

# **EFK를 활용한 로그 수집과 분석 (3일)**

**(Elasticsearch + Fluentd + Kibana)**

**2022 – 09**

## 2 일차 일차 : EFK, Fluentd

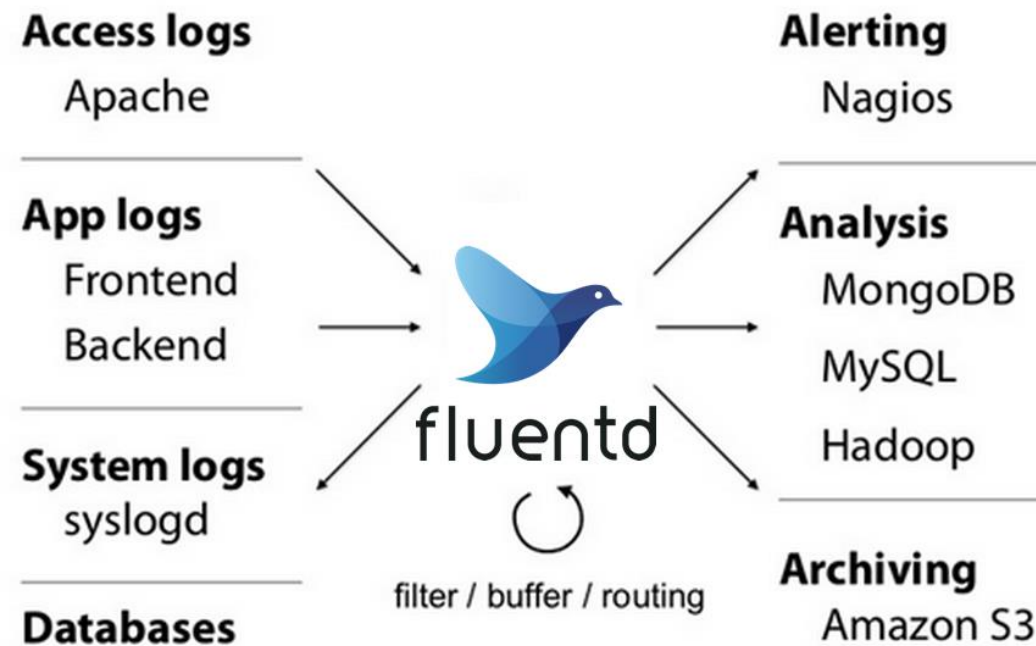
- Fluentd를 활용한 Log 수집 분석
- Fluentd Input/Output plugin의 종류 및 데이터 수집과 분석
- 다양한 Log 수집 Pipeline 구축(실습)
  - 1) Ngix Log
  - 2) Syslogd
  - 3) Database와 Files log

## 교육 내용 및 일정 소개

교육 내용	주제	시간	학습 내용
	Fluentd 의 구조와 기능	1H	▪ Fluentd 개요 및 로그 수집 방법(data source & output)
	Fluentd를 통한 외부 로그 및 메트릭 수집 및 분석	1H	▪ Fluentd 파이프라인 구성 (Fluentd Configuration)
		1H	▪ Fluent Input/Output plugin의 종류 및 데이터 수집과 분석
	서버, DB, Web 서버 등 다양한 로그 수집 Pipeline 구성 및 실습	1H	▪ [실습] Fluentd(td-agent) 설치 및 td-agent.conf 파일 설정
		1H	▪ [실습] Nginx log 수집 및 분석
		1H	▪ [실습] 시스템 로그(rsyslogd) 수집 및 분석
		1H	▪ [실습] MySQL DB log 연동

