User/Group based access control for ElasticSearch + Kibana

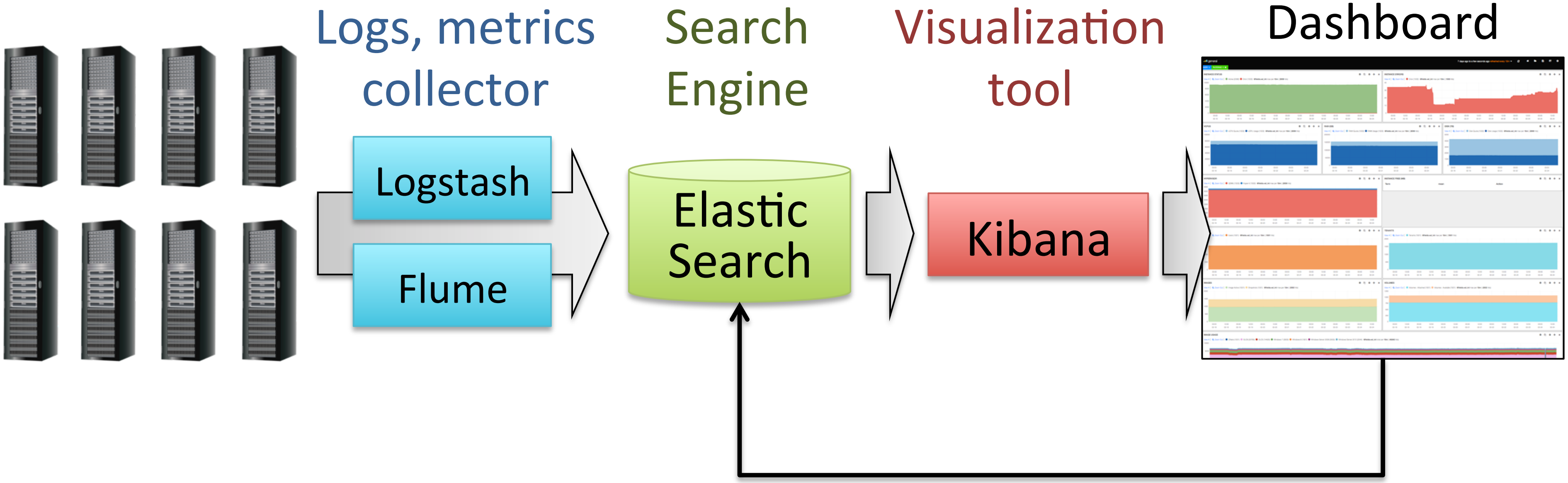
Wataru Takase

Computing Research Center, KEK, Japan

HEPiX Fall 2016 Workshop

Kibana and ElasticSearch

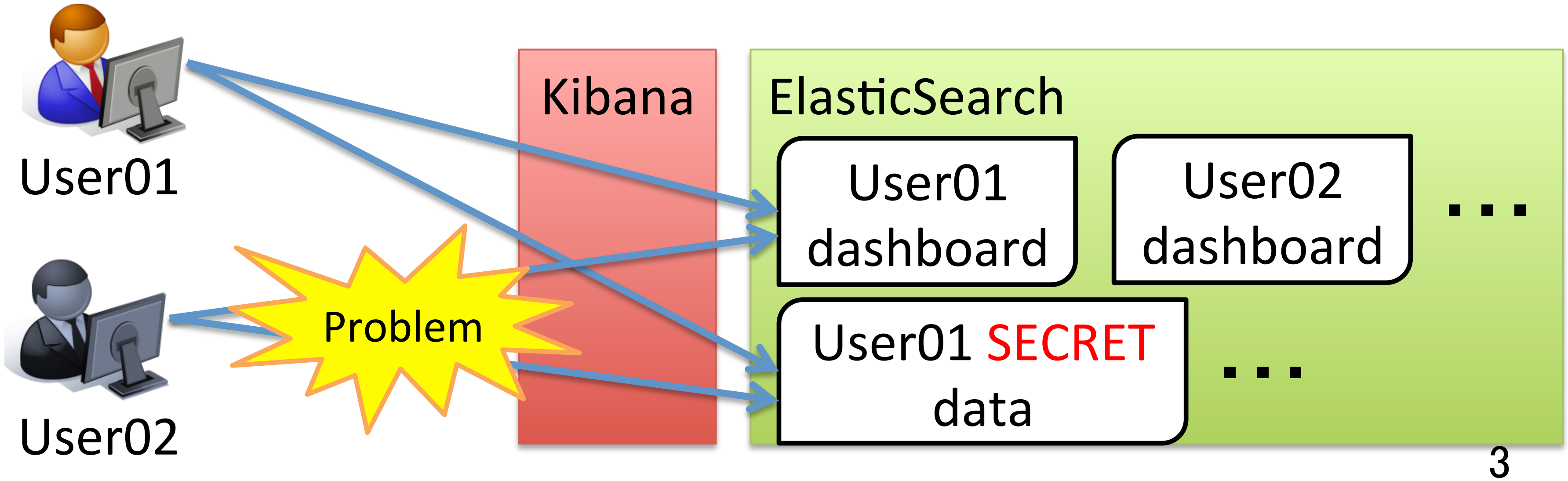
- Provide a great monitoring platform



Dashboards are stored in ElasticSearch

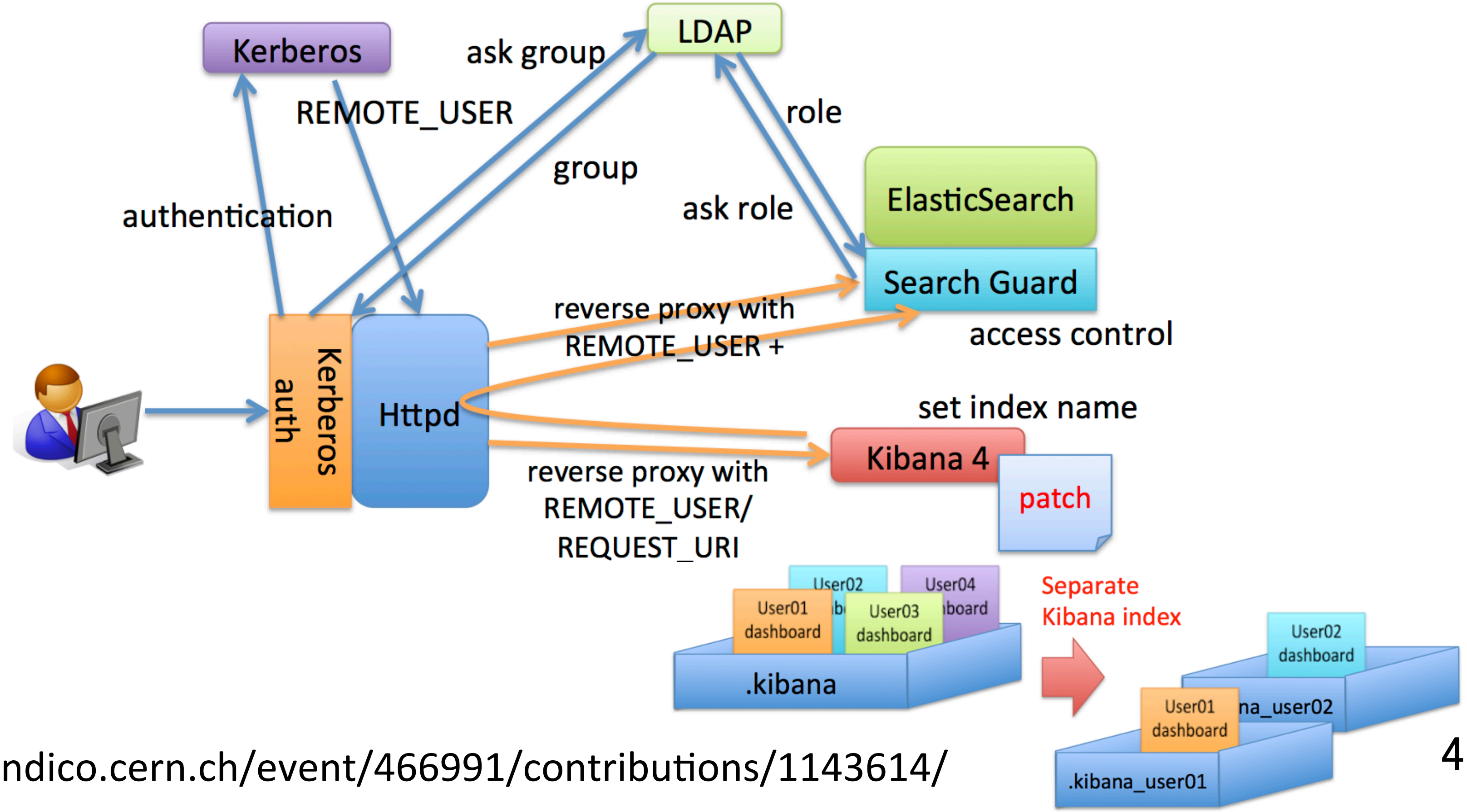
Motivation

- Kibana + ElasticSearch lack access control feature
- Multiple users/groups use single Kibana + ElasticSearch
  - – Any user can access to all ElasticSearch data
  - – Need access control



Last HEPiX: We Provided a Solution

- Kerberos authentication integration
- Kibana patch enabled to user/group based dashboard separation
- Search Guard enabled user/group based ElasticSearch access control



https://indico.cern.ch/event/466991/contributions/1143614/

