Mohammed Aboullaite | 25 Feb 2018 | 3 min read

Secure kibana dashboards using keycloak

Kibana is an open source data visualization plugin for Elasticsearch. It provides visualization capabilities on top of the content indexed on an Elasticsearch cluster. Users can create bar, line, and scatter plots, or pie charts and maps on top of large volumes of data.

But while using kibana, you'll sooner or later face the need to secure it. Kibana itself doesn't support authentication or restricting access to dashboards and we need to use either the official solution from elastic: xpack security, or alternative solutions like search-gard or nginx.

This post, adds another option based on the open source identity and access management from Redhat: keycloak

Keycloak

Keycloak is a security server that allows for outsourcing and delegating all the authentication and authorization aspects. It's open-source, flexible, and agnostic of any technology, it is easily deployable/adaptable in its own infrastructure.

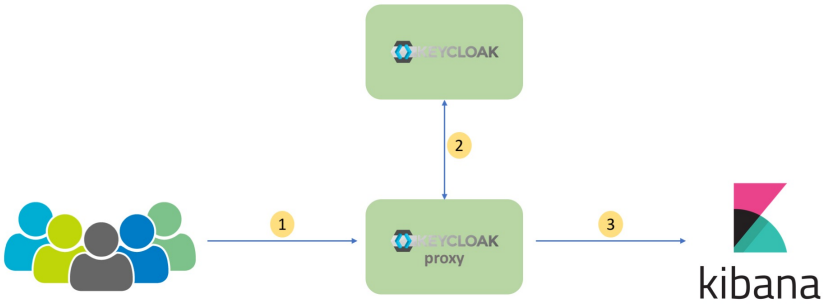
Management system, user federation for third parties like LDAP and more.

Keycloak also has an HTTP(S) proxy that we can put in front of web applications and services that don't have a built in authentication.. we can set up URL filters so that certain URLs are secured either by browser login and/or bearer token authentication.

Obviously, we'll be using the keycloak proxy to secure access to our kibana dashboards

How it works

The mode of operation is summed up in 3 simple steps:



1. External traffic is directed to the keycloak proxy. The proxy decides based on it configuration if the destination needs authentication.
2. The keycloak Proxy work together with Keycloak and redirects the user to the authentication server so the user can login.
3. After a successful login the proxy forwards the user to kibana instance.

Aboullaite Med Failure sucks but instructs!

Below is a `docker-compose` file describing our 5 services:

* `postgres` : as the main database for keycloak
* `keycloak` : our IAM server
* `keycloak-proxy` : The http proxy to secure access to kibana
* `elasticsearch` : elasticsearch instance
* `kibana` : kibana instance

```yaml
version: '3'

services:
  postgres:
      image: postgres
      container_name: postgres
      volumes:
        - postgres_data:/var/lib/postgresql
      environment:
        POSTGRES_DB: keycloak
        POSTGRES_USER: keycloak
        POSTGRES_PASSWORD: password
  keycloak:
      image: jboss/keycloak:3.4.3.Final
      container_name: keycloak
      environment:
        POSTGRES_PORT_5432_TCP_ADDR: postgres
        POSTGRES_DATABASE: keycloak
        POSTGRES_USER: keycloak
        POSTGRES_PASSWORD: password
        KEYCLOAK_USER: admin
        KEYCLOAK_PASSWORD: password
      ports:
        - 8080:8080
      depends_on:
        - postgres
  keycloak-proxy:
      image: jboss/keycloak-proxy:3.4.2.Final
      container_name: keycloak-proxy
```

Aboullaite Med Failure sucks but instructs!

```
          HTTPS_PORT: 8443
          BASE_PATH: /
          REALM_NAME: kibana
          AUTH_SERVER_URL: http://keycloak:8080/aut
          CLIENT_ID: kibana
          ROLE_ALLOWED: user
          SSL_REQUIRED: external
        volumes:
          - $PWD/conf:/opt/jboss/conf
        ports:
          - 8180:8180
        depends_on:
          - keycloak
    elasticsearch:
        image: docker.elastic.co/elasticsearch/elas
        container_name: elasticsearch
        environment: ['http.host=0.0.0.0', 'transpc

    kibana:
        image: docker.elastic.co/kibana/kibana-oss:
        container_name: kibana
        environment:
          - ELASTICSEARCH_USERNAME=elasticsearch
          - ELASTICSEARCH_PASSWORD=elastic
          - ELASTICSEARCH_HOST=elasticsearch
          - ELASTICSEARCH_PORT=9200
        depends_on: ['elasticsearch']

  volumes:
    postgres_data:
        driver: local
```

Note that the config directory mounted with `keycloak-proxy` contains the `proxy.json` file, the configuration file needed by the proxy. See the proxy documentation for more details.

