
HDP Operations:
Security
Rev 1.1.0
HORTONWORKS
UNIVERSITY

Approximate Daily Schedule

Day 1 - Lessons	Day 2 - Lessons	Day 3 - Lessons
• Definition of Security • Securing Sensitive Data • Integrating HDP security • What Security Tool for Use Case • HDP Security Prerequisites • Ambari Server Security • Kerberos Deep Dive	• Enable Kerberos • Apache Ranger Installation • Apache Ranger KMS	• Secure Access With Ranger • Apache Knox Overview • Apache Knox Installation • Ambari Views for Controlled Access

Introductions

- Your name
- Your job role and responsibilities
- Your Linux administration experience, if any
- Your Hadoop experience, if any
- Your expectations for the course

Class Logistics

- Days and hours
- Breaks and lunch
- Facility information (if applicable)
  - Exits, restrooms, break room…
- Courseware
  - Do you have it?
  - Where do you get it?
- Technical information
  - Wireless access, cloud access, virtual machine information…


Definition of Security
HORTONWORKS
UNIVERSITY

Objectives

After completing this lesson, students should be able to:

- Define security
- Explain the five pillars of security
- List the Apache projects that address each pillar

We begin by discussing the five pillars of cluster security.

Definition of Security

- Oxford Dictionaries:
  - The state of being free from danger or threat
- Cambridge Dictionaries:
  - Freedom from risk and the threat of change for the worse
- Merriam-Webster Dictionaries:
  - The state of being protected or safe from harm

Definition of Computer Security

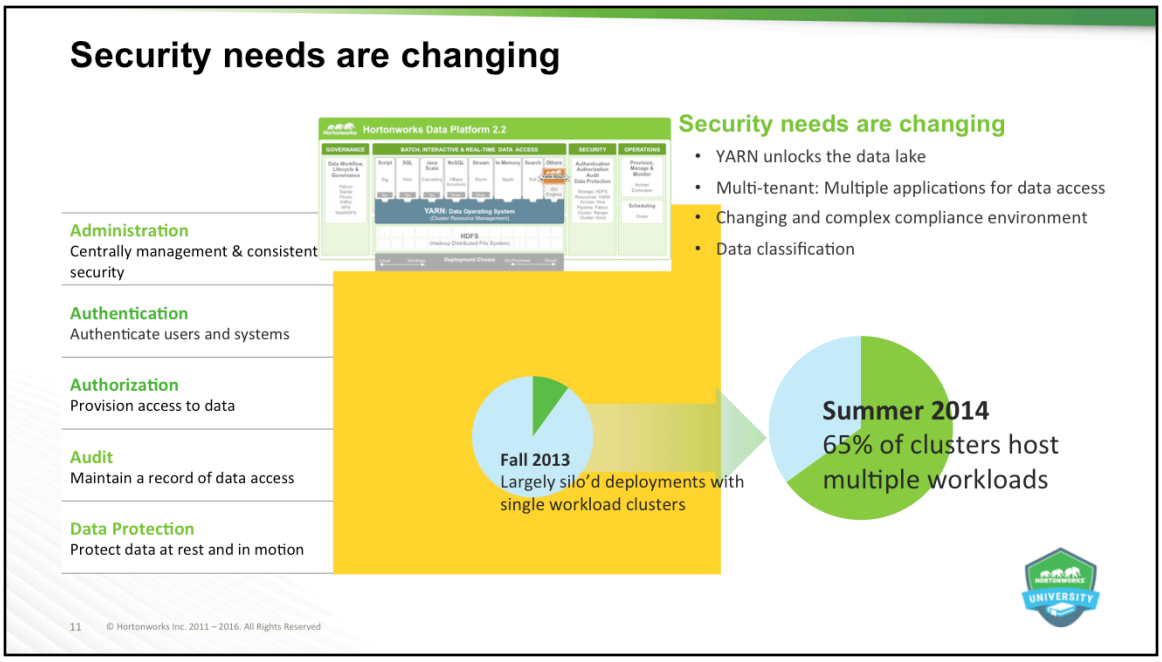
- Wikipedia:
  - Is the protection of information systems from theft or damage to the hardware, the software and to the information on them, as well as from disruption or misdirection of the services they provide.
  - Covers all the processes and mechanisms by which digital equipment, information and services are protected from unintended or unauthorized access, change or destruction and the process of applying security measures to ensure confidentiality, integrity and availability of data both in transit and at rest.

Applies to protect against combination of:

-h/w and s/w….as well as services(apis)

-Malicious and unintentional access/modification

-data in transit and at rest



Story of hadoop security is closely tied to story of hadoop

With the evolution of Hadoop 2 with invention of YARN, customer deployments have moved from silo'd single node clusters to larger clusters hosting multiple workloads

With this Multi-tenancy and increased compliance requirements and consolidation of clusters => bigger 'bank' for the 'data robbers'

To protect against these there is a need for comprehensive security, as your cluster is only as secure as the weakest link

This is done via multiple layers or pillars: authentication, authorization, audit, data protection and central administration

Overview

- The creation of a Hadoop-powered Data Lake can provide a robust foundation for a new generation of analytics and insight.
- It's important to consider security before launching or expanding a Hadoop initiative
- By making sure that data protection are built into the environment, one can leverage the full value of advanced analytics without exposing your business to new risks.
- To ensure effective protection for our customers, we use a holistic approach based on Five Pillars of Security

What we see is that as companies rush to put Big Data to work for their business, these new ways of operating can sometimes get ahead of IT's ability to digest their full implications. There's no question that the creation of a Hadoop-powered Data Lake can provide a robust foundation for a new generation of analytics and insight, but it's important to consider security before launching or expanding a Hadoop initiative. By making sure that data protection and governance are built into your Big Data environment, you can leverage the full value of advanced analytics without exposing your business to new risks.

Hortonworks understands the importance of security and governance for every business. To ensure effective protection for our customers, we use a holistic approach based on five pillars:

- Administration
- Authentication and Perimeter Security
- Authorization
- Audit
- Data protection


Objectives
The Definition of Security
The Five Pillars of Security
13
© Hortonworks Inc. 2011 – 2016. All Rights Reserved

Five Pillars of Security

Administration
Central management & consistent security

Authentication
Authenticate users and systems

Authorization
Provision access to data

Audit
Maintain a record of data access

Data Protection
Protect data at rest and in motion

In each of these areas, Hortonworks provides differentiated capabilities beyond those of other vendors to help customers achieve the highest possible level of protection. As a result, Big Data doesn't have to incur big risks—and companies can put it to work without sacrificing peace of mind.

The Security Challenge

In order to protect any data system you must implement the following

Administration
Centrally management & consistent security

Authentication
Authenticate users and systems

Authorization
Provision access to data

Audit
Maintain a record of data access

Data Protection
Protect data at rest and in motion

3 Reasons for Data Protection

- **Malicious intent**
- **Unintentional breach**
- **Compliance**

Why are we protecting ourselves?

In case of Malicious intent or

In case of Unintentional breach or

For Compliance

Holistic Approach

- Effective Hadoop Security Depends on Holistic Approach
- Piecemeal protections are no more effective for a Data Lake than they would be in a traditional Data repository
- No point in securing the primary access path to the data lake when a user can simply access the same data through a different path
- Framework for comprehensive security revolves around five pillars
  - administration
  - authentication/perimeter security
  - authorization
  - audit
  - data protection

Piecemeal protections are no more effective for a Data Lake than they would be in a traditional repository. There's no point in securing the primary access path to the data lake when a user can simply access the same data through a different path.