Catch up with the Fast-paced Developments

Our target versions

	Last HEPiX	This HEPiX	Next HEPiX?
Kibana	4.1	4.5	5.x
ElasticSearch	1.5	2.3	5.x
Search Guard	0.5	2.3	5.x

- The Kibana patch is no longer adaptable
  - Need to develop new one
- Current Search Guard configuration/usage is completely different from the old one
  - Need investigation

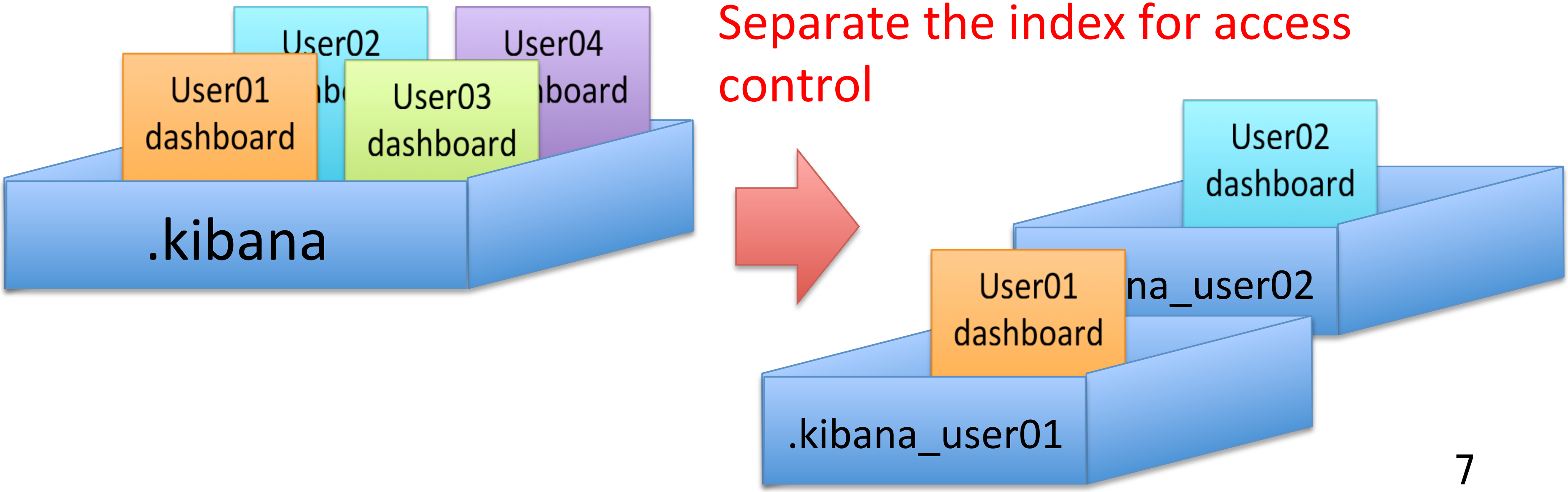
In This Talk

- Report on our latest R&D experience updated from the last HEPiX
  1. Development of a Kibana plugin for dashboard separation
  2. Investigation and contribution for Search Guard
  3. Development of a Flume patch for SSL connection
  4. Measurement of Search Guard-ed ElasticSearch performance

1. Development of a Kibana plugin: Motivation

- Problem
  - 1 Kibana instance uses only 1 Kibana index (1 database)

