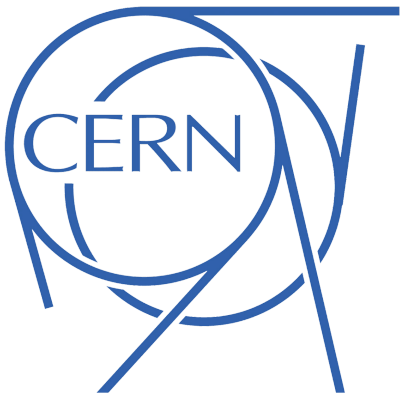




# Evaluate Apache Ranger to provide comprehensive security across the CERN Hadoop ecosystem

CERN Summer Student Program 2023

Clément Lucas

Supervisors: Emil Kleszcz and Miltiadis Gialousis

Department IT-DA-DS

June 26, 2023 - September 15, 2023

# Table of contents:

# Abstract

This report provides an overview of a 12-week internship project conducted at CERN, focusing on the evaluation and implementation of Apache Ranger to enhance security management within the CERN Hadoop ecosystem. The Hadoop ecosystem, designed to manage large data volumes, was integrated with Apache Ranger to achieve fine-grained access control, centralised security management, LDAP integration, audit tracking, and policy enforcement. This report outlines the steps taken to deploy Apache Ranger manually, emphasising the challenges faced during installation, configuration, and integration with CERN's LDAP system. It discusses plugin setup for critical Hadoop components and the storage of audit logs. It concludes by highlighting successful accomplishments and proposing future improvements to enhance Apache Ranger's suitability for CERN's production environment.

# 1. Introduction

## 1.1. Apache Ranger

The Hadoop ecosystem is a collection of open-source software components and tools designed to store, process, and analyse large volumes of data in a distributed computing environment. It was initially created to address the challenges of working with big data, with which CERN is very familiar.

This ecosystem is made up of several critical production components, to which I had to add Apache Ranger. This software has many benefits my department was looking after when installed in a Hadoop ecosystem:

- Helps to manage fine-grained access control over Hadoop and related components (Apache HDFS, Apache HBase, etc.)
- Offers a centralised security framework
- Has a user-friendly interface to easily manage and enforce policies within the Hadoop ecosystem
- Gives the possibility to connect to an LDAP server to apply policies to members (CERN employees and related accounts) and groups stored in ActiveDirectory at CERN

- Enables audit tracking, access control, and policy analytics for deeper protection of this environment
- Can provide the ability to delegate administration of certain data to other group owners, with an aim of decentralising data ownership

## 1.2. My internship

This document provides a report on the project I participated in during my 12-week internship at CERN.

The project I was working on was titled "Evaluate Apache Ranger to provide comprehensive security across the CERN Hadoop ecosystem".

So the aim of my internship was to install Apache Ranger on a development environment to verify how it integrates with the Hadoop Ecosystem at CERN namely HDFS, Yarn, HBase, MapReduce, and Spark History Server. After that my task was to prepare some use cases and policies for the existing production setup making use of the Apache Ranger's features.

If I'd had the time, Apache Knox as a reverse proxy would have been evaluated, with a prototype in a DEV environment especially focusing on the Knox SSO feature, but the challenges encountered during the deployment of Apache Ranger did not allow me to do so.

# 2. Deploying Apache Ranger

As I was asked to install manually Apache Ranger, without Docker or any other deployment tool, I had to follow some steps to have a ready to use Apache Ranger instance:

- Download the Apache Ranger distribution package from the official website or from the Github repository
- Build Apache Ranger on a cluster node. This involves extracting the downloaded package, configuring the necessary environment variables and building the software.
- Set up a database (e.g., MySQL, PostgreSQL, Oracle) to store Ranger's metadata and policies. Configure the database connection properties in Ranger's configuration.
- Configure, start and access Ranger Admin UI. This web-based interface provides a user-friendly way to configure and control Ranger.
- Set up Ranger User Sync to synchronise user and group information between our LDAP server and Ranger.
- Configure Apache Ranger to work with our components. This includes setting up connectors, such as the HDFS plugin, and configuring policies for access control.
- Define and manage policies that specify who has access to what resources and what actions they can perform. These policies are crucial for access control.
- Configure Ranger to generate and store audit logs, which are crucial for monitoring and compliance purposes.
- Ensure that the Ranger policies are actively enforced.

