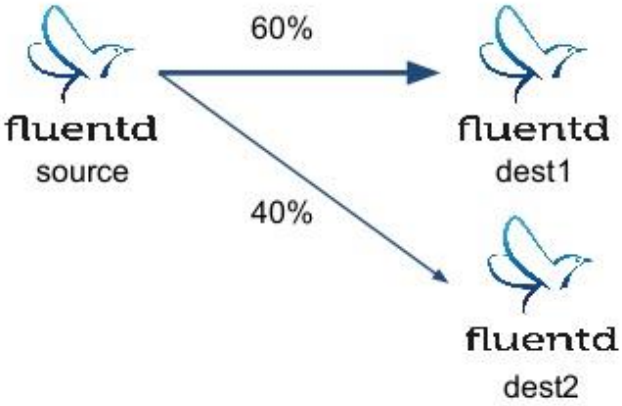



```
<match openstack.*>
  type forward
  <server>
    host primary
  </server>
  <server>
    host secondary
    standby
  </server>
</match>
```

24

Fluentd Transport: forward

- Weighted Load Balancing

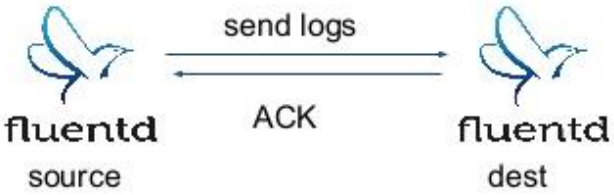



```
<match openstack.*>
  type forward
  <server>
    host dest1
    weight 60
  </server>
  <server>
    host dest2
    weight 40
  </server>
</match>
```

25

Fluentd Transport: forward

- At-least-one Semantics (may affect performance)



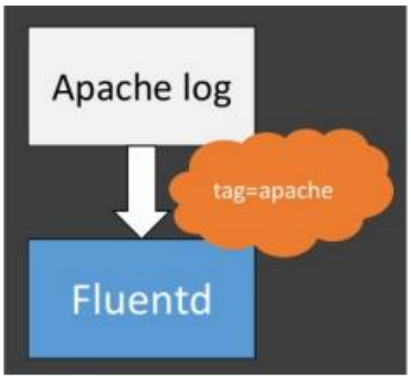
Logs are re-transmitted until ACK is received

```
<match openstack.*>
  type forward
  require_ack_response
  <server>
    host dest
  </server>
</match>
```

26

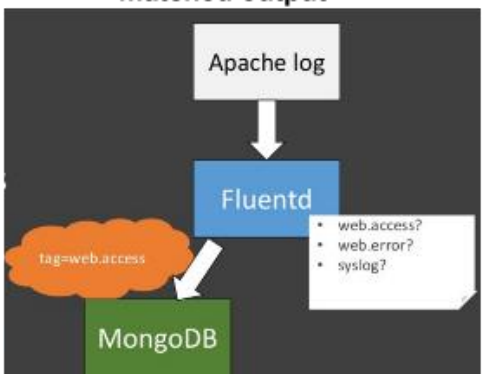
Lifecycle of Fluentd Event

All Input events should be tagged:

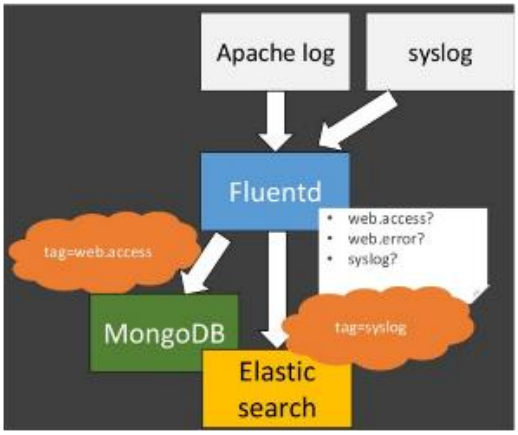


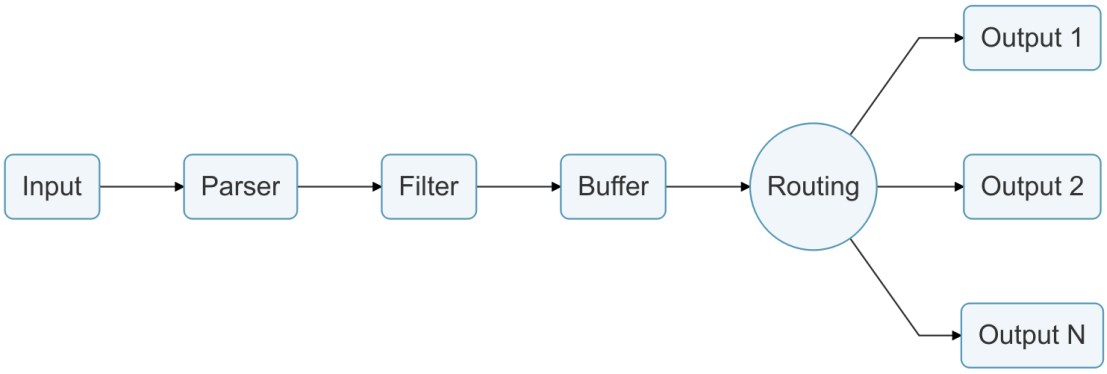
It matches the tag against the outputs:

Fluentd sends the event to matched output



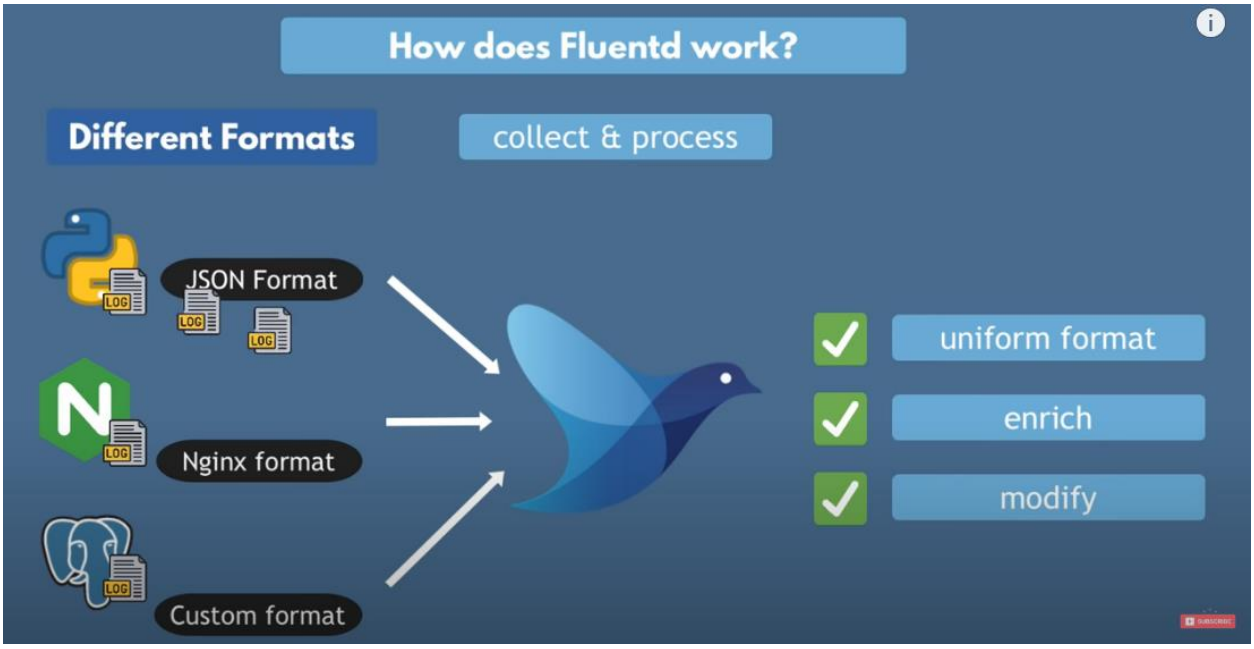
It can support multiple sources and destinations