➡ All user's dashboards are stored in the same index

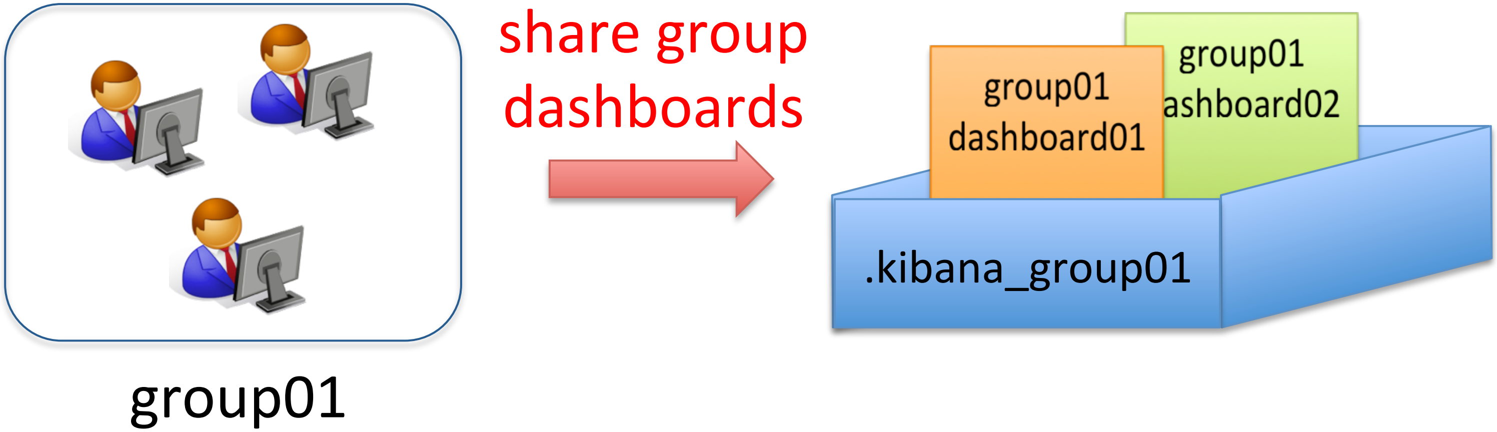


Separate the index for access control

1. Development of a Kibana plugin: Motivation

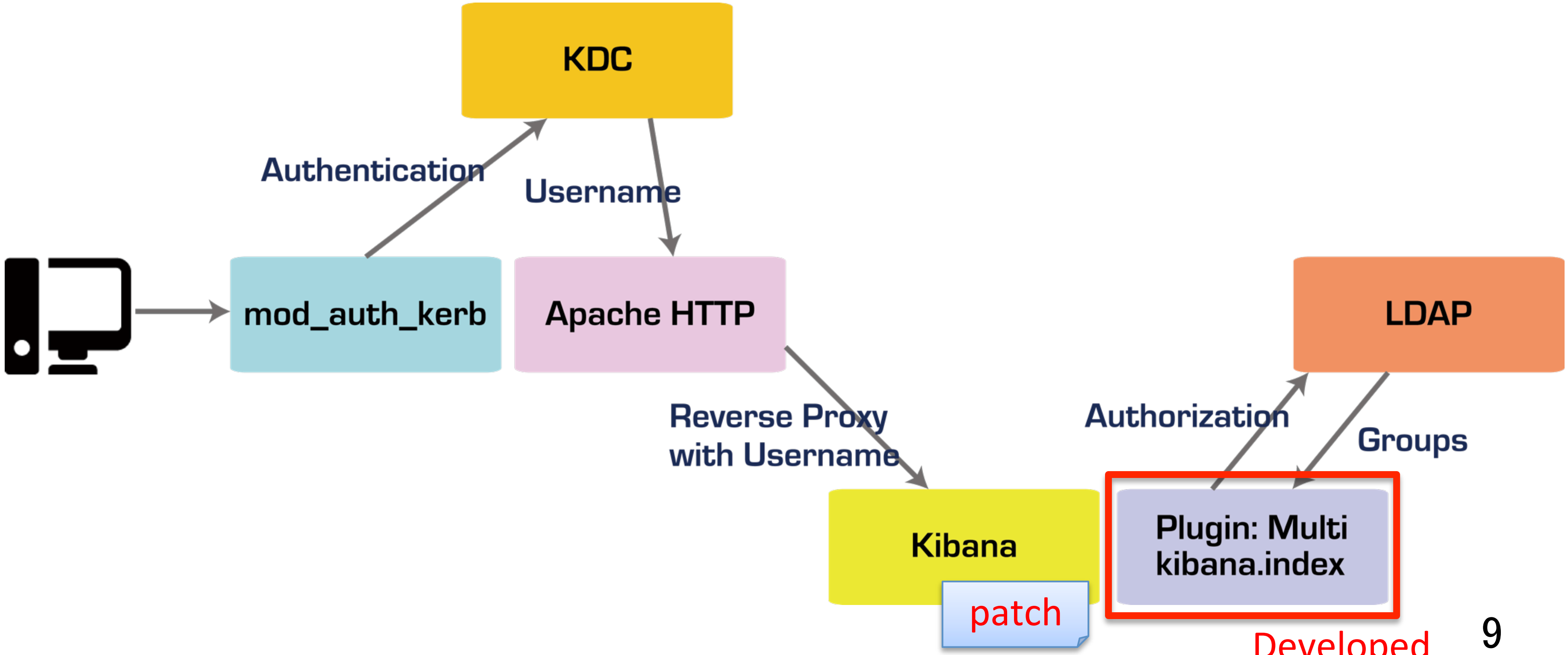
- Group based Kibana Index separation is useful

⇨ Users can share a Kibana index among a group

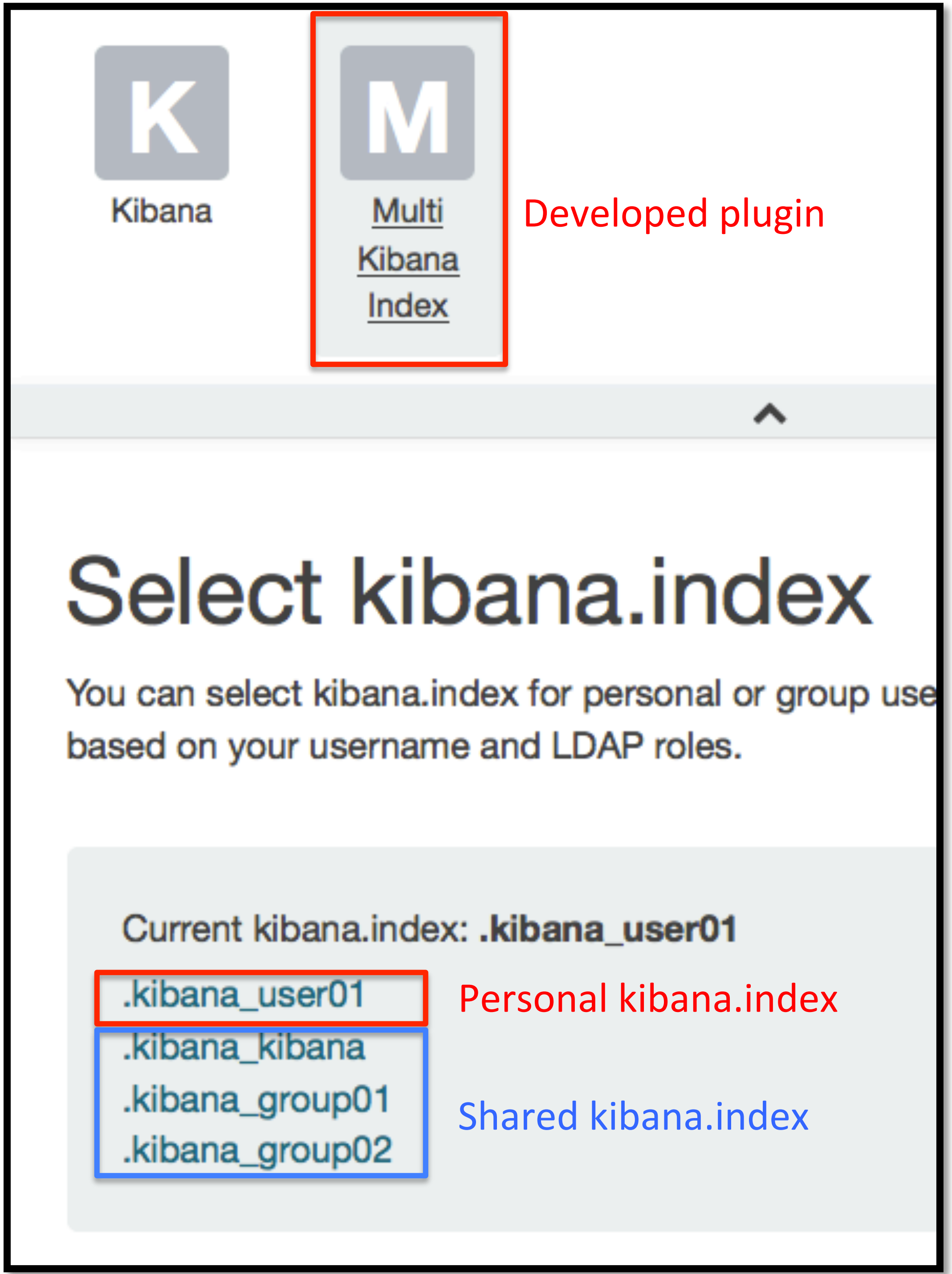


Development of a Kibana plugin

- Fetches user's groups from LDAP and displays available Kibana index list on Kibana interface
  - Personal Kibana index, shared Kibana indices
- User can switch Kibana index depending on the situation



Developed



Developed plugin

Select kibana.index

You can select kibana.index for personal or group use based on your username and LDAP roles.

