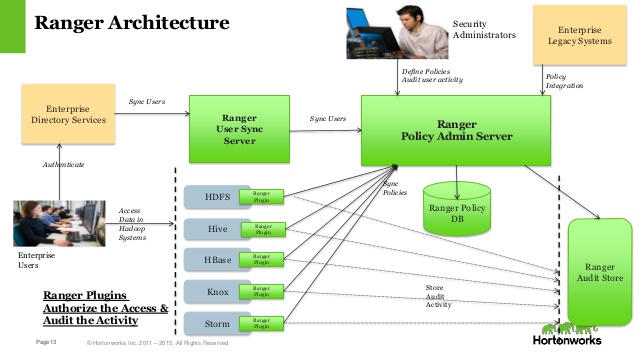
Apache Ranger

# 1.简介

Apache ranger是一个Hadoop集群权限框架，提供操作、监控、管理复杂的数据权限，提供一个集中的管理机制，提供基于yarn的Hadoop生态圈的所有数据权限，可以对Hive、HBase进行细粒度的数据访问控制，如下图所示：



Ranger系统架构图

Ranger的总体架构如上所示，主要包括以下三个组件：

1. Admin Server，以RESTFUL形式提供策略的增删改查接口，同时内置一个Web管理页面
2. Agent Plugin，嵌入到个系统的执行流程中，定期从AdminServer拉取策略，根据策略的执行访问决策树，并且定期记录审计。
3. User Sync，定期从LDAP/File中加载用户，上报给AdminServer
4. Ranger Audit Store，定期从Admin拉取策略，根据策略进行执行访问决策树，定期记录访问审计

通过操作Apache Ranger控制台，管理员可以轻松的通过配置策略来控制用于访问HDFS文件夹、HDFS文件、数据库、表及字段权限。这些策略可以为不同的用户和组来设置，同时其权限管理可与hadoop无缝对接。Apache Ranger具有以下目标：

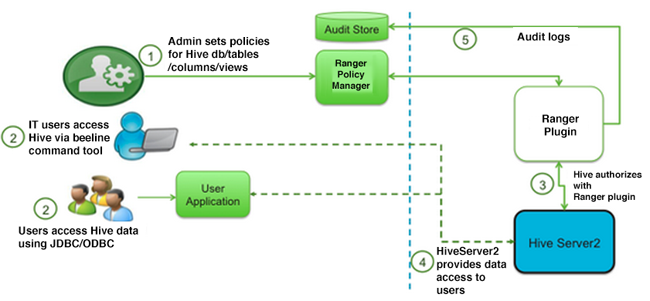
1. 集中安全管理，在中央UI中或使用REST API来管理所有与安全相关的任务
2. 使用Hadoop组件/工具执行特定行为（或操作）并通过中央管理工具进行细粒度的授权管理
3. 标准化所有Hadoop组件的授权方法
4. 增强对不同授权方法的支持，如基于角色的访问控制，基于属性的访问控制
5. 在Hadoop的所有组件中集中审核用户访问和与安全相关的管理操作

具有以下特性：

1. 基于策略（Policy-based）的访问权限模型
2. 通用的策略同步与决策逻辑，方便控制插件的扩展接入
3. 支持HDFS、YARN、HBase等控制插件，且可扩展
4. 内置基于LDAP、文件的用户同步机制，且可扩展
5. 统一的用户管理页面，包括策略管理、审计查看、插件管理等

# 2.Apache Ranger原理分析

Apache Ranger鉴权的本质是通过读取安全组件时生成的配置文件以及组件自带的jar包，通过hook的方式调用各个组件服务达到权限管理。在安装服务插件的过程中，主要指向三个步骤：将插件自带的conf更新到系统安装的服务conf下；将插件自带的lib更新到系统安装的服务lib下；将install.properties生成的xml文件，更新到系统安装的服务conf下。其Ranger权限管理流程图分析（以hive为例）：



如上图，鉴权过程分为5个步骤：

1. 集群管理者为Hive数据库设置安全策略
2. 集群访问者通过beeline或者JDBC访问Hive中的数据
3. 访问请求通过Ranger plugin进行安全验证，该插件具体的执行流程在下面会进行具体分析
4. 验证通过后，HiveServer2向用户访问请求数据
5. Range plugin根据用户和请求生成审计日志