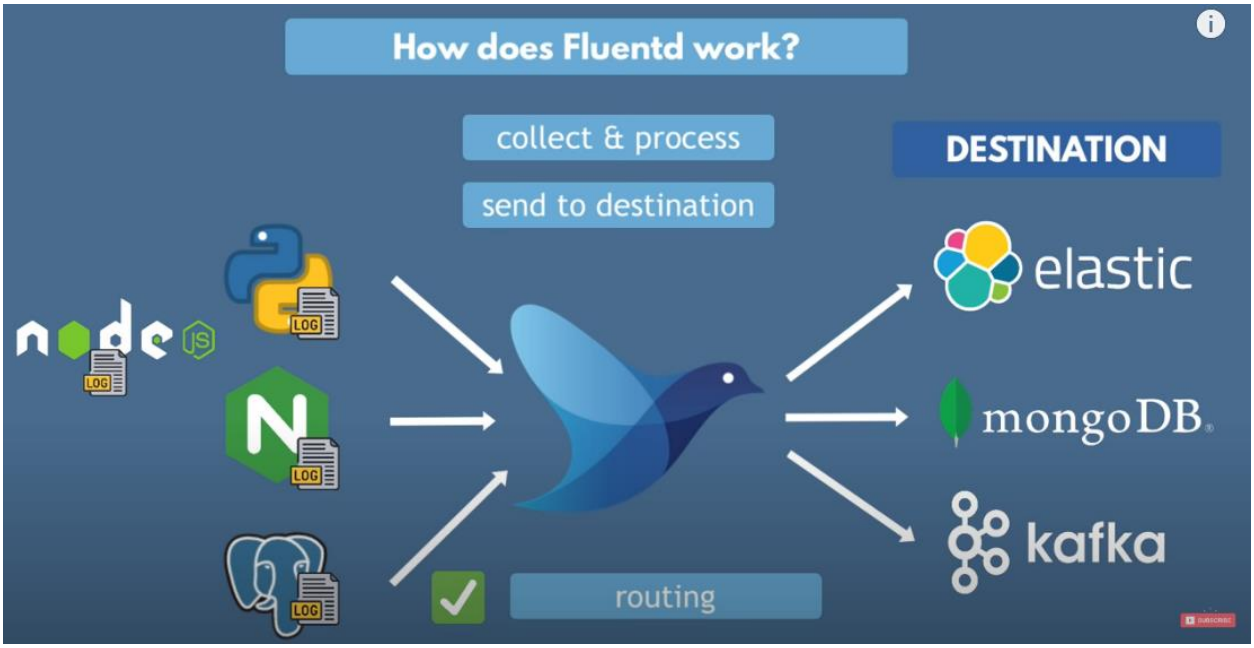


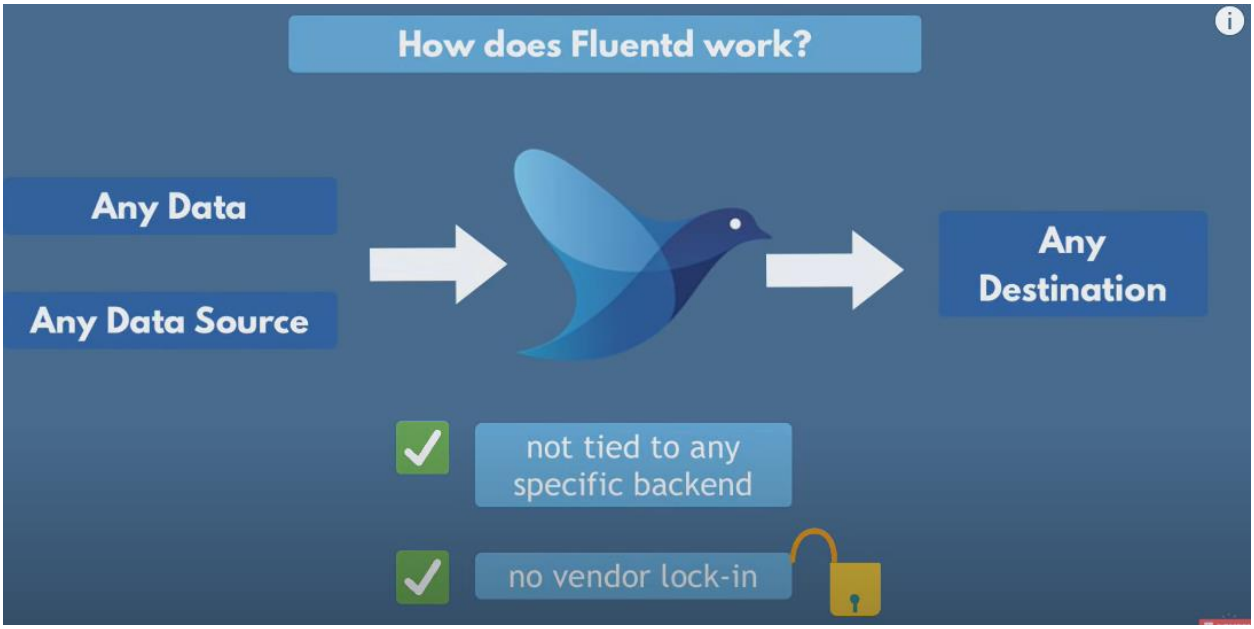


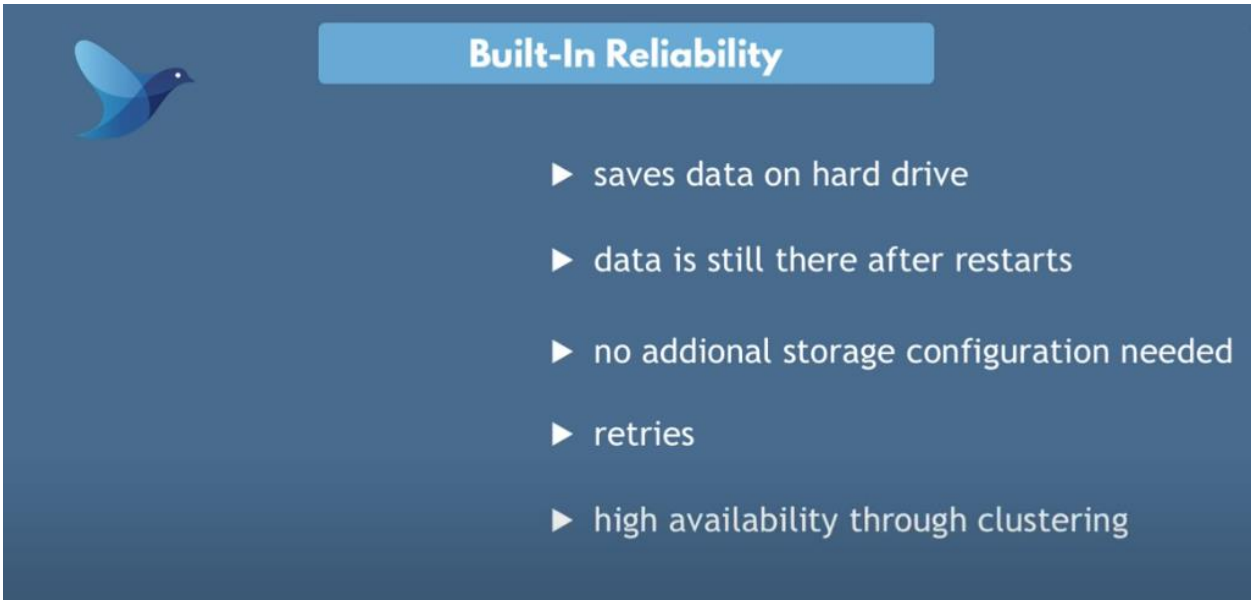
Fluentd Event Structure

- `tag`: From where an event originated; used for message routing
- `time`: The epoch time at which an event occurred
- `record`: The event log content as a JSON object