Hortonworks firmly believes that effective Hadoop security depends on a holistic approach. Our framework for comprehensive security revolves around five pillars: administration, authentication/ perimeter security, authorization, audit and data protection.

Five Pillars of Security

Administration Central management & consistent security	How do I set policies across the cluster?
Authentication Authenticate users and systems	Who am I? Prove it!
Authorization Provision access to data	What can I do?
Audit Maintain a record of data access	What did I do?
Data Protection Protect data at rest and in motion	Can I encrypt data at rest and over the wire?

Lets talk a little about what these terms mean

Security Administration

- Hadoop Security Strategy Must Address All Five Pillars
- Must have a consistent implementation approach to ensure their effectiveness
- Must address questions and provide enterprise-grade coverage
- Any weak pillar introduces threat vectors to the entire Hadoop-powered data lake
- Centralized Security Framework to Manage Fine Grained Access Control

Security administrators must address questions and provide enterprise-grade coverage across each of these pillars as they design the infrastructure to secure data in Hadoop. If any of these pillars remains weak, it introduces threat vectors to the entire data lake. In this light, your Hadoop security strategy must address all five pillars, with a consistent implementation approach to ensure their effectiveness.

Administration

- Identify Known/Unknown Vulnerabilities
- Design Measures and Strategies to Mitigate Vulnerabilities
- Implement and Configure Security Software
- Create, Implement, Manage and Monitor Security Policy

In order to deliver consistent security administration and management, Hadoop administrators require a centralized user interface—a single pane of glass that can be used to define, administer and manage security policies consistently across all the components of the Hadoop stack. Hortonworks addressed this requirement through Apache Ranger, an integral part of HDP, which provides a central point of administration for the other four functional pillars of Hadoop security.

Authentication

- Process of verifying the identity of a particular user or service
- Usually based on username and password
- Ensuring user/service is really the one they are claiming to be
- Says nothing about the access rights of the user
- Precedes authorization

In order to deliver consistent security administration and management, Hadoop administrators require a centralized user interface—a single pane of glass that can be used to define, administer and manage security policies consistently across all the components of the Hadoop stack. Hortonworks addressed this requirement through Apache Ranger, an integral part of HDP, which provides a central point of administration for the other four functional pillars of Hadoop security.

Authorization

- Granting or denying access to specific resources based on the requesting user's identity or authentication
- Performed through access control lists
- Supports authorization by limiting access to
  - Service API's
  - File Permissions
  - Job Execution
  - Administrative Permissions
  - Job Information Accessible to Authorized Users

Usually works by limiting access to…

Audit

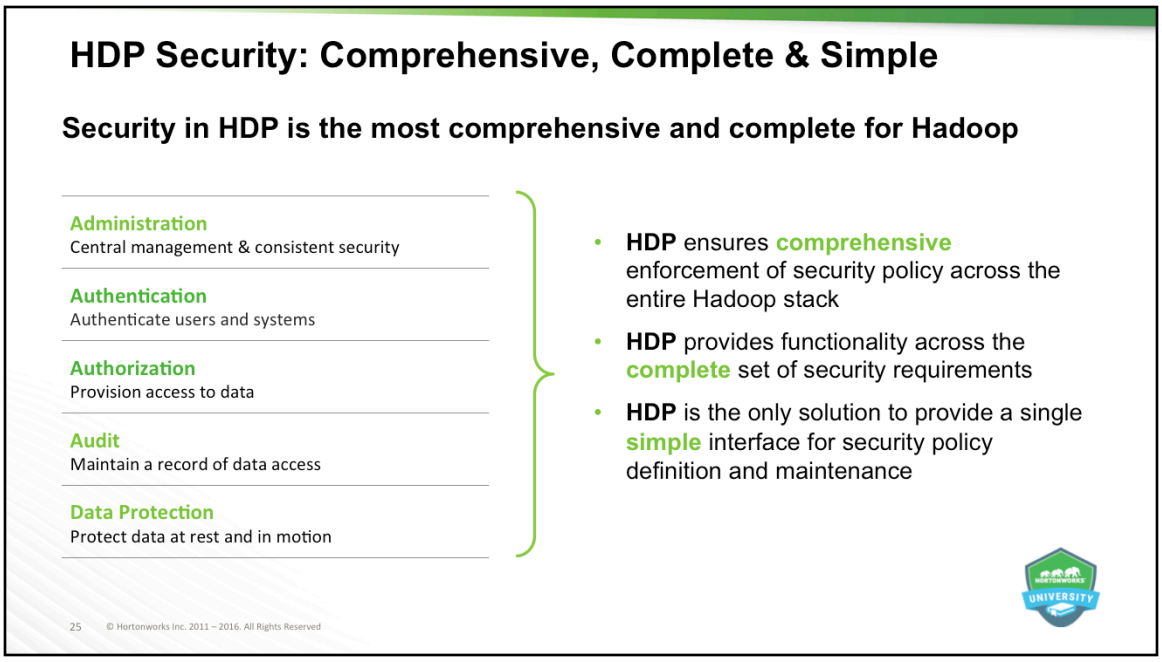
- Final Step in Security Implementation
- Process of Testing and Ensuring Policy is Working
- Review Audit Logs and Reports

Data Protection

- Process of Safeguarding Important Information from Corruption/Loss
- Ensure Information is not Tampered/Sniffed when
  - Transferred Over Unsecure Network
  - Stored On Unsecure Disks

Applied to data at rest (someone walkign away with disk) and in motion (someone snooping)

We've discussed the five pillars of cluster security and why they are important in an HDP environment. Now let's introduce some of the tools and platforms available for HDP Security Administrators to help make their HDP clusters as secure as possible.



HDP provides the most comprehensive, complete and simple set of security functionally for Hadoop.

HDP enables COMPREHENSIVE enforcement of a single policy across the entire stack

Hortonworkstake a holistic view on the enterprise security requirements and ensure that Hadoop can not only define but also apply a complete policy.

Our contributions to securing Hadoop start with integration of Kerberos and include HDFS encryption and a myriad of other contributions we have made to the community in nearly every project of the Hadoop stack and we have incubated Apache Knox to address perimeter security.

And consistent with our approach, EVERYTHING we do is in the open and via the open Apache community. To this end, we acquired a company last year and took the once proprietary code and donated to the community as an ASF governed project

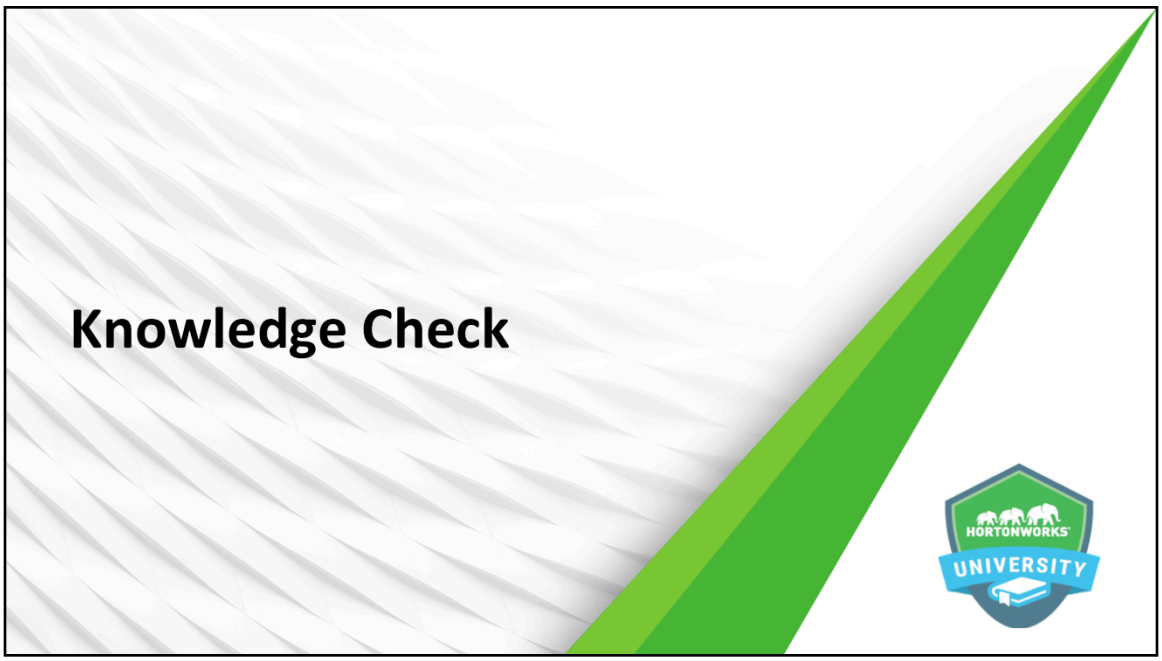
HDP Security: Comprehensive, Complete & Simple