# Fluentd: Open-Source Log 수집기

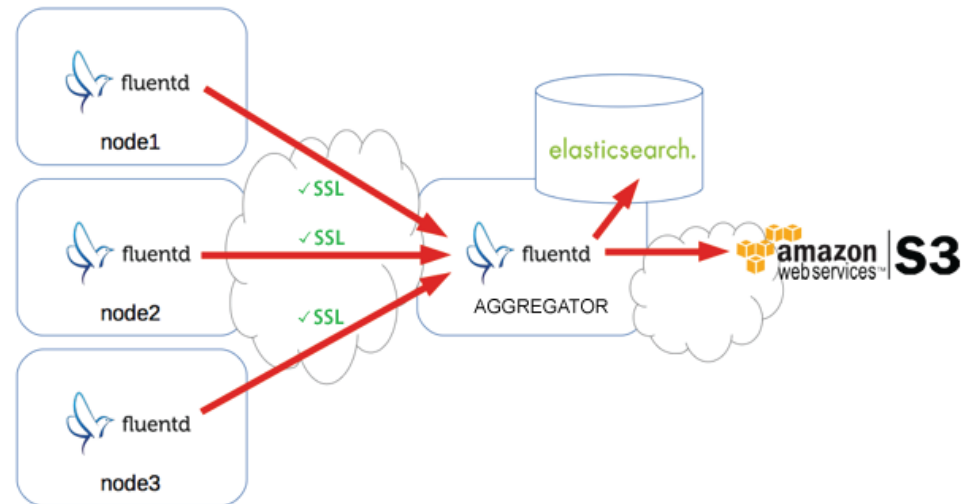
- RDBMS, NoSQL, IaaS, SaaS, Hadoop 수집된 로그를 거의 실시간으로 수집기 Fluentd로 안전하게 전송



<https://github.com/fluent/fluentd>

## Fluentd를 활용한 Log 수집 분석 사례 실습

- 웹 서버에서 Apache httpd 로그 및 syslog를 수집
- 수집된 로그를 거의 실시간으로 수집기 Fluentd로 안전하게 전송
- 수집된 로그를 Elasticsearch 및 Log 저장
- 실시간으로 Kibana로 데이터를 시각화



## Fluentd가 내부에서 처리하는 데이터의 특징

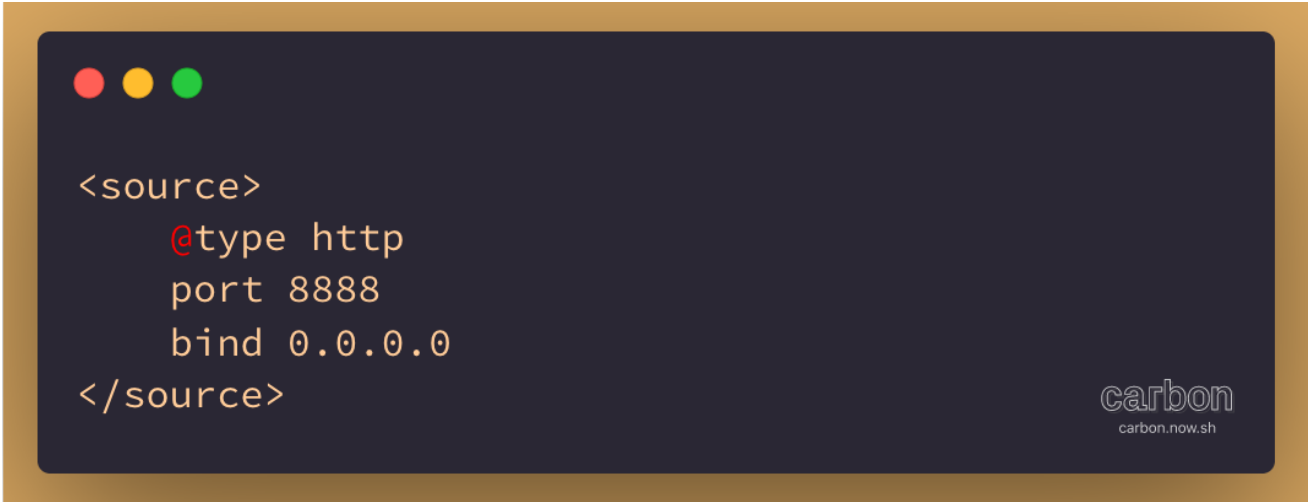
- 이벤트 | Event의 구성
- Fluentd가 읽어들이는 데이터는 tag, time, record 로 구성된 이벤트(Event) 로 처리
  - tag: 이벤트를 어디로 보낼지 결정하기 위한 구분값
  - time: 이벤트가 발생한 시간
  - record: 데이터 (JSON)
- 이벤트 컴포넌트 중에서 tag는 Fluentd 내부 Routing의 기본으로, 각 이벤트들이 설정된 tag에 맞는 태그로 흘러 들어갈 수 있게끔 구분해 주는역할 수행