Current kibana.index: **.kibana_user01**

.kibana_user01

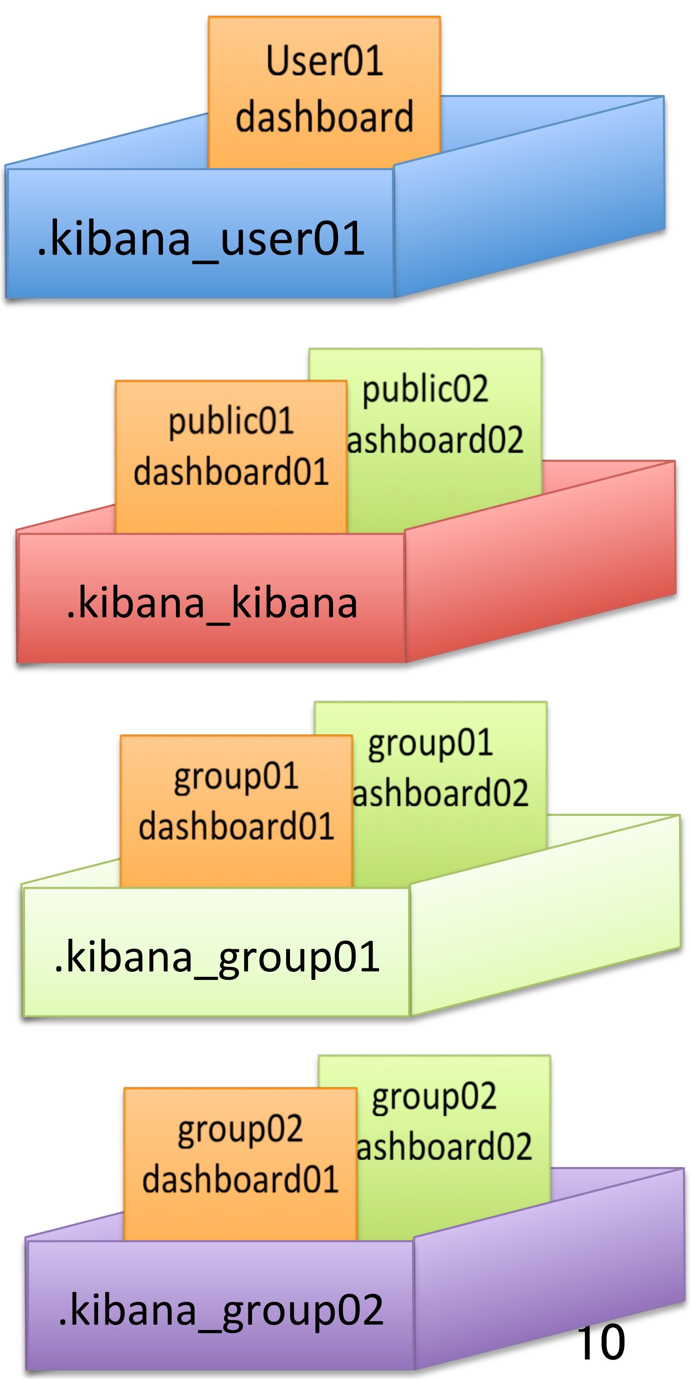
.kibana_kibana

.kibana_group01

.kibana_group02

Personal kibana.index

Shared kibana.index

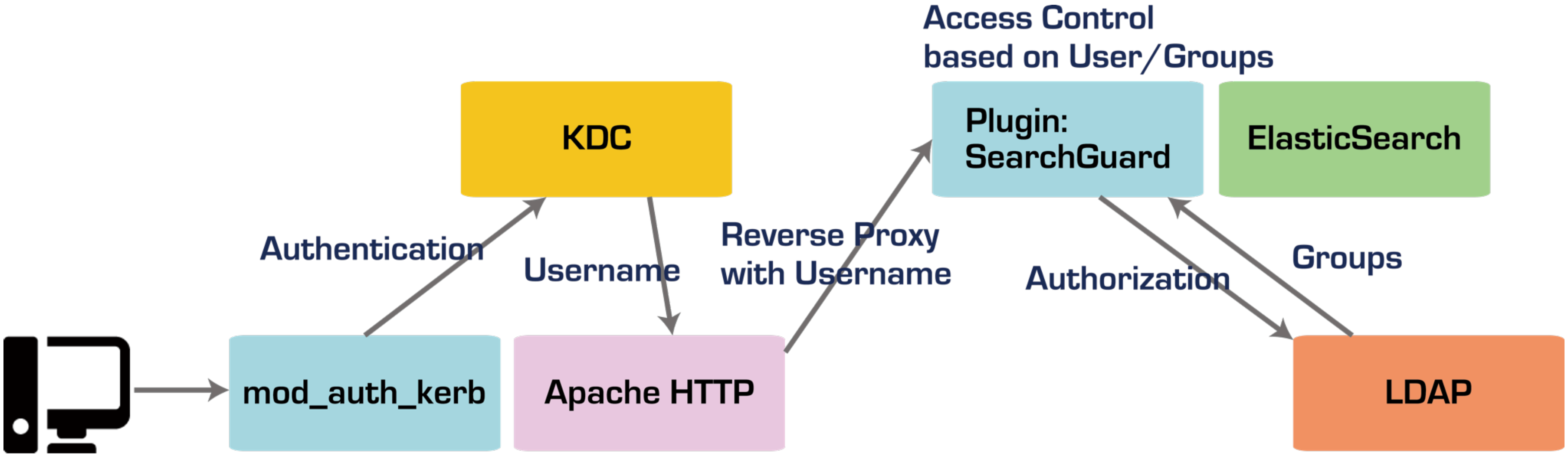


2. Investigation of Search Guard

- ElasticSearch plugin
  - Flexible REST/transport layer access control based on user/group
  - Supported by Floragunn
    - http://floragunn.com/searchguard
    - https://github.com/floragunncom/search-guard
  - Dual licensed
    - All core features are available free of charge
    - Enterprise license provides additional features