In order to protect any data system you must implement the following

Administration Central management & consistent security	Only HDP delivers a single administrative console to set policy across the entire cluster	**Apache Ranger**
Authentication Authenticate users and systems	Integrate with existing AD and LDAP authentication for perimeter and project access	**Apache Knox, Native Kerberos**
Authorization Provision access to data	Work within all Apache projects to provide consistent authorization controls	**Apache Ranger**
Audit Maintain a record of data access	Maintain a record of events across all components that is consistent and accessible	**Apache Ranger**
Data Protection Protect data at rest and in motion	Wire and storage encryption in Hadoop. Refer partner encryption solutions for more advanced needs	**HDFS, Hadoop, Ranger KMS**

Five Pillars of Security

Administration Central management & consistent security	Apache Ranger
Authentication Authenticate users and systems	Apache Knox Kerberos
Authorization Provision access to data	Apache Ranger
Audit Maintain a record of data access	Apache Ranger
Data Protection Protect data at rest and in motion	HDFS (Data Encryption) Apache Ranger (KMS) Partners


Knowledge Check
HORTONWORKS
UNIVERSITY

Questions

1. What is the definition of security?
2. What is computer security?
3. What are the five pillars of security?
4. Which Apache projects address security?

1. The state of being free from danger or threat; Freedom from risk and the threat of change for the worse; The state of being protected or safe from harm

2. Is the protection of information systems from theft or damage to the hardware, the software and to the information on them, as well as from disruption or misdirection of the services they provide. Covers all the processes and mechanisms by which digital equipment, information and services are protected from unintended or unauthorized access, change or destruction and the process of applying security measures to ensure confidentiality, integrity and availability of data both in transit and at rest

3. Administration, Authentication, Authorization, Audit and Data Protection

4. Apache Knox, Apache Ranger & Ranger KMS, HDFS Data Encryption


Summary
HORTONWORKS
UNIVERSITY

Summary

- Security is the state of being free from danger or threat
- Computer security is the protection of information systems from theft or damage to the hardware, the software and to the information on them, as well as from disruption or misdirection of the services they provide
- The five pillars of security are administration, authentication, authorization, audit and data protection
- Apache Knox, Ranger, Ranger KMS and HDFS Data Encryption address security concerns in Hadoop with a holistic approach

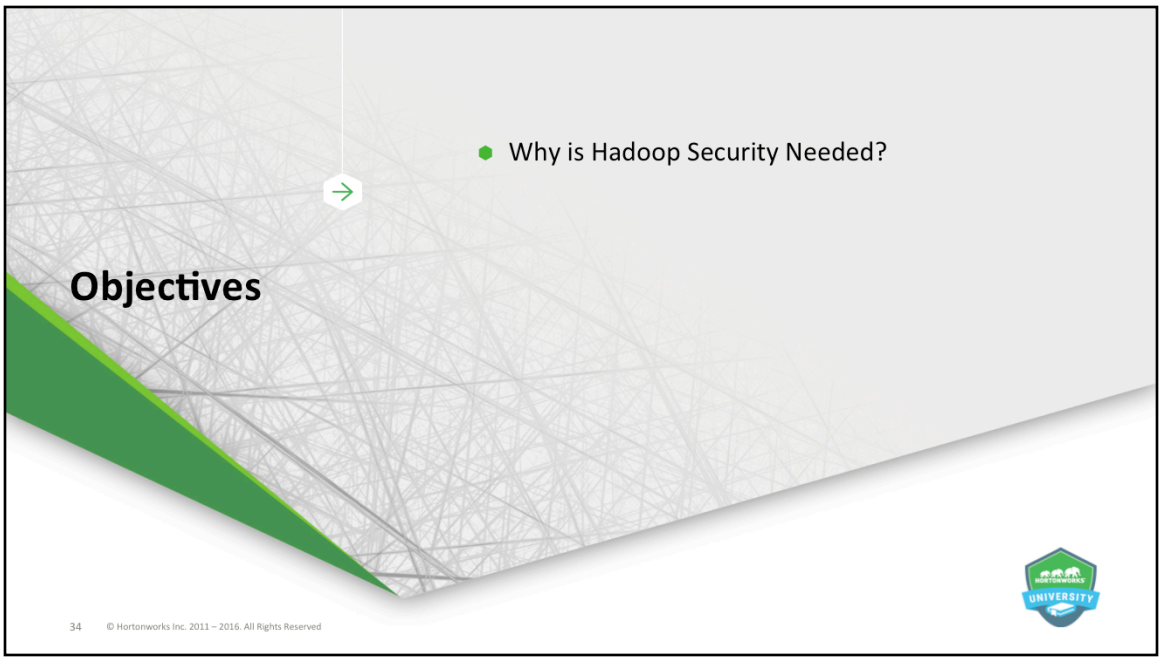
This summary page lists some of the main points from this lesson.


Securing Sensitive Data
HORTONWORKS
UNIVERSITY

Objectives

After completing this lesson, students should be able to:

- Explain why security is needed in Hadoop
- List examples of sensitive data
- Explain the need to secure sensitive data
- List the Apache projects that secure sensitive data


Why is Hadoop Security Needed?
Objectives
34
© Hortonworks Inc. 2011 – 2016. All Rights Reserved

History of Hadoop Security

- Insufficient Authentication/Authorization of Both User/Services
- Framework Did Not Perform Mutual Authentication
- Malicious User Can Impersonate Services
- Minimal Authorization Allows Anyone to Read/Write Data
- Arbitrary Java Code Can Be Executed By User/Service Account
- File Permissions Easily Circumvented
- Only Disk Encryption

Security Implications

- Data is essential new driver of competitive advantage
- Hadoop plays critical role in modern data architecture by
  - Providing low-cost
  - Scale-out data storage
  - Value-add processing
- Any internal or external breach of this enterprise-wide data can be catastrophic
  - Privacy violations
  - Regulatory infractions
  - Damage to corporate image
  - Damage to long-term shareholder value

The consensus is strong among leading companies in every industry: data is an essential new driver of competitive advantage. Hadoop plays a critical role in the modern data architecture by providing low-cost, scale-out data storage and value-add processing. The successful Hadoop journey typically starts with Data Architecture Optimization or new Advanced Analytic Applications, which leads to the formation of a Data Lake. As existing and new types of data from sensors and machines, server logs, clickstreams, and other sources flow into the Data Lake, it serves as a central repository based on shared Hadoop services that power deep organizational insights across a large, broad and diverse set of data.