I will now give details on some of these steps.

## 2.1. Building process

Firstly, we had to choose which version to build. In fact, the latest released version of Apache Ranger is the 2.4.0 but the development team is already working on the 3.0.0 , considered as a snapshot for the moment. At some point, we were switching versions and building Apache Ranger all over again to know if some issues were fixed on the latest update but in general we were focusing on 2.4.0.

Apache Ranger is developed in Java, so to build the software I was using Maven. During the process, I had to deal with the lack of up-to-date documentation for Apache Ranger and had to find out which versions to specify to manage dependencies in the CERN environment for the existing production components. Moreover, there were some plugins that were definitely not needed to our ecosystem, leading to such a command at the end:

```
mvn clean compile package install -DskipTests=true -Dspotbugs.skip=true
-Dchkstyle.skip=true -Dassembly.plugin.version=3.1.0
-Dhadoop.version=3.3.0 -Dhbase.version=2.3.4 -Dzookeeper.version=3.6.1
-Dhive.version=3.0.0 -Dmysql-connector-java.version=8.0.28
-pl '!plugin-ozone, !plugin-solr, !plugin-nifi, !plugin-nifi-registry,
!plugin-kudu, !plugin-kms, !ranger-ozone-plugin-shim, !storm-agent,
!ranger-storm-plugin-shim, !ranger-solr-plugin-shim,
!ranger-atlas-plugin-shim, !plugin-atlas, !plugin-kylin,