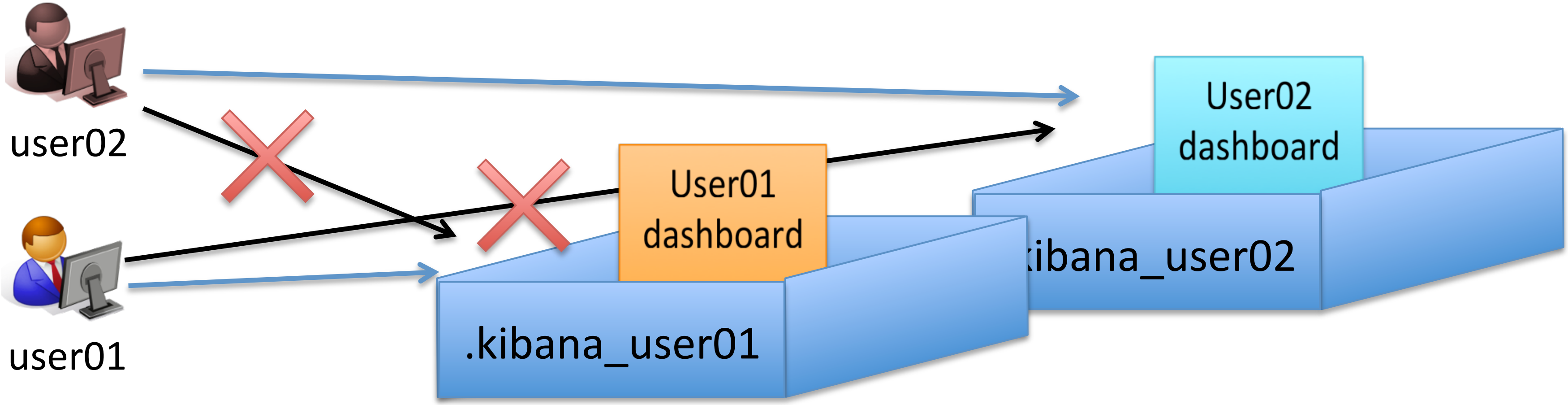
Search Guard + LDAP Authorization

- Search Guard supports multiple auth back-ends
  - YAML files based configuration
- We use proxy based authentication and LDAP authorization features for user/group based access control
- Our solution works fine with new Search Guard



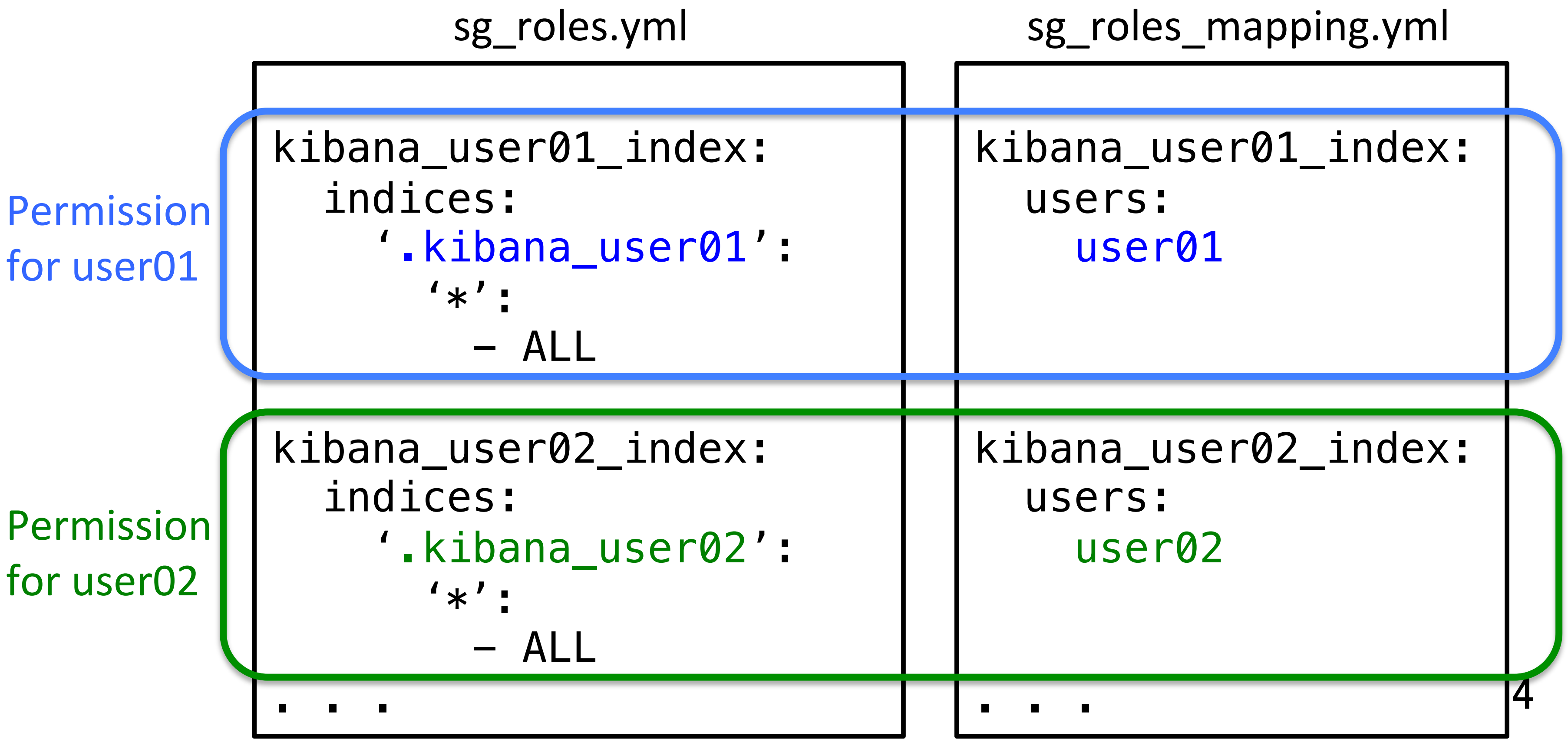
Contribution for Search Guard: Motivation

- In the case that each user has own index and each index allows access only from the owner...



Contribution for Search Guard: Motivation

- Admin has to define permissions for every user
- Whenever new user is registered, admin has to add permission



sg_roles.yml

```
kibana_user01_index:
  indices:
    ‘.kibana_user01’:
      ‘*’:
        - ALL

kibana_user02_index:
  indices:
    ‘.kibana_user02’:
      ‘*’:
        - ALL
. . .
```

sg_roles_mapping.yml

```
kibana_user01_index:
  users:
    user01

kibana_user02_index:
  users:
    user02
. . .
```