The need to protect the Data Lake with comprehensive security is clear. As large and growing volumes of diverse data are stored in the Data Lake, it comes to hold the crown jewels of your company—the vital and often highly sensitive data that has shaped and driven your business over a long history. However, the external ecosystem of data and operational systems feeding the Data Lake is highly dynamic and can introduce new security threats on a regular basis. Users across multiple business units can access the Data Lake freely and refine, explore and enrich its data at will, using methods of their own choosing, thereby increasing risks of exposure to

Security Implications

```
$ whoami
baduser
$ hadoop fs -ls /tmp
Found 2 items
drwx-wx-wx   - ambari-qa hdfs   0 2015-07-14 18:38 /tmp/hive
drwx------   - hdfs      hdfs   0 2015-07-14 20:33 /tmp/secure
$ hadoop fs -ls /tmp/secure
ls: Permission denied: user=baduser, access=READ_EXECUTE, inode="/tmp/secure":hdfs:hdfs:drwx------
```

Good right?

Security Implications

```
$ whoami
baduser
$ hadoop fs -ls /tmp
Found 2 items
drwx-wx-wx   - ambari-qa hdfs 0 2015-07-14 18:38 /tmp/hive
drwx------   - hdfs      hdfs 0 2015-07-14 20:33 /tmp/secure
$ hadoop fs -ls /tmp/secure
ls: Permission denied: user=baduser, access=READ_EXECUTE, inode="/tmp/secure":hdfs:hdfs:drwx------
```

Good right? – Look Again!!!

```
$ HADOOP_USER_NAME=hdfs hadoop fs -ls /tmp/secure
Found 1 items
drwxr-xr-x   - hdfs hdfs      0 2015-07-14 20:35 /tmp/secure/blah
```

What if that user did the following:

```
$ HADOOP_USER_NAME=hdfs hadoop fs –rm –R -skiprash /
```

In unsecured Hadoop clusters, there is a way to circumvent security where user can impersonate super user by setting env var

This shows you that w/o kerbero there really isn't much security on Hadoop clusters


Why is Hadoop Security Needed?
What are You Securing?
Objectives
39
© Hortonworks Inc. 2011 – 2016. All Rights Reserved

Sensitive Data

- Encompasses Wide Range of Information
  - Ethnic or Racial Origin
  - Political Opinion
  - Religious or Similar Beliefs
  - Memberships
  - Physical/Mental Health Details
  - Personal Life
  - Criminal/Civil Offences
- Protected by Civil Rights

Sensitive Data

- Information that Relates One As
  - Consumer
  - Client
  - Employee
  - Patient
  - Student
- Personnel Identifiable Information
  - Contact Information
  - Identification Cards/Numbers
  - Birth Date
  - Parent Names
- Right to Access and Know How Other Access


Objectives
Why is Hadoop Security Needed?
What are You Securing?
Why are You Securing?
42
© Hortonworks Inc. 2011 – 2016. All Rights Reserved

Compliance Adherences

- HIPAA - Health Insurance Portability and Accountability Act of 1996
- HITECH - The Health Information Technology for Economic and Clinical Health Act
- PCI DSS - Payment Card Industry Data Security Standard
- SOX - The Sarbanes-Oxley Act of 2003
- ISO - International Organization Standardization
- COBIT - Control Objectives for Information and Related Technology

- Corporate Security Policies

Critical to Business

- Success is Based on Confidence and Trust of Customers
- Success Involves Protecting the Customers
- Protecting Losses Increases Business Revenue
- Avoiding:
  - Security Breaches
  - Civil and Criminal Lawsuits
  - Additional Government Regulation


Objectives
Why is Hadoop Security Needed?
What are You Securing?
Why are You Securing?
How are You Securing?
45
© Hortonworks Inc. 2011 – 2016. All Rights Reserved

HDP's - Five Pillars of Security

Administration Central management & consistent security	Apache Ranger
Authentication Authenticate users and systems	Apache Knox Kerberos
Authorization Provision access to data	Apache Ranger
Audit Maintain a record of data access	Apache Ranger
Data Protection Protect data at rest and in motion	HDFS (Data Encryption) Apache Ranger (KMS) Partners

How to Secure

- Kerberos
  - Originally developed for MIT's Project Athena in the 1980s
  - The most widely deployed system for *authentication*
  - Shipped with all major computer operating systems
  - MIT developed/maintains implementations for
    - Linux/Unix
    - Apple Macintosh
    - Windows

Kerberos was originally developed for MIT's Project Athena in the 1980s and has grown to become the most widely deployed system for authentication and authorization in modern computer networks. Kerberos is currently shipped with all major computer operating systems and is uniquely positioned to become a universal solution to the distributed authentication and authorization problem of permitting universal "single sign-on" within and between federated enterprises and peer-to-peer communities. MIT has developed and maintains implementations of Kerberos software for the Apple Macintosh, Windows and Unix operating systems.

KEEP THIS AT HIGH LEVEL – DETAILS IN FOLLOWING MODULES

How to Secure

- Apache Ranger
  - Centralized Security Framework to Manage Fine Grained Access Control
  - *Administration* Console Can Easily Manage
    - Policies for Accessing a Resource (File, Directories, Database, Table Column) for Users/Groups
    - Enforce *Authorization* Policies within Hadoop
    - Enable *Audit* Tracking and Policy Analytics
- Apache Ranger KMS
  - Scalable Cryptographic Key Management Service Supporting HDFS "data at rest" *Encryption*
  - Based on Hadoop KMS Originally Developed by the Apache Community
  - Hadoop KMS Stores Keys by Default in File-Based Java Keystore
  - Extends Native Hadoop KMS Functionality By Allowing to Store Keys in Secure Database

Ranger Key Management Service (Ranger KMS) is a open source, scalable cryptographic key management service supporting HDFS "data at rest" encryption*.

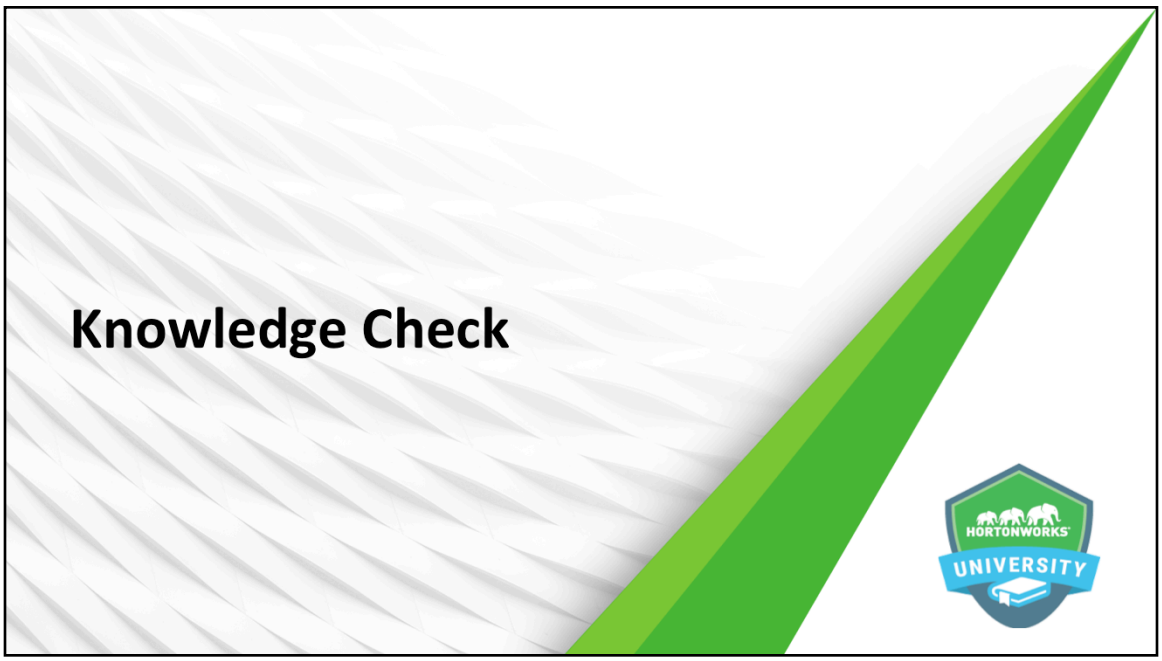
Ranger KMS is based on the Hadoop KMS originally developed by the Apache community. The Hadoop KMS stores keys in a file-based Java keystore by default. Ranger extends the native Hadoop KMS functionality by allowing you to store keys in a secure database.