服务的权限管理的执行逻辑是通用的，将Agent Plugin的鉴权逻辑嵌入到决策流程中是Apache Ranger的实现核心。

## 2.1权限模型

权限模型无非是定义了”用户-资源-权限”这三者的关系，Ranger基于Policy来抽象这种关系，进而扩展本身的权限模型。为了简化模型，使用下面的表达式来描述：

*Policy = Service + List<Resource> + AllowACL+ DenyACL*

*AllowACL = List<AccessItem> allow + List<AccessItem> allowException*

*DenyACL = List<AccessItem> deny + List<AccessItem> denyException*

*AccessItem = List<User/Group> +List<AccessType>*

从”用户-资源-权限”的角度进行详解:

1. 用户，由User或Group来表达，User代表访问资源的用户，Group代表用户所属的用户组
2. 资源，<Service,Resource>二元组来表达，一条Policy唯一对应一个Service，但可以对应多个Resource
3. 权限，由<AllowACL,DenyACL>二元组来表达，两者都包含两组AccessItem，AccessItem则描述一组用户与一组访问之间的关系，在AllowACL中表示允许执行，而DenyACL中表示拒绝执行

下面列出几种常见系统的模型实体枚举值：

|  |  |  |
| --- | --- | --- |
| Service | Resource | Access Type |
| HDFS | Path | Read/Write/Execute |
| YARN | Queue | Submit/Admin |
| HBase | Table,Column Family,Column | Read/Write/Create/Admin |
| Hive | Database,Table,Column | Select/Update/Created/Drop  /Alter/Index/Lock |

以AllowACL为例，假定要将资源授权给一个用户组G1，但是用户组里的某个用户U1除外，只需要增加一条包含G1的AccessItem到AllowACL\_allow，同时增加一条包含U1的AccessItem到AllowACL\_allowException即可。

一条Policy <= {alow,allowException,deny,denyException}四组AccessItem，作用的优先级由高到低依次：denyException > deny > allowException>allow，访问决策树流程图如下：



如果没有Policy能决策访问，Ranger可以选择将决策下放给系统自身的访问控制层，比如HDFS的ACL。

## 2.2Agent Plugin的嵌入

将Agent Plugin的鉴权逻辑嵌入到决策流程有两种方法：

1. **实现可扩展的接口**

多数的系统在实现时都有考虑功能扩展性的问题，一般会为核心模块暴露出可扩展的接口，访问控制模块也不例外。Ranger通过实现访问控制接口，将自己的逻辑嵌入到各个系统。下表列出了Ranger插件对几个常见系统的扩展接口：

|  |  |
| --- | --- |
| Service | Extensible Interface |
| YARN | org.apache.hadoop.yarn.security.YarnAuthorizionProvider |
| HBase | org.apache.hadoop.hbase.coprocessor.RegionObserver |
| Hive | org.apache.hadoop.hive.ql.security.authorization.plugin.HiveAuthorizer |
| Kafka | kafka.security.auth.Authorizer |
| Storm | backtype.storm.security.auth.IAuthorizer |
| Solr | org.apache.solr.security.AuthorizationPlugin |

1. **代码注入**

不排除少数系统没有将暴露访问控制扩展点，这种情况下，Ranger依赖Java代码注入机制（java.lang.instrument）来实现逻辑嵌入。

以HDFS插件为例，Ranger使用ClassFileTransformer，直接修改HDFS访问控制类的FSPermissionChecker的clsss File，将checkPermission方法替换成Ranger的自定义实现。其实现框架如下所示：

代码Inject



HDFS文件认证流程

# 3.Apache Ranger的安装

在第一部分介绍了Ranger的系统架构图，组件基本包括Admin Server，Agent Plugin,User Sync和Ranger Audit Store基本部分，下面介绍其安装和使用：

1. Ranger的编译（略）,编译后在target目录下生成需要的tarball
2. 安装mysql，修改密码/设置远程可连接
3. 安装Admin Server， Ranger的控制台

*tar -zxvf ranger-0.5.3-admin.tar.gz*

配置文件install.properties

*setup\_mode=SeparateDBA*

*DB\_FLAVOR=MYSQL*

*db\_root\_user=root*

*db\_root\_password=12345*

*db\_host=localhost*

*db\_name=ranger*

*db\_user=root*

*db\_password=12345*

*audit\_store=db*

*audit\_db\_name=ranger\_audit*

*audit\_db\_user=rangerlogger*

*audit\_db\_password=12345*

*policymgr\_external\_url=http://sgserver006:6080*

*policymgr\_http\_enabled=true*

在mysql中先创建database ranger和ranger\_audit，然后执行：

*./setup.sh*

启动ranger admin

*ranger-admin start*

验证是否安装成功，访问：http://<ranger-host>:6080，用admin/admin登陆，出现登陆页面则说明成功

1. 安装Ranger Usersync

同步linux系统用户到ranger数据库，需要安装Ranger Usersync，安装步骤如下：

*$tar –zxvf ranger-0.5.3-usersync.tar.gz*

*$cd ranger-0.5.3-usersync*

修改install.propeties，修改参数如下：

*POLICY\_MGR\_URL=http://<ranger-admin>:6080 （Ranger-admin 访问地址）*

*SYNC\_SOURCE=unix （同步 linux 下的用户）*

*SYNC\_INTERVAL=1 （同步时间间隔：分钟）*

*logdir=/var/log/ranger/usersync*

初始化设置Ranger Usersync，执行：

*./setup.sh*

修改ranger-usersync文件夹用户权限，ranger:hadoop ,在ranger用户下启动Ranger Usersync:

*./ranger-usersync-services.sh start*

验证是否安装成功，在Ranger控制台可以看到users中的同步的用户信息。

1. 安装hdfs plugin

将ranger的hdfs plugin插件安装在namenode所在的节点

*cd /cmss/bch/bc1.3.2/ranger/*

*tar –zxvf ranger-0.5.3-hdfs-plugin.tar.gz*

*mv ranger-0.5.3-hdfs-plugin ranger-hdfs*

配置install.properties：

*POLICY\_MGR\_URL=http://<ranger-admin>:6080 （Ranger-admin 访问地址）*

*REPOSITORY\_NAME=hadoopdev （服务名称）*

*XAAUDIT.DB.IS\_ENABLED=true*

*XAAUDIT.DB.FLAVOUR=MYSQL*

*XAAUDIT.DB.HOSTNAME= <MYSQL-HOST>*

*XAAUDIT.DB.DATABASE\_NAME=ranger\_audit*

*XAAUDIT.DB.USER\_NAME=root*

*XAAUDIT.DB.PASSWORD=123456*

将hadoop的配置文件conf放到ranger目录下，同时将HDFS Jar也指向ranger目录的lib目录，创建软链接

*ln -s /cmss/bch/bc1.3.2/hadoop/etc/hadoop/ /cmss/bch/bc1.3.2/ranger/hadoop/conf*

*ln -s /cmss/bch/bc1.3.2/hadoop/share/hadoop/hdfs/lib/ /cmss/bch/bc1.3.2/ranger/hadoop/lib*

执行启动hdfs的插件：

*./enable-hdfs-plugin.sh*

1. 配置hdfs-site.xml的相关参数

*dfs.permissions.enabled = true*

*dfs.permissions = true*

*dfs.namenode.inode.attributes.provider.class =*

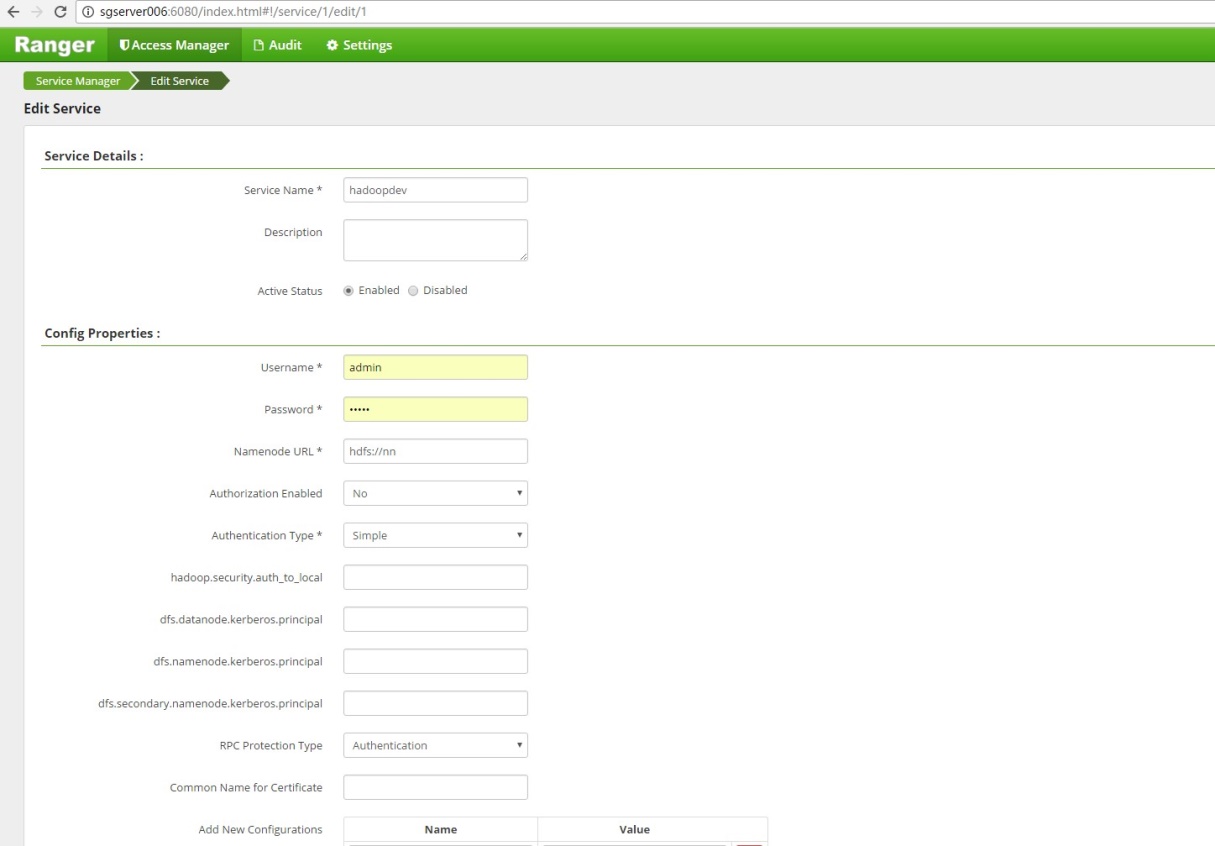
*org.apache.ranger.authorization.hadoop.RangerHdfsAuthorizer*