Permission for user01

Permission for user02

Development of a Search Guard Patch

- Enables to set username variable in configuration file and releases the admin from the troublesome task
- Has been merged to upstream

sg_roles.yml

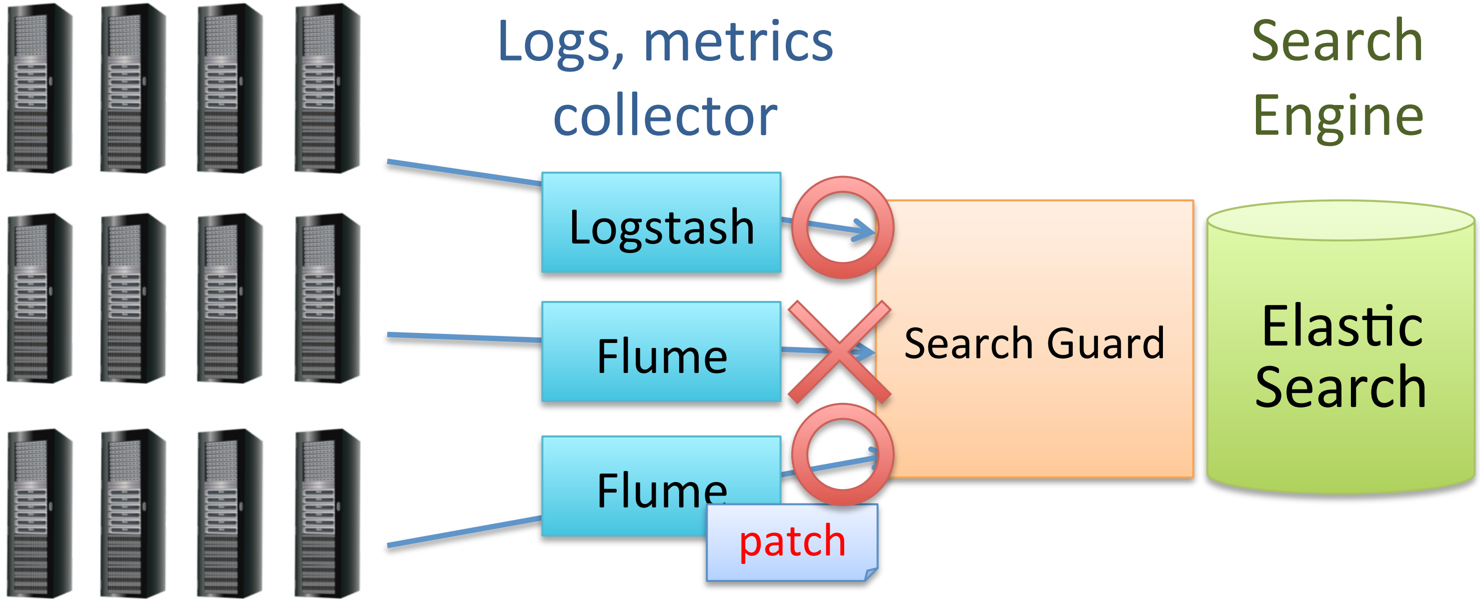
```
kibana_own_index:
  indices:
    ‘.kibana_${user_name}’:
      ‘*’:
        - ALL
```

sg_roles_mapping.yml

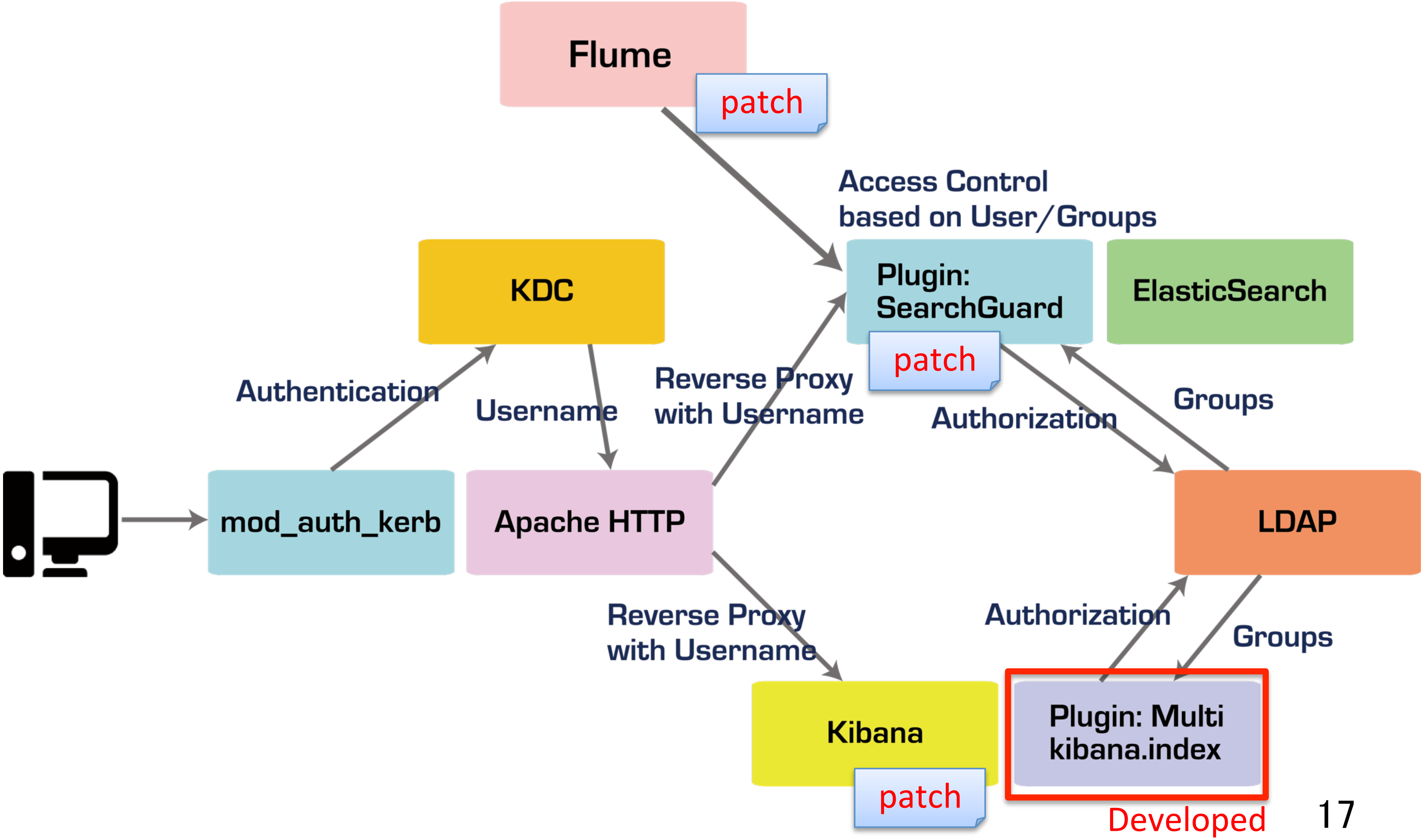
```
kibana_own_index:
  users:
    ‘*’
```

3. Development of a Flume Patch

- Flume connection is refused by Search Guard because Search Guard requires SSL connection
  - Flume only can do plan text connection
- Developed a patch to be able to support SSL