KEEP THIS AT HIGH LEVEL – DETAILS IN FOLLOWING MODULES

How to Secure

- HDFS Data Encryption
  - Data At Rest Encryption Implements End-to-End *Encryption* of Data Read From/Written To HDFS
  - End-to-End *Encryption* – Means Data is Encrypted/Decrypted by Client
  - HDFS Does Not Have Access to Unencrypted Data/Keys
- Apache Knox
  - Gateway System that Provides Single Point of *Authentication* and Access
  - Provide Perimeter Security for Hadoop REST API's
  - Exposes a Single URL Hierarchy that Aggregates REST API's of a Hadoop Cluster

KEEP THIS AT HIGH LEVEL – DETAILS IN FOLLOWING MODULES


Knowledge Check
HORTONWORKS
UNIVERSITY

Questions

1. What is the new driver of competitive advantage?
2. What role does Hadoop play in modern data architecture?
3. What are some types of sensitive data?
4. Name compliance adherences that exists today?
5. What HDP component is used for authentication, authorization and audit?

1. Data is essential new drive of competitive advantage.

2. Provides low cost, scale out data storage and value-add processing

3. Ethnic or Racial Origin, Political Opinion, Religious or Similar Beliefs, Memberships, Physical/Mental Health Details, Personal Life, and Criminal/Civil Offences, Consumer, Client, Employee, Patient, Student, Contact, ID Cards/Numbers, Birth Dates and Parent Names.

4. HIPPA, HITECH, PCI DSS, SOX, ISO, COBIT, Corporate Security Policies

5. Apache Knox & Kerberos, Apache Ranger, Apache Ranger


Summary
HORTONWORKS
UNIVERSITY

Summary

- Security in Hadoop is needed to address historical issues in Hadoop
- Data is the new driver of competitive advantage
- Any data breach can be catastrophic
- There are many types of data that are considered sensitive
- Adherence to compliance regulations and standards are critical to the business

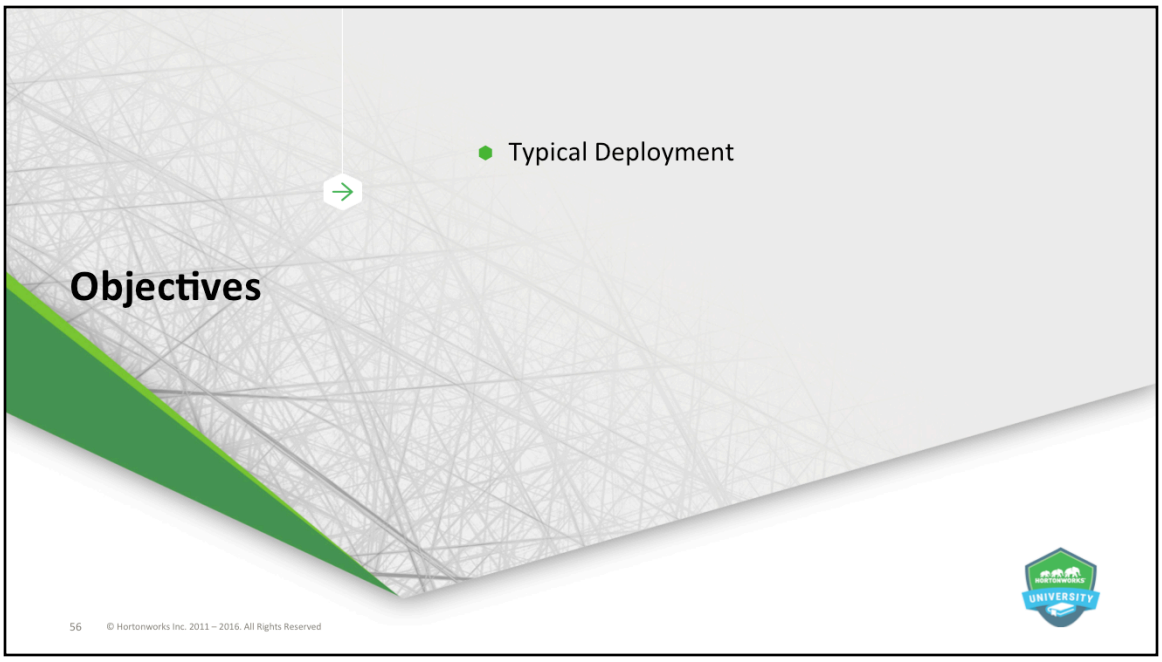
This summary page lists some of the main points from this lesson.


Integrating HDP Security
HORTONWORKS
UNIVERSITY

Objectives

After completing this lesson, students should be able to:

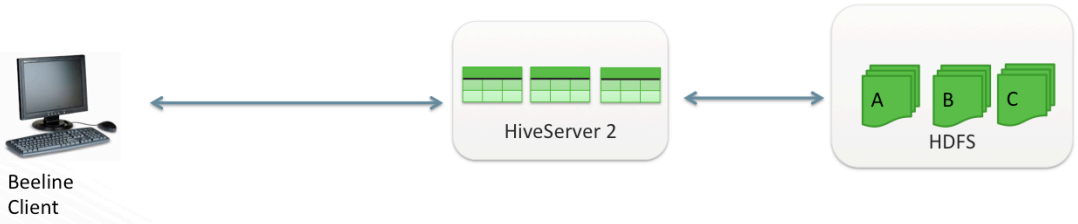
- Describe a typical multi-step security deployment approach
- List and describe the component used to integration security by pillar
  - Administration – Apache Ranger
  - Authentication – Apache Knox and Kerberos
  - Authorization – Apache Ranger
  - Audit – Apache Ranger
  - Data Protection – HDFS Data Encryption, Apache Ranger KMS


Typical Deployment
Objectives
56
© Hortonworks Inc. 2011 – 2016. All Rights Reserved

