Storing Audit messages in Azure Blob Storage

Overview

Apache Ranger's audit framework can be configured to send access audit logs generated by Apache Ranger plug-ins to several types of destinations. It is now possible to use the HDFS audit destination to store audit data in Azure Blob Storage by using Hadoop's Azure Support. This document provides details of configuring Ranger's HDFS audit provider to store audit to Azure Blob Storage.

Introduction

Since Ranger uses the *Hadoop's Azure Support* to write to *Azure Blob Storage,* configuring HDFS audit provider to store audit messages in *Azure Blob Storage* is the same as setting up a HDFS audit provider - with a just few additional Azure specific properties.

Configuring the Azure specific properties for Auditing

Manual install

If you are manually installing a plugin by editing the install.properties file then you need to define the following additional properties specific to Azure.

Storage location

The specific container where the audits would be stored. This is similar to the destination directory for regular HDFS audit provider.

Item	Description
Property Name in install.properties	`XAAUDIT.HDFS.DESTINATION_DIRECTORY`
Example value	`wasb://ranger_audit_container@my_azure_account.blob.core.windows.net/ranger/audit`
Notes	If you have an HDFS cluster backed by Azure Blob Storage then this should be the value of following property fr om the core-site.xml file. `fs.defaultFS`

Account Information

Azure account information is composed of several parts and their details can be found here. Following is how they you would configure it for ranger audit.

Property Name in Install.properties	Example Value	Notes
`XAAUDIT.HDFS.AZURE_ACCOUNTNAME`	`my_azure_account`	This is
`XAAUDIT.HDFS.AZURE_ACCOUNTKEY`	<your encrypted access key>	This is `fs.az`
`XAAUDIT.HDFS.AZURE_SHELL_KEY_PROVIDER`	`org.apache.hadoop.fs.azure.ShellDecryptionKeyProvider`	This is `fs.az`

`XAAUDIT.HDFS.AZURE_ACCOUNTKEY_PROVIDER`	`/usr/lib/python2.7/dist-packages/hdinsight_common/decrypt.sh`	This is `fs.az`

Ambari Install

For an Ambari install please add the following properties to the *Advanced ranger-<service-type>-audit* configuration of the service.

Item	Description
Property Name	`xasecure.audit.destination.hdfs.dir`
Example Value	`wasb://ranger_audit_container@my_azure_account.blob.core.windows.net/ranger/audit`
Notes	• If you have an HDFS cluster backed by Azure Blob Storage then this should be the value of following property from the core-site.xml file. `fs.defaultFS`

In addition specify the following in the *Custom ranger-<service-type>-audit* configuration of the service.

#	Item	Description
1	Property Name	`xasecure.audit.destination.hdfs.config.fs.azure.account.key.<account-name>.blob.core.windows.ne`
	Example Value	<your encrypted access key>
	Notes	• Replace the placeholder <account-name> in the property name with your Azure Blob storage account-name. • This is the following property in core-site.xml. `fs.azure.account.key.<account-name>.blob.core.windows.net`
2	Property Name	`xasecure.audit.destination.hdfs.config.fs.azure.shellkeyprovider.script`
	Example Value	`org.apache.hadoop.fs.azure.ShellDecryptionKeyProvider`
	Notes	• This is following property in core-site.xml. `fs.azure.account.keyprovider.<account-name>.blob.core.windows.net`
3	Property Name	`xasecure.audit.destination.hdfs.config.fs.azure.account.keyprovider.<account-name>.blob.core.wi`
	Example Value	`/usr/lib/python2.7/dist-packages/hdinsight_common/decrypt.sh`
	Notes	• Replace the placeholder <account-name> in the property name with your Azure Blob storage account-name. • This is following property in core-site.xml. `fs.azure.shellkeyprovider.script`