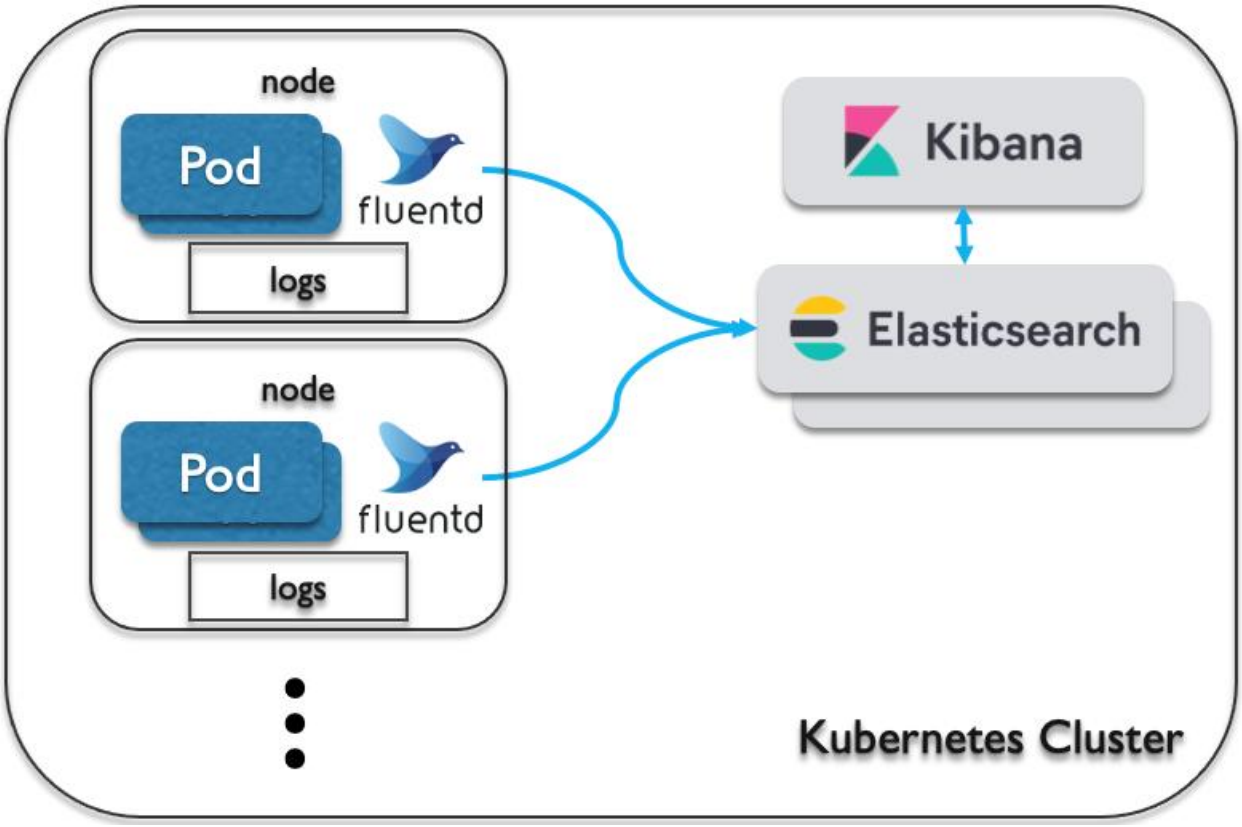



How does Fluentd work?
collect & process
send to destination
DESTINATION
elastic
mongoDB
kafka
routing


How does Fluentd work?
Any Data
Any Data Source
Any Destination
not tied to any specific backend
no vendor lock-in