## Fluentd 설정 파일의 확인

- Fluentd를 실행하기 위해, 각 라이프사이클에 해당하는 지정자들을 설정 파일(Configuration File)에 작성. Fluentd 에이전트가 해당 설정 파일을 물고 실행
- 기본적으로 Ubuntu의 경우, 설정 파일은 /etc/td-agent 경로 아래 td-agent.conf라는 파일로 설정
- Ubuntu에서 fluentd를 설치하는 경우, 관련 패키지 에러가 발생할수 있음
  - 패키지 관련 에러가 발생하면, apt-get을 이용하여 에러 로그에서 나타내고 있는 해당 패키지를 설치
  - Ubuntu 버전에 따라 제공하는 Fluentd 버전이 다르므로 반드시 확인
- td-agent 커맨드로 fluentd agent를 실행
- 백그라운드에서 데몬으로 수행하려면, -d 옵션과 --log 옵션을 추가하여 로그를 확인

## 플러그인의 종류: Input Plugin

- Input Plugin : 데이터가 어디로부터 오는지 설정
- input 플러그인은 source 태그 안에 정의한다.
  - 예로 http source의 경우
  - HTTP 8888 포트에 바인딩되어, 해당 포트의 Server를 지속해서 리스닝하고 있음을 표현



```
<source>  
  @type http  
  port 8888  
  bind 0.0.0.0  
</source>
```

carbon  
carbon.now.sh



## Filter Plugin : 이벤트를 어떤 프로세스로 처리하는지

- 아래의 설정을 예로 들면 test.cycle이라는 태그를 가진 이벤트 중에서, action key에서 logout이라는 패턴을 제외시키고, 제외되지 않은 메시지만을 match 태그로 전달할 수 있도록 함
- 실제로 http에서 json={"action":"login", "user":2} 형태의 이벤트가 전달됨

```
<source>
  @type http
  port 8888
  bind 0.0.0.0
</source>

<filter test.cycle>
  @type grep
  <exclude>
    key action
    pattern ^logout$
  </exclude>
</filter>

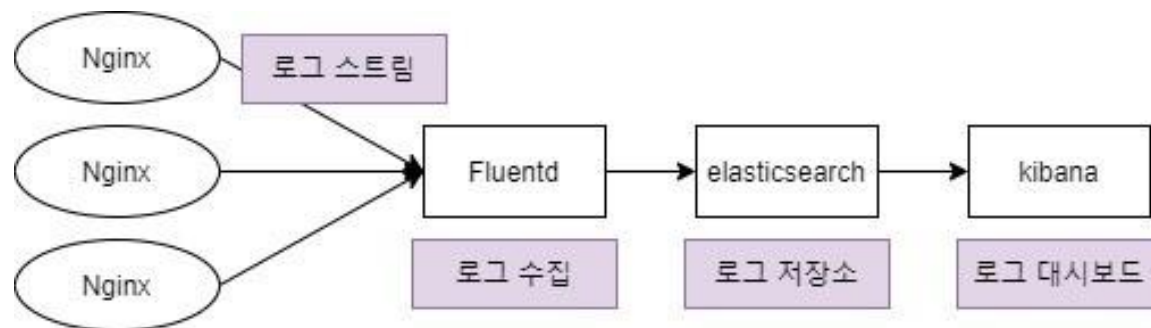
<match test.cycle>
  @type stdout
</match>
```

## Output Plugin : 이벤트를 어디로 전달하는지 설정

- output 플러그인은 match 태그 안에 정의
- tag와 매치되는 이벤트들을 다른 시스템으로 전달할 때 주로 사용
- v1.0부터 Buffering과 Flushing에 대한 설정을 match 태그 내부에 buffer 태그로 따로 정의하도록 변경