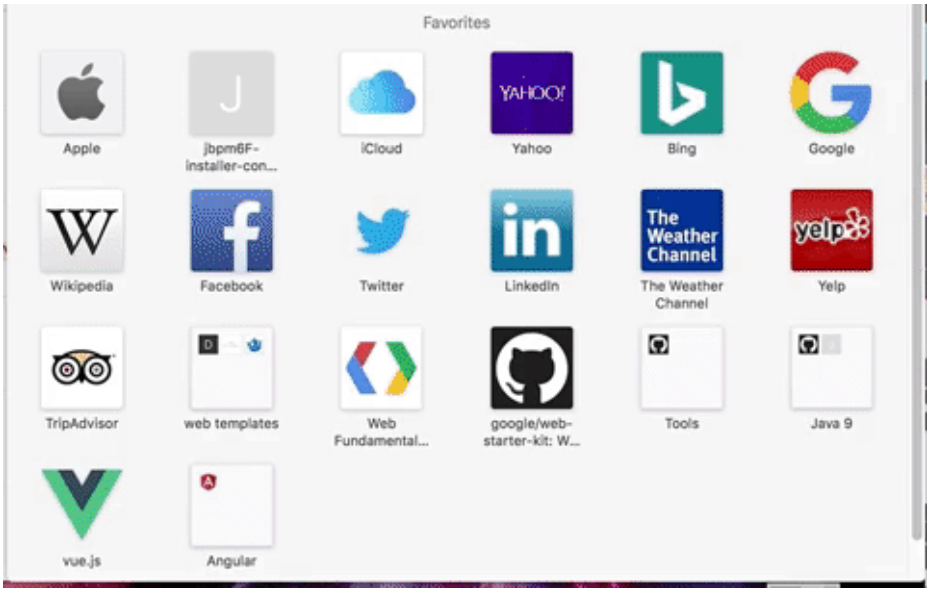
Aboullaite Med Failure sucks but instructs!

```
    "bind-address": "0.0.0.0",
    "http-port": "${env.HTTP_PORT}",
    "https-port": "${env.HTTPS_PORT}",
    "applications": [
        {
            "base-path": "${env.BASE_PATH}",
            "adapter-config": {
                "realm": "${env.REALM_NAME}",
                "auth-server-url": "${env.AUTH_SE
                "public-client": true,
                "resource": "${env.CLIENT_ID}",
                "ssl-required": "${env.SSL_REQUIF
            },
            "constraints": [
                {
                    "pattern": "/*",
                    "roles-allowed": [
                        "${env.ROLE_ALLOWED}"
                    ]
                }
            ]
        }
    ]
}
```

Once the services are up, we need to login to the keycloak admin console, add a new `kibana` Realm, create a `user` role and add some users to it (same as described on your `proxy.json` file). We also have to add a new `kibana` Client and a Valid Redirect URI, something like `http://keycloak-proxy:8180/*` . you can find more details on how to setup these steps on the keycloak documentation.

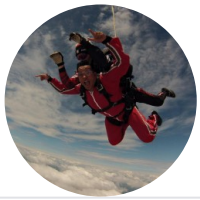
Voila! We've successfully restricted access to our kibana instance

Aboullaite Med Failure sucks but instructs!



The complete code is available on github.

Devops elk keycloak



Mohammed Aboullaite

ALSO ON **ABOULLAITE BLOG**

3 years ago • 1 comment
Speed up your java application Docker ...

4 years ago • 2 comments
Reactive Streams example - Java 9

Aboullaite Med Failure sucks but instructs!

Join the discussion…

LOG IN WITH

OR SIGN UP WITH DISQUS ?

Name

Lukas Mrtvy • 4 years ago

Nice.
Clear way to do it should be a Nginx reverse proxy with auth_request module in front of keycloak-proxy, just like oauth2 does it:
https://github.com/bitly/oa...
They are preparing GO alternative to JAVA keycloak-proxy, which will be much more lightweight

1 • Reply • Share ›

Nabil Servais • 4 years ago

Nice solution, but it doesn't work. I have an http 403 code after the login.

1 • Reply • Share ›

Mohammed Aboullaite Mod → Nabil Servais • 4 years ago

Make sure to assign role to user and that this role is the one declared on your `proxy.json` file

• Reply • Share ›

Bhargav Patel → Mohammed Aboullaite • 3 years ago

I had same role in proxy.json but getting same error

1 • Reply • Share ›

X Kongphop S't → Mohammed Aboullaite • 2 years ago

Can you show keycloak config?

• Reply • Share ›

Terence Monteiro • 5 months ago

Tried it but the keycloak proxy service has image as jboss/keycloak-proxy:3.4.3.Final and that does not

Pramod • 2 years ago

Tried the similar setup, with direct elk docker and non-docker keycloak setup, but receiving 503 error. proxy (http://localhost:8180) redirects to keycloak, authenticates and redirects to proxy, then again proxy has internal redirection and fails

Redirection from keycloak server to this: http://localhost:8180/?state=03e6c8f5-b849-4930-9631-5640632d8b6c&session_state=f6de7178-dafd-402b-b4b6-f4af4af0de2d&code=89009ad2-ebfb-4aab-8f2e-f522cafa7b94.f6de7178-dafd-402b-b4b6-f4af4af0de2d.20201f19-1bbe-432e-931b-7151547de576

Above redirects to below URL with 503 error: http://localhost:8180/?session_state=f6de7178-dafd-402b-b4b6-f4af4af0de2d

Any help would be greatly appreciated

see more

Reply • Share ›

Bhargav Patel • 3 years ago

Thanks for tutorial.

Reply • Share ›

Adipa Wijayathilaka (Boarnoah) • 4 years ago

Interesting solution.At the end of the day you're using Key Cloak here at a similar level of functionality to using reverse proxying Kibana and hiding it behind a

Aboullaite Med Failure sucks but instructs!