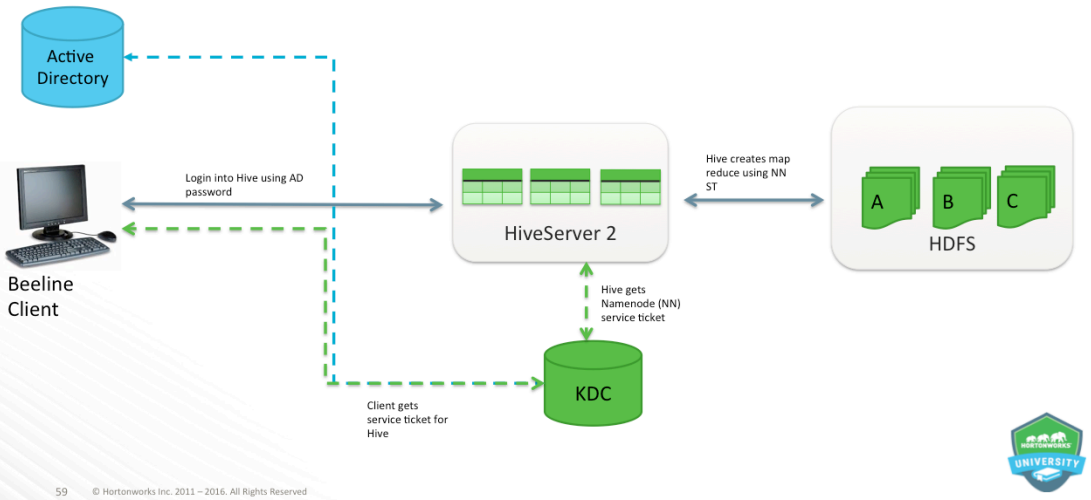
HDP's - Five Pillars of Security

Administration Central management & consistent security	Apache Ranger
Authentication Authenticate users and systems	Apache Knox Kerberos
Authorization Provision access to data	Apache Ranger
Audit Maintain a record of data access	Apache Ranger
Data Protection Protect data at rest and in motion	HDFS (Data Encryption) Apache Ranger (KMS) Partners

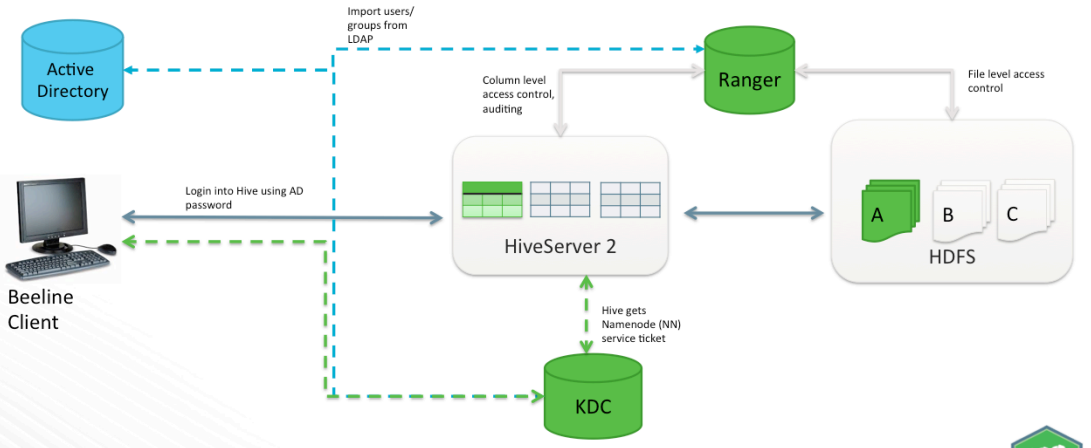
Typical Flow – SQL Access through Beeline client

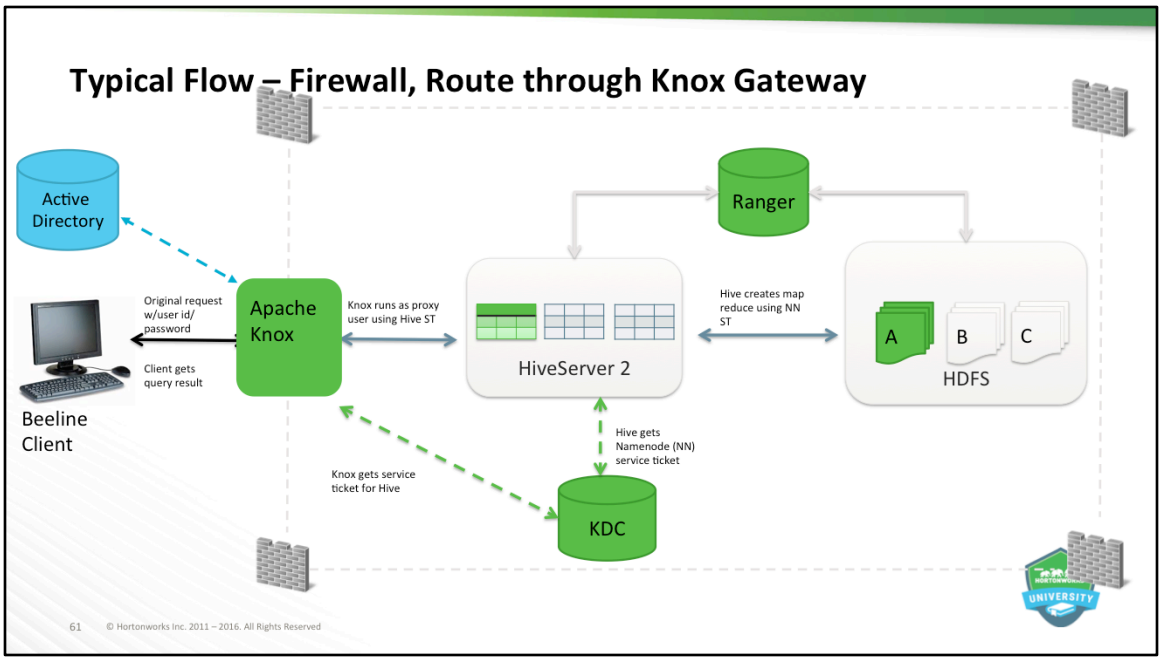


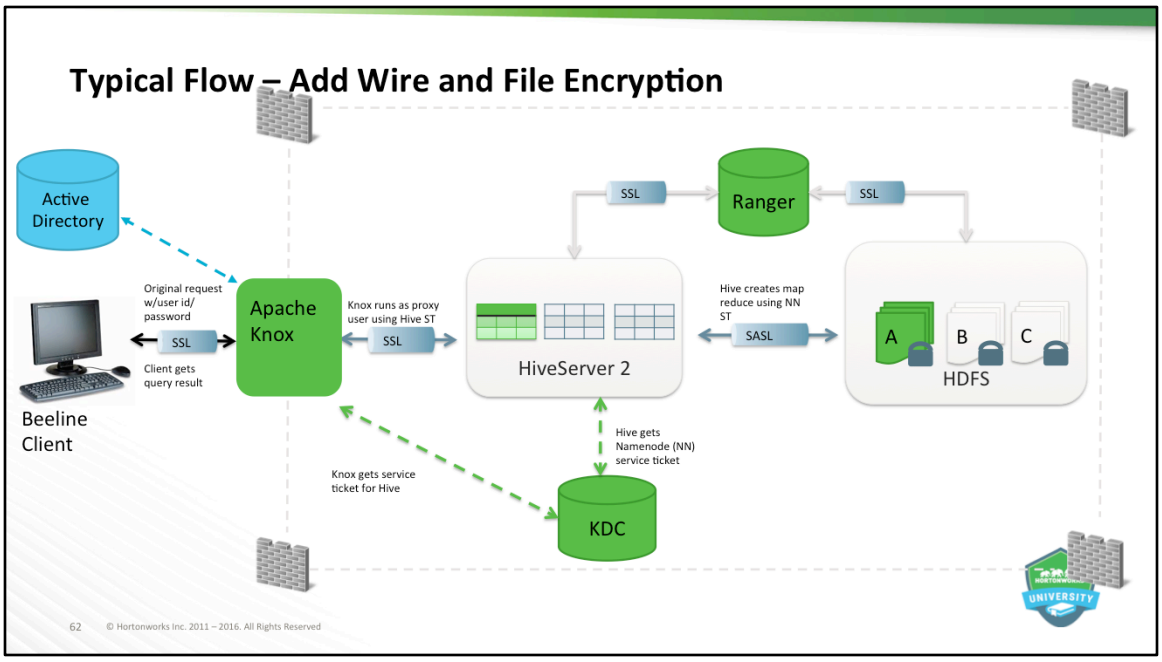
Typical Flow – Authenticate through Kerberos