## Fluentd를 활용한 Log 수집 분석과 데몬셋(Daemonset)

- Nginx를 컨테이너 이미지로 POD에 배포, 그러면 nginx의 로그는 스트림으로 계속 출력되고
- 이때 로그들을 모아줄 수집기가 필요하다.
- 그게 Fluentd !
- Fluentd는 로그를 수집하는 툴이기 때문에 모든 node에 동일하게 배포되어야 하며, 그래서 DaemonSet으로 배포를 함



## EFK 실습 예제 I : EFK를 이용한 간단한 Log 수집 및 시각화

- 간단한 Log 생성 Python 코드: `main.py`

```
$ tail -f log.json
```

```
$sudo service td-agent status
```

```
$ sudo ls /etc/td-agent/.
```

```
$ sudo vi /etc/td-agent/td-agent.conf
```

```
$ sudo service td-agent stop
```

```
$ sudo service td-agent start
```

```
$ python main.py
```

## EFK 실습 예제 I : EFK를 이용한 간단한 Log 수집 및 시각화

- td-agent 설치

```
$ curl -L https://toolbelt.treasuredata.com/sh/install-ubuntu-xenial-td-agent2.sh | sh
```

- 데몬 실행

```
$ /etc/init.d/td-agent restart (재시작)
```

```
$ /etc/init.d/td-agent status (상태 확인)
```

- 시작시 자동 스크립트 실행

```
$ sudo update-rc.d td-agent defaults 95 10
```

```
$ sudo service td-agent status
```

## Fluentd Forwarder 예제 II: Ngix Log 수집

- Fluentd (on Ubuntu Precise) 설치

```
$ sudo curl -L http://toolbelt.treasuredata.com/sh/install-ubuntu-precise.sh | sh
```

- Web Log 서버의 Fluentd 접속 권한 설정

```
$ sudo chmod og+rx /var/log/httpd
```

```
$ sudo chmod og+r /var/log/messages /var/log/secure /var/log/httpd/*
```