!ranger-kylin-plugin-shim'
```

## 2.2. Installing Apache Ranger admin

When it comes to configuring the admin part, the lack of documentation and my lack of personal experience were particularly noticeable and I had to look and investigate around to come to conclusions about the necessary configurations.

Some aspects I had to overcome were the use of Kerberos credentials, since the cluster is set to use theKeberbos authentication, and also the filters defined to be sure that the connection with the CERN LDAP instance was made properly and then that the members and groups from ActiveDirectory were sorted and filtered in an expected way in Apache Ranger

I also found that the software is more resource intensive with this setup, so I had to increase the maximum allocated memory in the configuration from the default 1g of java heap size and this also led to migration from one development cluster to another where we had more resources to run on using the CERN virtual machines from OpenStack.

## 2.3. Integrating with CERN LDAP

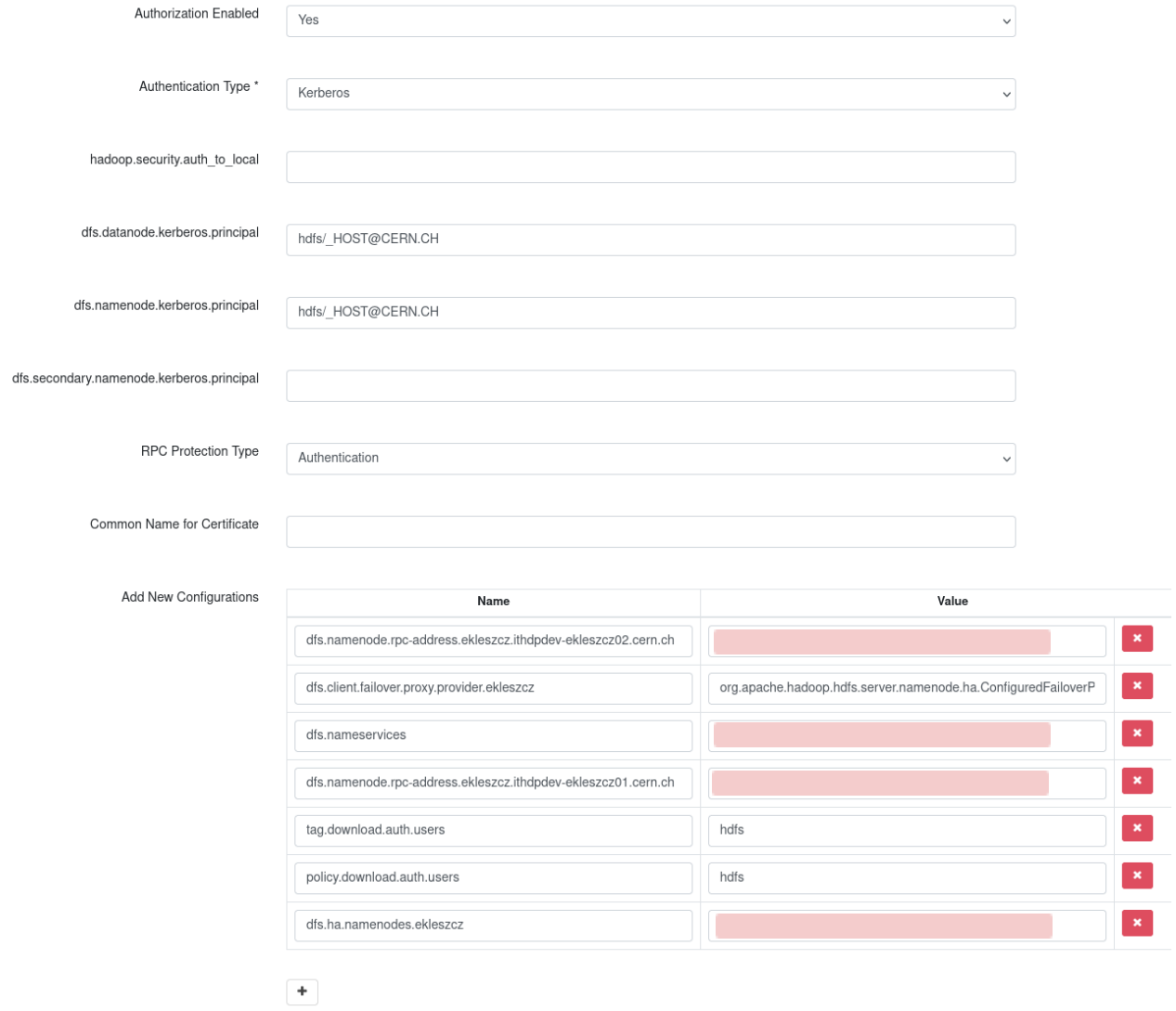
What had to be taken into account is the fact that the Hadoop team developed its own jar file to manage their LDAP nested groups mappings: LdapNestedGroupsMapping.jar
So to be able to synchronise with the LDAP instance, I had to include this jar file in all my lib folders to make sure that the implementation worked well.