重启HDFS生效

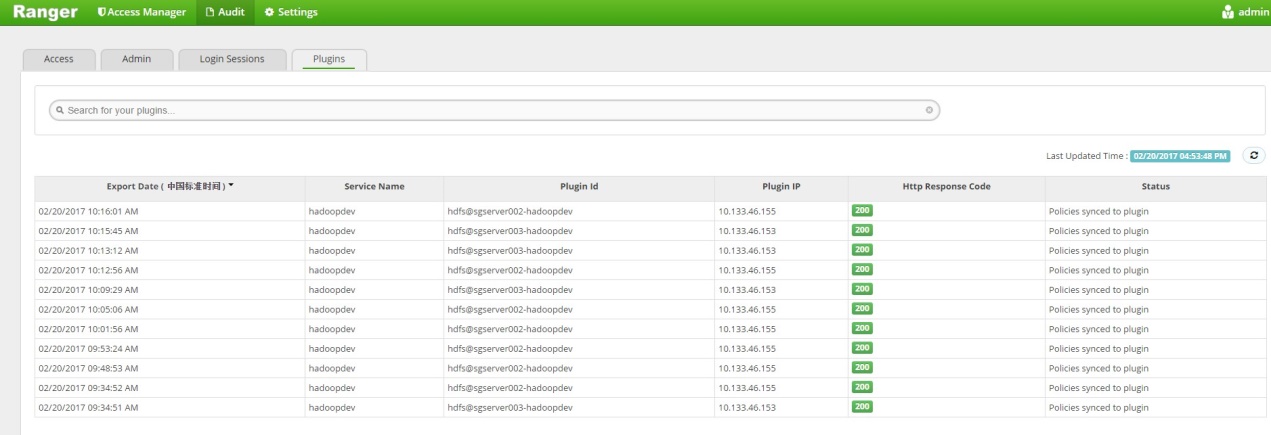
# 4.Apache Ranger的使用

1）登陆ranger admin

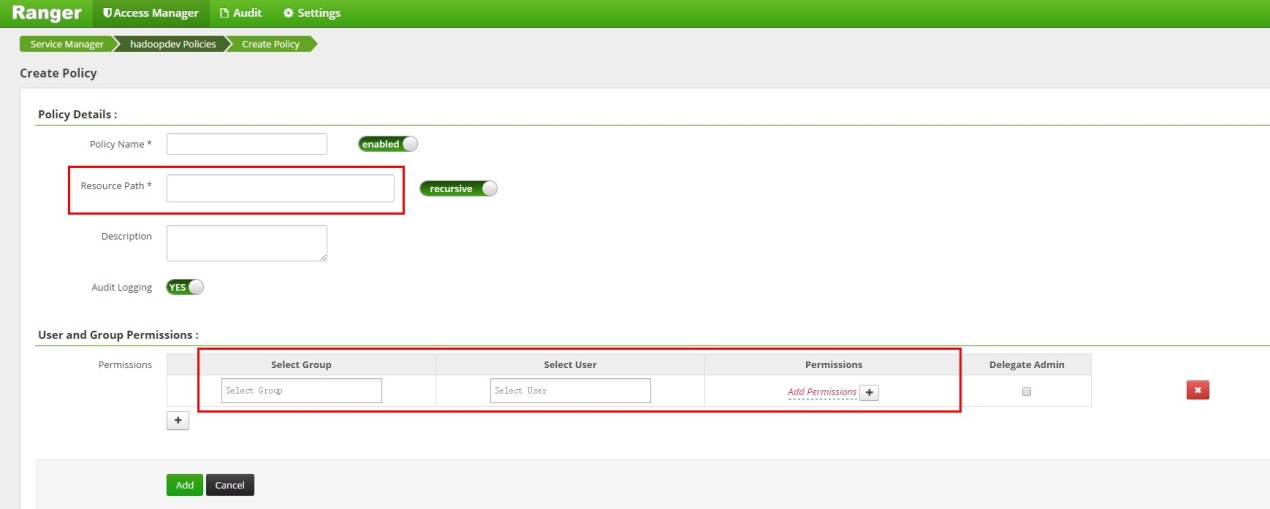
2）点击HDFS plugin添加按钮，添加服务，输入定义的服务名hadoopdev(在配置参数时设置的服务名称，要保持一致)