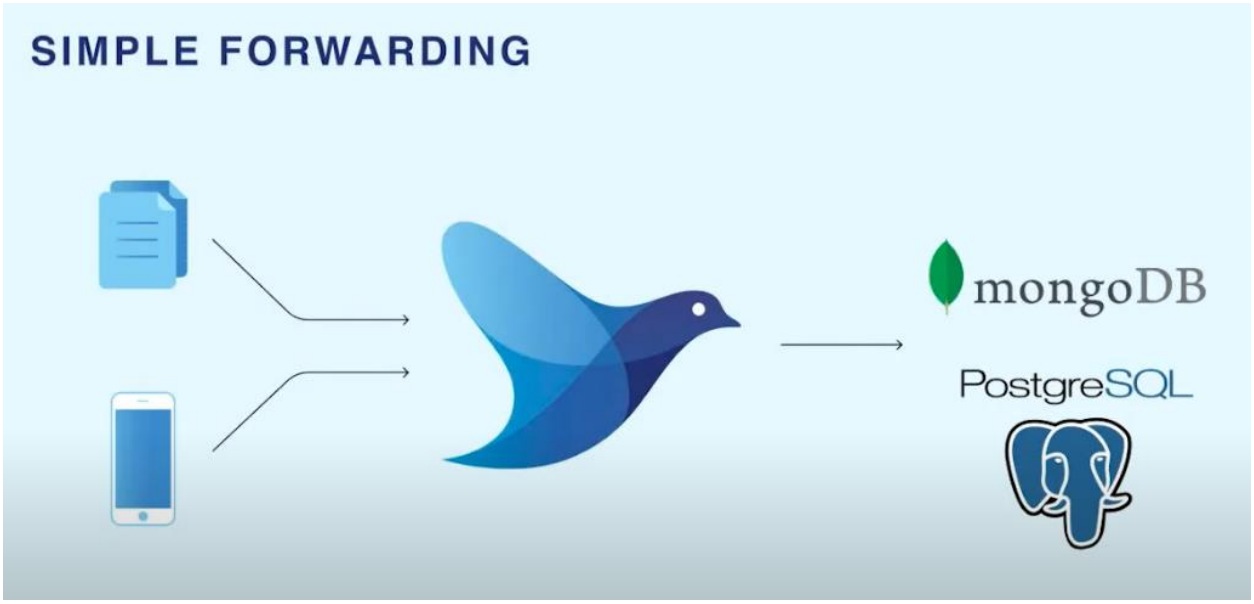
EFK(**Elastic FluentD Kibana)**

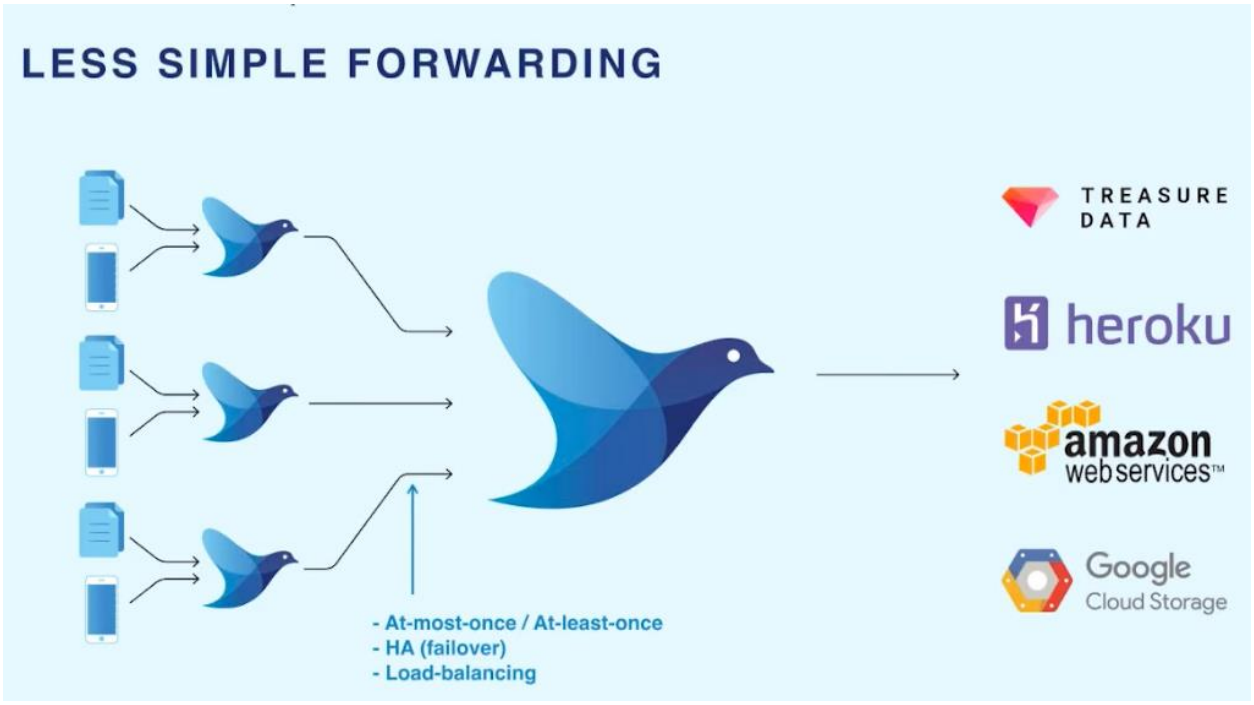


Fluentd vs Logstash

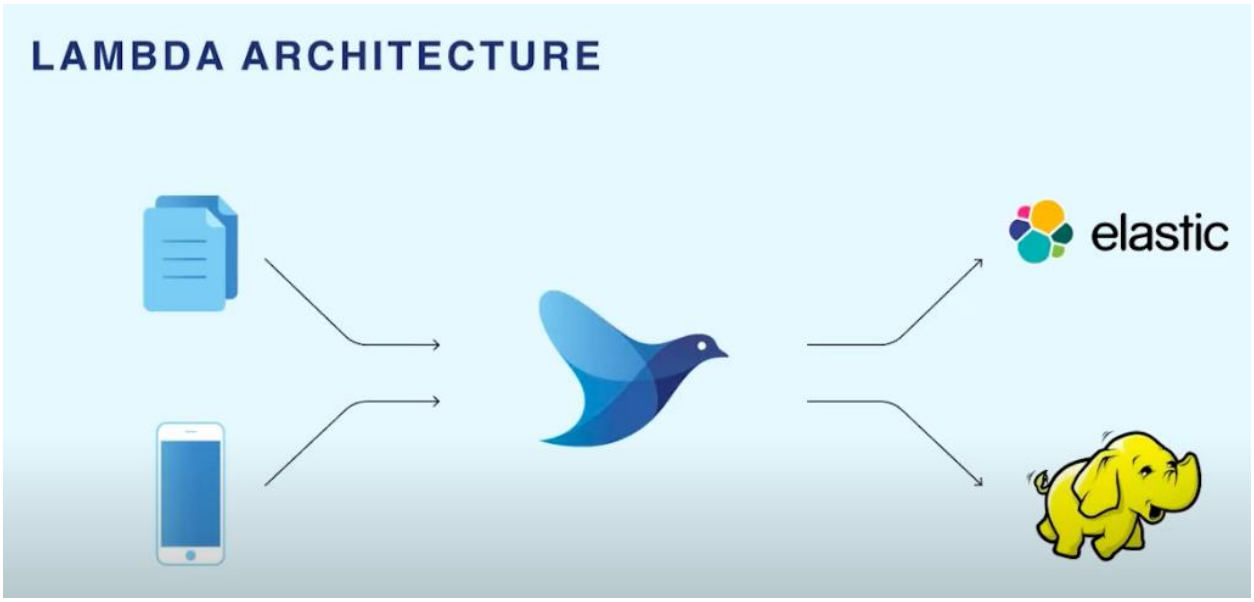
	Platform	Event Routing	Plugin Ecosystem	Transport	Performance
Logstash	Linux & Windows	Algorithmic statements	Centralized	Deploy with Redis for reliability.	Uses more memory. Use Elastic Beats for leafs.
Fluentd	Linux & Windows	Tags	Decentralized	Built-in reliability but hard to configure.	Uses less memory. Use Fluent Bit and Fluentd Forwarder for leafs.