Also, I couldn't use the xldap.cern.ch instance, the most popular endpoint for LDAP instance at CERN, because it can only answer to anonymous requests and Apache Ranger doesn't support anonymous bindings as it has been discovered during the tests. So I had to adapt and use another endpoint instead, cerndc.cern.ch, that allows bindings with basic authentication (also with Kerberos but this was not configured).

## 2.4. Using plugins to access Hadoop components

I had the possibility to install the plugins corresponding to the 3 most important plugins linked to the CERN Hadoop ecosystem model: mainly HDFS, Yarn, and HBase/Zookeeper.

In the process, it was necessary to add different additional settings in each plugin configuration, for example to be able to use 2 URLs for the HDFS namenodes (in HighAvailability mode used at CERN) such as shown below:




*Picture 1: HDFS plugin configuration*

## 2.5. Audit logs

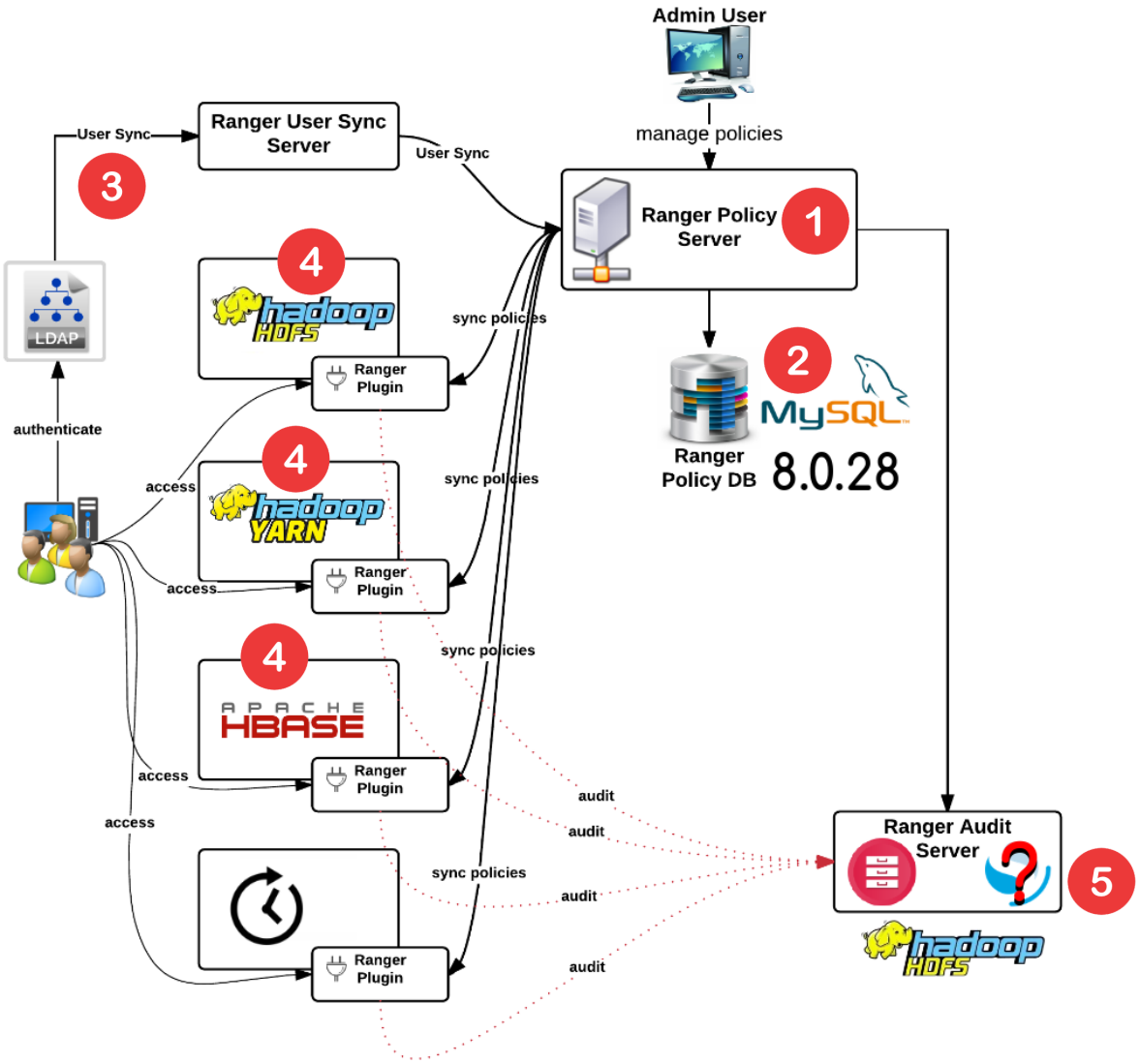
In the context of Apache Ranger, audit logs refer to the detailed records or logs that represent all relevant activities and events related to access control, authorization, and policy enforcement within the Apache Ranger security management framework. When using a plugin, it's possible to define what would be the kind of audit logs being saved, for example it would be possible to save only logs stating that someone got a denied access to a resource. Then for each policy it's possible to decide if the impact of the policy should be audited or not.

At the beginning, I was aiming at using Opensearch to store audit logs since it provides a complete web interface with the possibility to have a lot of plots in dashboards depicting the activity of a service and also it was already being used in my department as a good indexing storage for structured documents. I've spent a lot of time trying to implement it since there were some compatibility issues with some dependencies and I had to fix and also report an issue about the date format pattern used for the logs that was not the same in Opensearch and on the Apache Ranger admin-side. However, even after having spent a considerable amount of time, there were still errors in the admin logs stating that some audit logs were not managed properly and thus not sent. In the end I was able to send only some of the logs to the corresponding OpenSearch index that I have created for that purpose. It is still not clear whether OpenSearch is fully supported by Apache Ranger and the next step would be to try to run this against an ElasticSearch instance however CERN is not supporting this commercial product anymore.

As it was not working as expected I had to fall back to another option. So I've started to save audit logs directly to HDFS and since that moment it has been working perfectly well. It's less convenient to access audit logs that way but at least we're sure about the reliability of this feature.