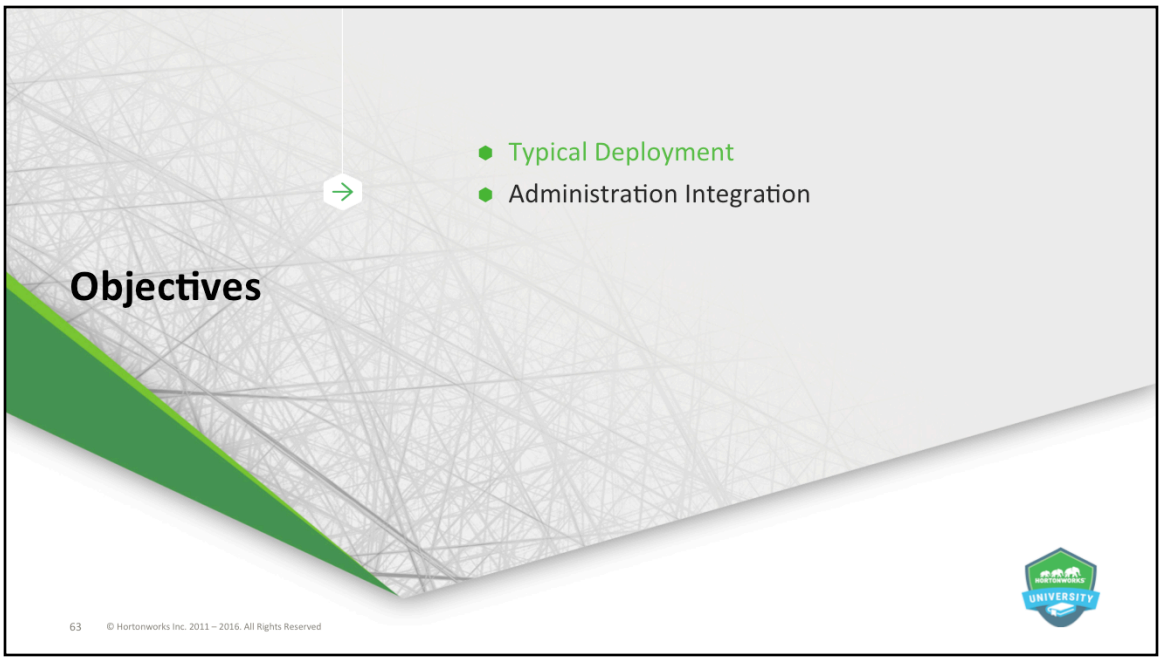


Typical Flow – Add Authorization through Ranger



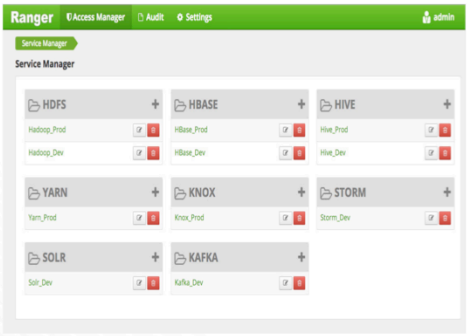

Typical Flow – Firewall, Route through Knox Gateway
Active Directory
Ranger
Original request w/user id/ password
Apache Knox
Knox runs as proxy user using Hive ST
HiveServer 2
Hive creates map reduce using NN ST
A
B
C
HDFS
Client gets query result
Beeline Client
Hive gets Namenode (NN) service ticket
Knox gets service ticket for Hive
KDC
61
© Hortonworks Inc. 2011 – 2016. All Rights Reserved

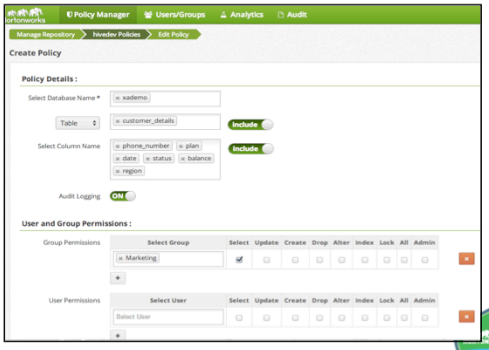

Typical Flow – Add Wire and File Encryption
Active Directory
Ranger
SSL
SSL
Original request w/user id/ password
Apache Knox
Knox runs as proxy user using Hive ST
SSL
SSL
HiveServer 2
Hive creates map reduce using NN ST
SASL
A
B
C
HDFS
Client gets query result
Beeline Client
Hive gets Namenode (NN) service ticket
Knox gets service ticket for Hive
KDC
62
© Hortonworks Inc. 2011 – 2016. All Rights Reserved

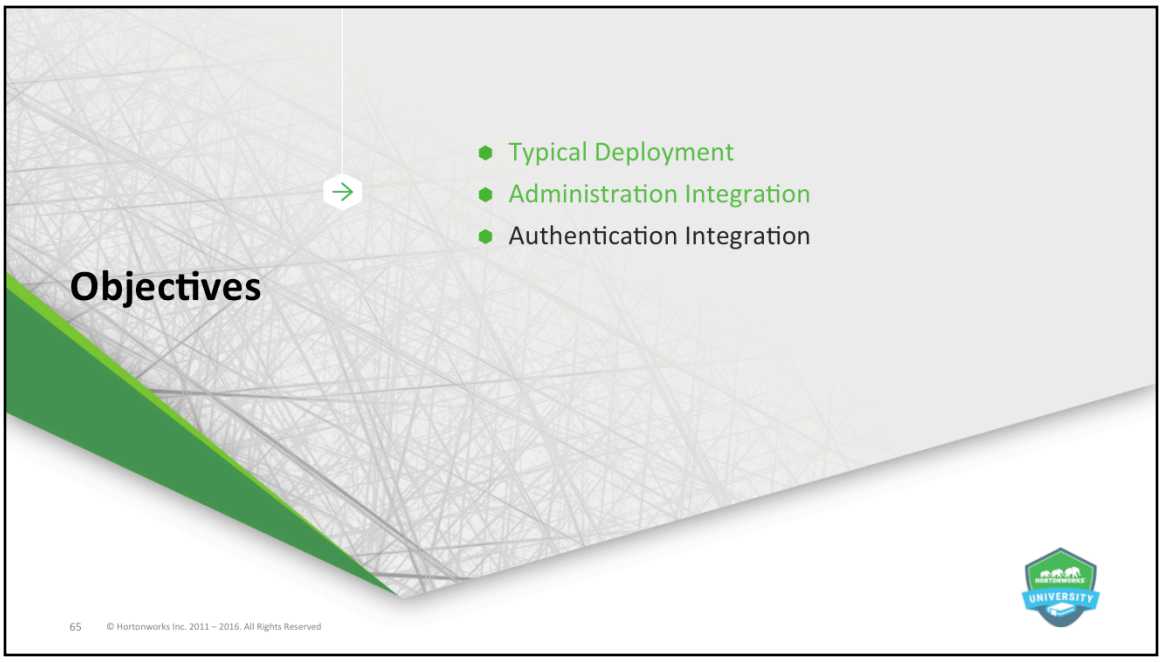

Typical Deployment
Administration Integration
Objectives
63
© Hortonworks Inc. 2011 – 2016. All Rights Reserved