USE CASES

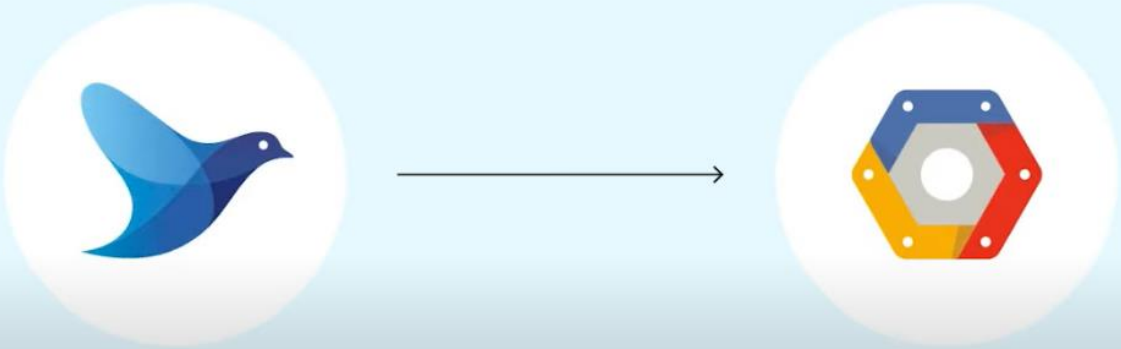

SIMPLE FORWARDING
mongoDB
PostgreSQL

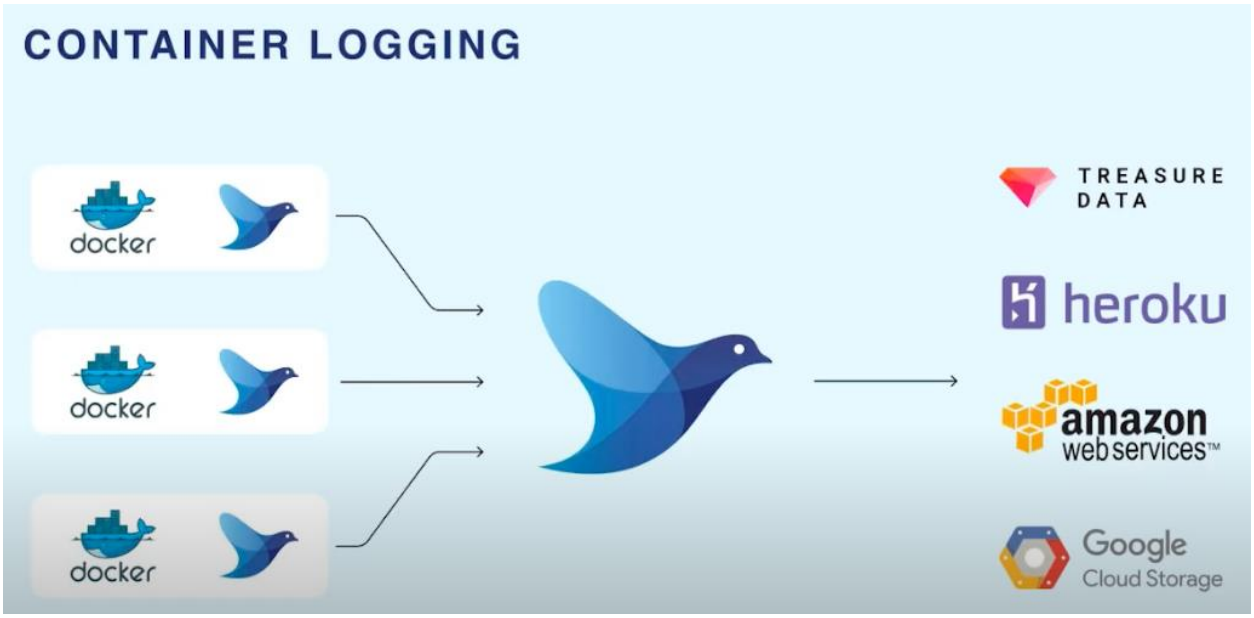

LESS SIMPLE FORWARDING
TREASURE DATA
heroku
amazon web services
Google Cloud Storage
- At-most-once / At-least-once
- HA (failover)
- Load-balancing

LAMBDA ARCHITECTURE



LOGGING TO GCP




CONTAINER LOGGING
docker
docker
docker
TREASURE DATA
heroku
amazon web services™
Google Cloud Storage