Overview of Our Updated Solution

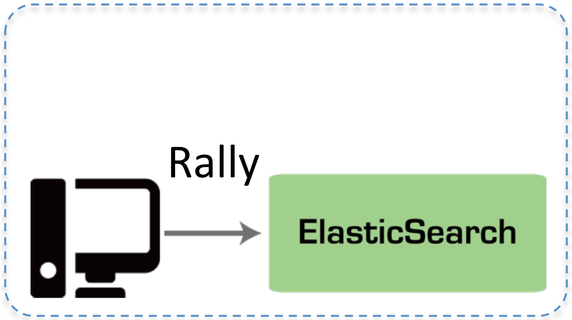


4. Measurement of Search Guard-ed ElasticSearch Performance by Rally

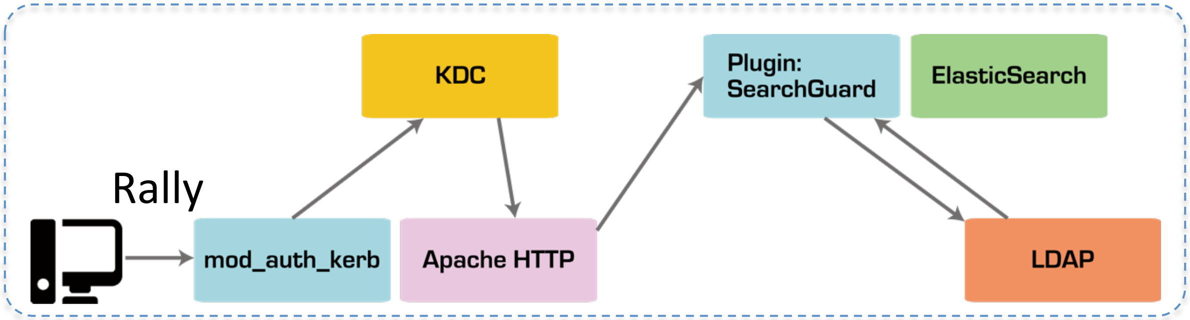
- What is Rally?
  - Benchmarking tool for ElasticSearch
    - https://www.elastic.co/blog/announcing-rally-benchmarking-for-elasticsearch
    - https://github.com/elastic/rally
  - Measures indexing throughput, query latency, aggregation latency, stats latency, etc...
  - Provides a few default scenarios and user can define customized one

Test Scenario

- Used Rally default scenario named “geonames”
  - Data source: http://www.geonames.org/
    - Provides geographical dataset
  - Indexes 8.6M documents (total 2.8GB) using 8 client threads and 5000 docs per bulk request against ElasticSearch
- Compared performance between normal ElasticSearch and Search Guard-ed ElasticSearch

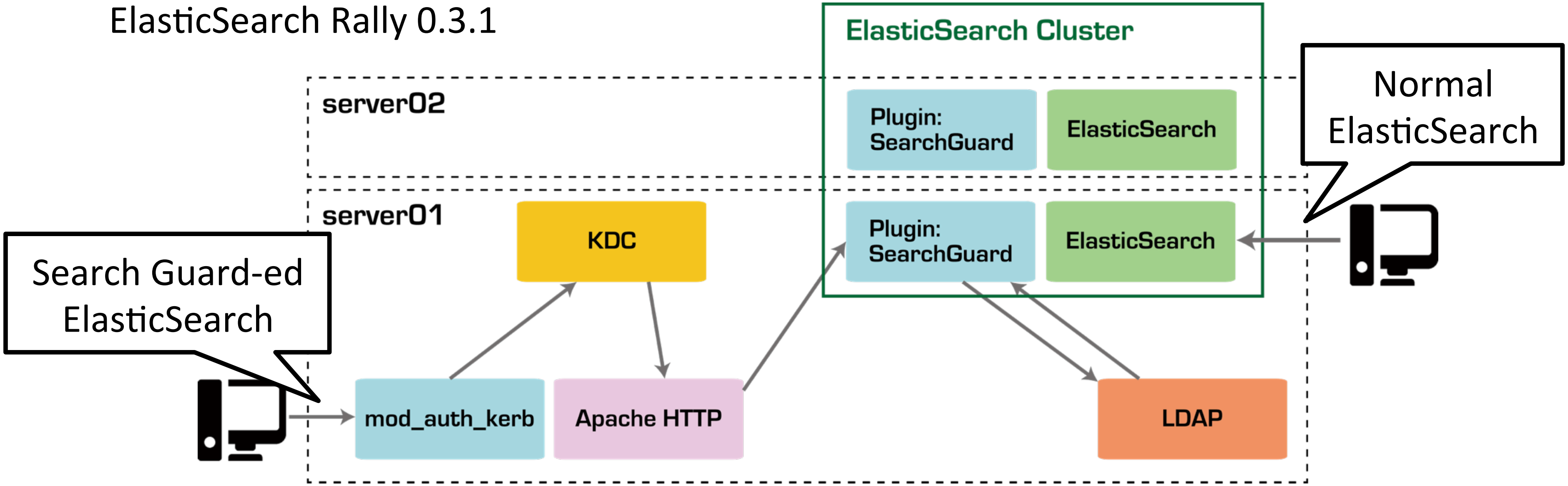


VS



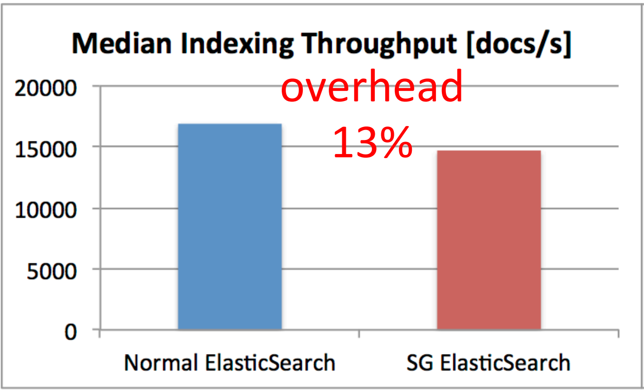
Test Environment

ElasticSearch Rally 0.3.1

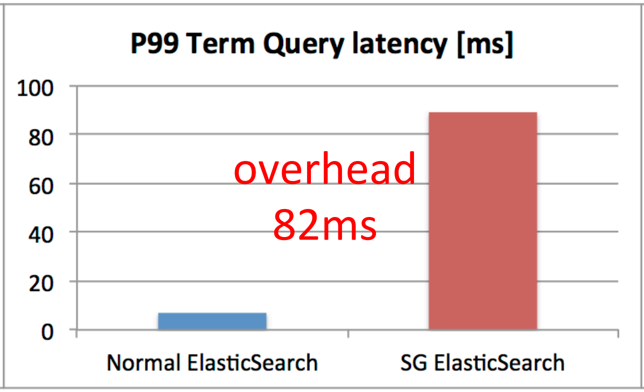


	server01 and server02
OS	CentOS 7
CPU	AMD Opteron 6212 2.6GHz 8 cores
RAM	8 GB
ElasticSearch	2.3.4
Search Guard	2.3.4

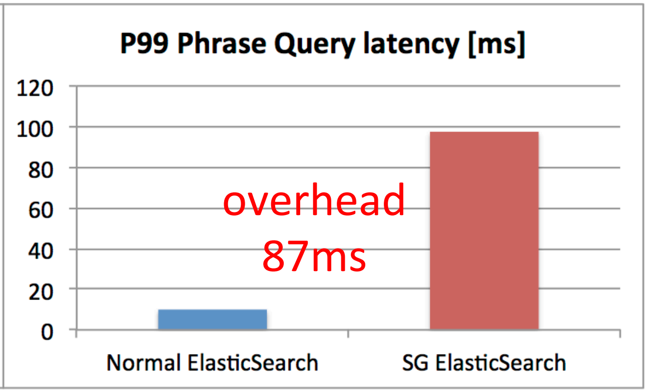
Results


Median Indexing Throughput [docs/s]