- /etc/rsyslogd.conf 수정

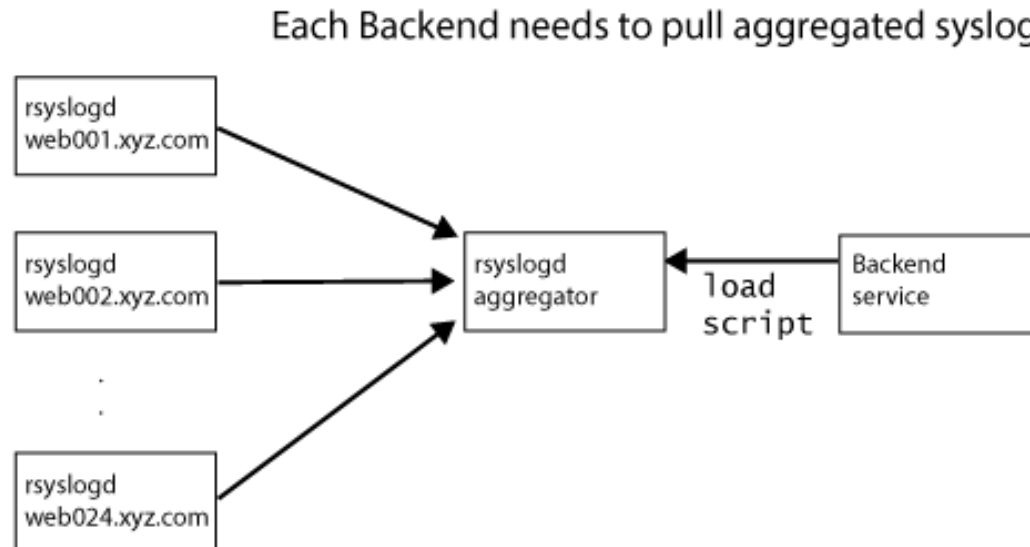
```
*.* @127.0.0.1:42185
```

- rsyslogd 재시작

```
$ sudo service rsyslog restart
```

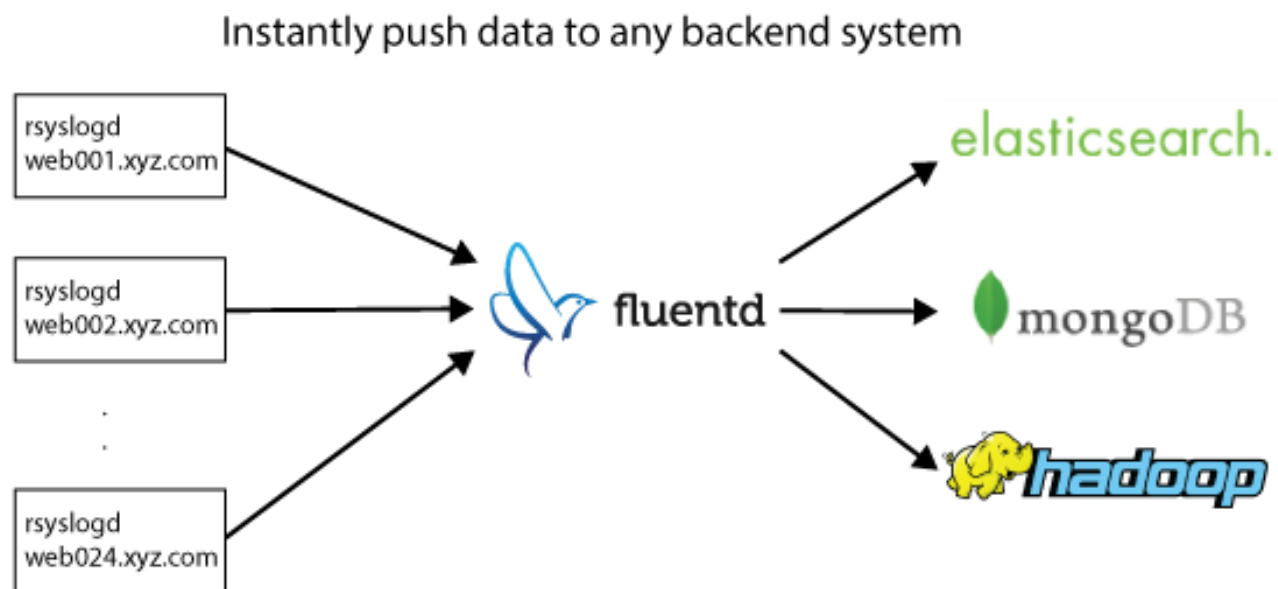
## Aggregating Rsyslogd 예제 III : Rsyslogd Output 수집

- rsyslogd는 syslog를 수집하고 집계하는 검증된 진정한 미들웨어



<https://www.fluentd.org/guides/recipes/rsyslogd-aggregation>

- Fluentd는 즉시 사용 가능한 많은 데이터 소비자를 지원합니다. 적절한 출력 플러그인을 설치하면 몇 가지 구성 변경으로 새 데이터 소스를 추가
- Fluentd는 조정 가능한 횟수 및 버퍼링 설정을 사용하여 각 소비자에게 데이터를 PUSH





- Fluentd 설정

```
$ curl -L http://toolbelt.treasuredata.com/sh/install-ubuntu-precise.sh | sh
```

- Elasticsearch 출력 플러그인을 설치하고, 실행

```
$ /usr/sbin/td-agent-gem install fluent-plugin-elasticsearch
```

- /etc/td-agent/td-agent.conf 설정

```
<source>
  type syslog
  port 42185
  tag rsyslog
</source>

<match rsyslog.**>
  type copy
  <store>
    # for debug (see /var/log/td-agent.log)
    type stdout
  </store>
  <store>
    type elasticsearch
    logstash_format true
    flush_interval 10s # for testing.
    host YOUR_ES_HOST
    port YOUR_ES_PORT
  </store>
</match>
```

/etc/td-agent/td-agent.conf 설정

- Elasticsearch로의 데이터 흐름을 다시 시작하고 확인

\$ sudo service td-agent restart

## Fluentd Forwarder 예제 IV: Fluentd MySQL slow log 연동

[https://github.com/JSJeong-me/EFK/blob/main/Day\\_02.md](https://github.com/JSJeong-me/EFK/blob/main/Day_02.md)