# 3. Conclusion

So to conclude my internship, this is how the Apache Ranger was like at the end:




*Picture 2: Schema of Apache Ranger implementation*

1. Apache Ranger WebUI was implemented and working seamlessly
2. The ranger Policy DB that we have set-up was a MySQL database, version 8.0.28 supported by the CERN DBoD service. This DB saves all the configured policies and all admin logs.
3. User Sync plugin was implemented, making it possible to synchronise Apache Ranger with the LDAP directories at CERN
4. 3 component plugins were configured and installed such as for: HDFS, Yarn and HBase. These plugins retrieve the policies about their components from the admin-side and implement them.
5. Audit logs from the plugins were ultimately stored in HDFS

# 4. Future Work

To ensure that Apache Ranger is suitable for production environments, several key elements can be addressed and overcome:

- Fixing Authentication with LDAP: Set up authentication with user account to the Ranger web UI. Currently it is done using only internal Apache Ranger accounts.
- Adding New Necessary Configurations to Puppet: Ensuring that Puppet, a configuration management tool, configurations cover all necessary aspects of Apache Ranger's setup and maintenance is important for scalability and ease of use.
- Fixing Missing Features in Current Plugins: Be sure to have autocompletion for Yarn plugin and thoroughly test all the plugins to make sure that all expected features are working.
- Testing HA and Failure Scenarios: Implementing high availability (HA) and robust testing through failure scenarios is vital. Ensuring that Ranger can gracefully handle system failures, maintain availability, and recover from issues without data loss is critical for production readiness.
- Exploring Other Plugins: Expanding plugin support for additional services commonly used in production environments would be beneficial. There are still some plugins available corresponding to that such as Kafka.
- Exploring roles in the admin web ui for outsourcing management of the policies to the respective project managers with proper access control between the projects.
- Presenting the work to the CERN user community and assuring sufficient level of support and documentation.

By addressing these key elements, Apache Ranger can become a more robust and reliable solution over time for securing and managing access to data in a CERN production environment. As proved with the example policies against production data this would greatly improve the management of the ACLs and thus the overall security of the critical Hadoop infrastructure at CERN.

# 5. Acknowledgments

I wanted to express my heartfelt gratitude for the incredible opportunity it has been to be a part of the CERN summer student program within the IT-DA-DS section.

These past 12 weeks have been an incredible journey of learning and discovery, so I would like to thank the IT-DA-DS section, which welcomed and integrated me, and also I owe it all particularly to my supervisors, Emil Kleszcz and Miltiadis Gialousis, and their support and mentorship.

# 6. References

## 6.1. References from CERN

- https://hadoop-admin-guide.web.cern.ch/
- https://hadoop-user-guide.web.cern.ch/hdfs/basicHDFS/

## 6.2. Other references

- Apache Ranger Github release tag 2.4.0
  https://github.com/apache/ranger/tree/release-ranger-2.4.0
- Quick start guide from Apache Ranger website
  https://ranger.apache.org/quick_start_guide.html
- Some FAQ on Apache Ranger website
  https://ranger.apache.org/faq.html
- Installing Maven 3.9.3
  https://www.tecmint.com/install-apache-maven-on-centos-7/https://www.tecmint.com/install-apache-maven-on-centos-7/

- Versions prerequisites
  https://docs.cloudera.com/HDPDocuments/HDP2/HDP-2.3.4/bk_installing_manually_book/content/installation_prerequisites_ranger.html

- About Hardware requirements
  https://docs.cloudera.com/cdp-private-cloud-upgrade/latest/release-guide/topics/cdpdc-ranger.html

- Configuring the Ranger Policy Administration Authentication Mode
  https://docs.cloudera.com/HDPDocuments/HDP2/HDP-2.6.1/bk_command-line-installation/content/configure-the-ranger-policy-administration-authentication-moades.html

- Providing Authorization with Apache Ranger
  https://docs.cloudera.com/HDPDocuments/HDP3/HDP-3.1.5/authorization-ranger/sec_authorization_ranger.pdf

- Ranger 2.2.0 Database Schema
  https://cwiki.apache.org/confluence/display/RANGER/Ranger+2.2.0+Database+Schema

- Installation instructions
  https://cwiki.apache.org/confluence/display/RANGER/Apache+Ranger+0.5.0+Installation#ApacheRanger0.5.0Installation-EnablingRangerHDFSPlugins

- HDFS Permissions Guide
  https://hadoop.apache.org/docs/stable/hadoop-project-dist/hadoop-hdfs/HdfsPermissionsGuide.html

- Elasticsearch audit logging schema
  https://github.com/apache/ranger/blob/release-ranger-2.4.0/security-admin/contrib/elasticsearch_for_audit_setup/conf/ranger_es_schema.json#52

- During building process, JAR entry pom.properties not found
  https://issues.apache.org/jira/browse/RANGER-4166?attachmentSortBy=dateTime