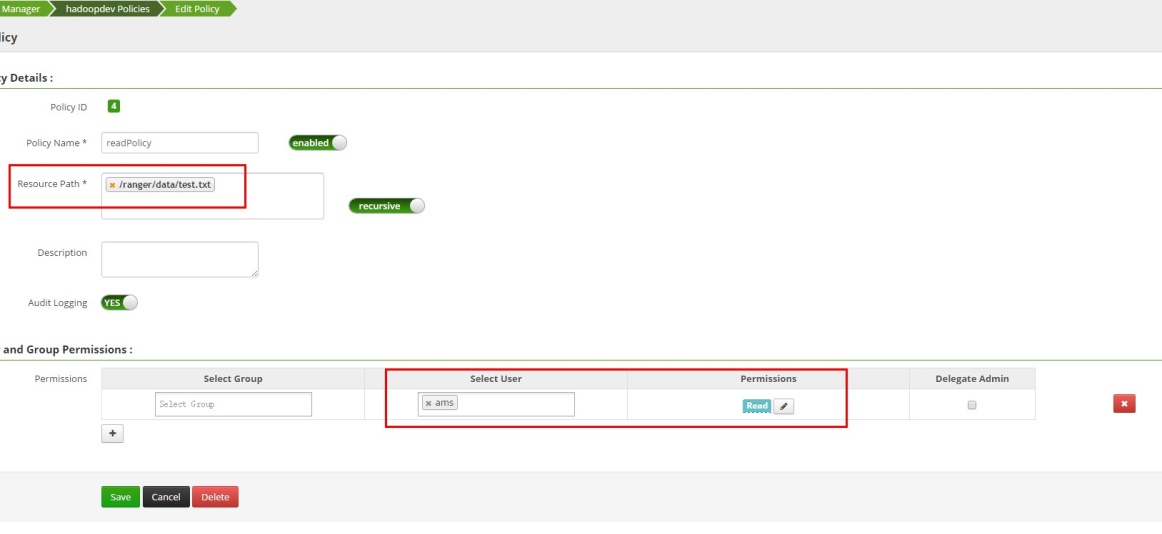
3）点击add，在audit –> plugins下回显示刚添加的组件（如下图所示），说明安装成功



1. 添加Policy,进入添加的Service hadoopdev，点击add New policy，添加



输入hdfs的路径资源，添加Group,User及Permissions等，如下面的例子中，为某个用户添加hdfs文件的读权限：



验证，未添加之前访问结果如下：

*$hdfs dfs -cat /ranger/data/test.txt*

*cat: Permission denied: user=ams, access=READ, inode="/ranger/data/test.txt":hdfs:hdfs:-rwx------*

*添加之后结果如下：*

*$hdfs dfs -cat /ranger/data/test.txt*

*Hello World*

*Bye World*

*Hello Hadoop*

*Byte Hadoop*

# 5.深入理解Ranger

## 5.1Audit Framework

Ranger中的Audit Framwork支持将audit logs持久化报错到RDBMS中，包括MySQL、Oracle等。Ranger的管理模块从关系型数据库中读取审计日志，并生成交互报告。

Audit logs的写入方式分为异步和同步：在同步模式下audit调用会将thread阻塞到完成数据库写入结束；在异步模式下，audit调用先将audit log写入到memory queue，不会影响系统的执行，向数据库中的写入可以用批处理的方式，当queue满后，可以周期性的生成droped audit log。

Apache Ranger中的audit执行序列图如下：

参考文献：

http://fx114.net/qa-72-150490.aspx