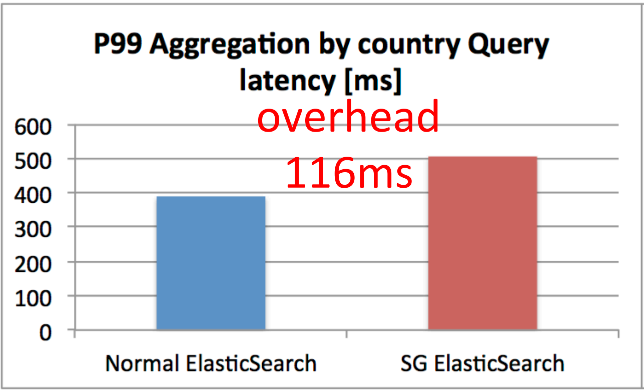
overhead 13%


P99 Term Query latency [ms]

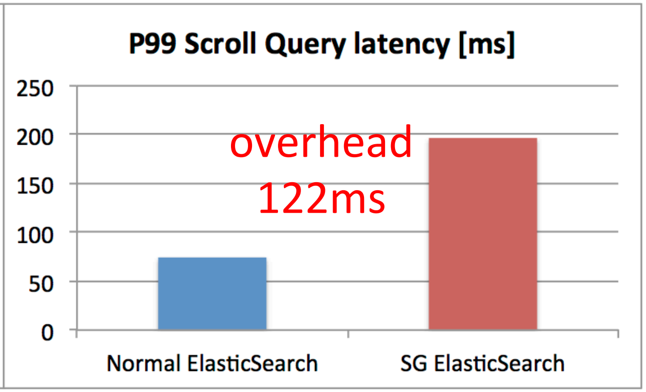
overhead 82ms


P99 Phrase Query latency [ms]

overhead 87ms


P99 Aggregation by country Query latency [ms]

overhead 116ms


P99 Scroll Query latency [ms]

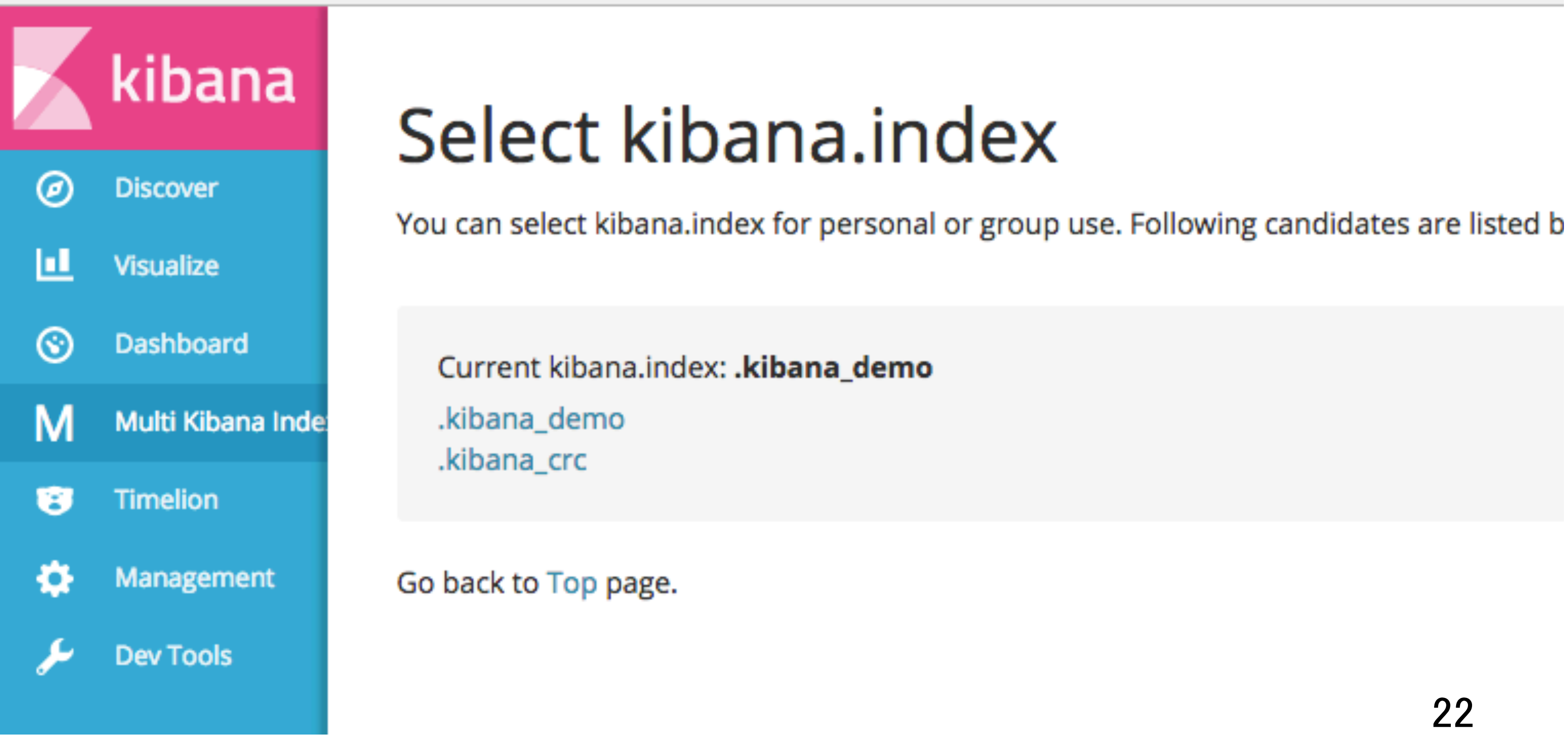
overhead 122ms

- P99: 99th percentile

- Overhead of each query: 80～120ms
  - Kerberos authentication
  - Reverse proxy
  - LDAP lookup
  - Search Guard access control

Future Work

- Adapting our solution to Kibana 5 and ElasticSearch 5



Select kibana.index

You can select kibana.index for personal or group use. Following candidates are listed b

Current kibana.index: **.kibana_demo**

.kibana_demo
.kibana_crc

Go back to Top page.

Summary

- In multi-user environment, user/group based access restriction and dashboard separation are necessary for secure use of Kibana and ElasticSearch
- We reported on our latest R&D experience in securing the services:
  1. Developed Kibana plugin allows user to switch Kibana index depending on the situation
  2. Our solution works fine with new Search Guard and our patch for more flexible configuration has been merged to the upstream
  3. Our Flume patch enables to push data to Search Guard-ed ElasticSearch
  4. Effect on performance of Search Guard-ed ElasticSearch:
     - Overhead of indexing throughput: 13%
     - Overhead of each query: 80〜120ms

Github References

- Patched Kibana
  - https://github.com/wtakase/kibana/tree/4.5-multi-kibana-indices-with-plugin
- Kibana plugin: multi kibana.index
  - https://github.com/wtakase/multi-kibana-index
- Search Guard Patches (Merged)
  - Support configurable OID
    - https://github.com/floraguncom/search-guard/pull/168
  - Use username variable at indices sections in sg_roles.yml
    - https://github.com/floraguncom/search-guard/pull/169
- Flume ElasticSearchSink2 for Search Guard
  - https://github.com/wtakase/ElasticsearchSink2/tree/search-guard-ssl