Administration – Apache Ranger

- Central Security Administration Provided Through Console
- Single Pane of Glass for Security Administrator
- Console Ensures Consistent Security Policy Coverage Across Entire Hadoop Stack






Objectives
Typical Deployment
Administration Integration
Authentication Integration
65
© Hortonworks Inc. 2011 – 2016. All Rights Reserved

Authentication – Apache Knox

- Gateway system to extend the reach of Apache Hadoop services to Outside users
- Simplifies Hadoop security for users who access cluster data/execute jobs.
- Integrates with Identity Management and SSO systems used in enterprises and
- Allows identity from these systems be used for access to Hadoop clusters
- Gateway Can Provide Security for Multiple Hadoop Clusters, with the Following advantages:
  - Simplifies access: Extends Hadoop's REST/HTTP services by encapsulating Kerberos to within the Cluster.
  - Enhances security: Exposes Hadoop's REST/HTTP services without revealing network details, providing SSL out of the box.
  - Centralized control: Enforces REST API security centrally, routing requests to multiple Hadoop clusters.
  - Enterprise integration: Supports LDAP, Active Directory, SSO, SAML and other authentication systems.

The Apache Knox Gateway ("Knox") is a system to extend the reach of Apache™Hadoop® services to users outside of a Hadoop cluster without reducing Hadoop Security. Knox also simplifies Hadoop security for users who access the cluster data and execute jobs.

Knox integrates with Identity Management and SSO systems used in enterprises and allows identity from these systems be used for access to Hadoop clusters.

Knox Gateways provides security for multiple Hadoop clusters, with these advantages:

Simplifies access: Extends Hadoop's REST/HTTP services by encapsulating Kerberos to within the Cluster.

Enhances security: Exposes Hadoop's REST/HTTP services without revealing network details, providing SSL out of the box.

Authentication – Apache Knox

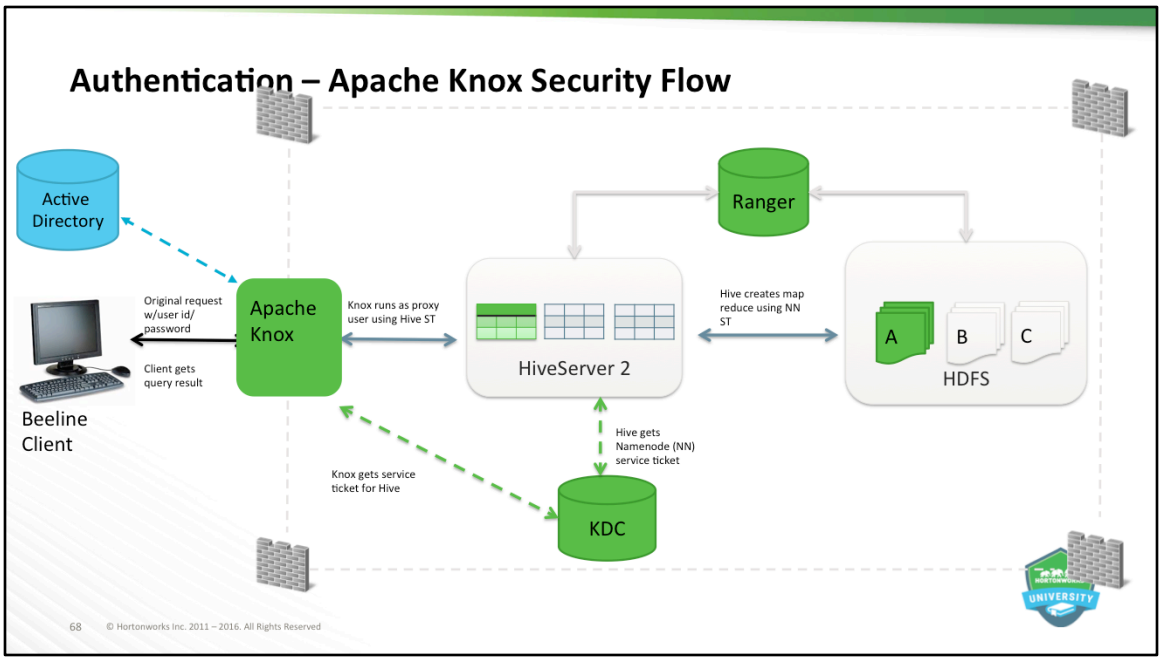
- Used with both Unsecured and Secured Kerberos Clusters
- Kerberos secured clusters provides an enterprise security solution that:
  - Integrates well with enterprise identity management solutions
  - Protects the details of the Hadoop cluster deployment
    - (hosts and ports are hidden from end users)
  - Simplifies the number of services with which a client needs to interact

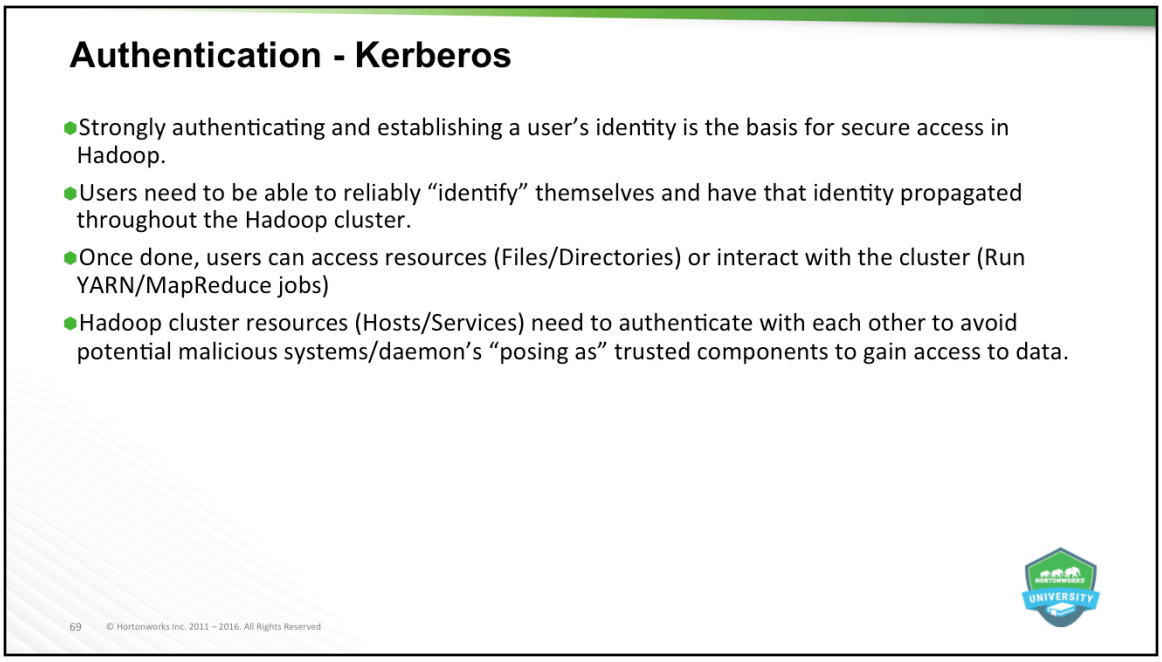
Knox can be used with both unsecured Hadoop clusters, and Kerberos secured clusters. In an enterprise solution that employs Kerberos secured clusters, the Apache Knox Gateway provides an enterprise security solution that:

Integrates well with enterprise identity management solutions

Protects the details of the Hadoop cluster deployment (hosts and ports are hidden from end users)

Simplifies the number of services with which a client needs to interact

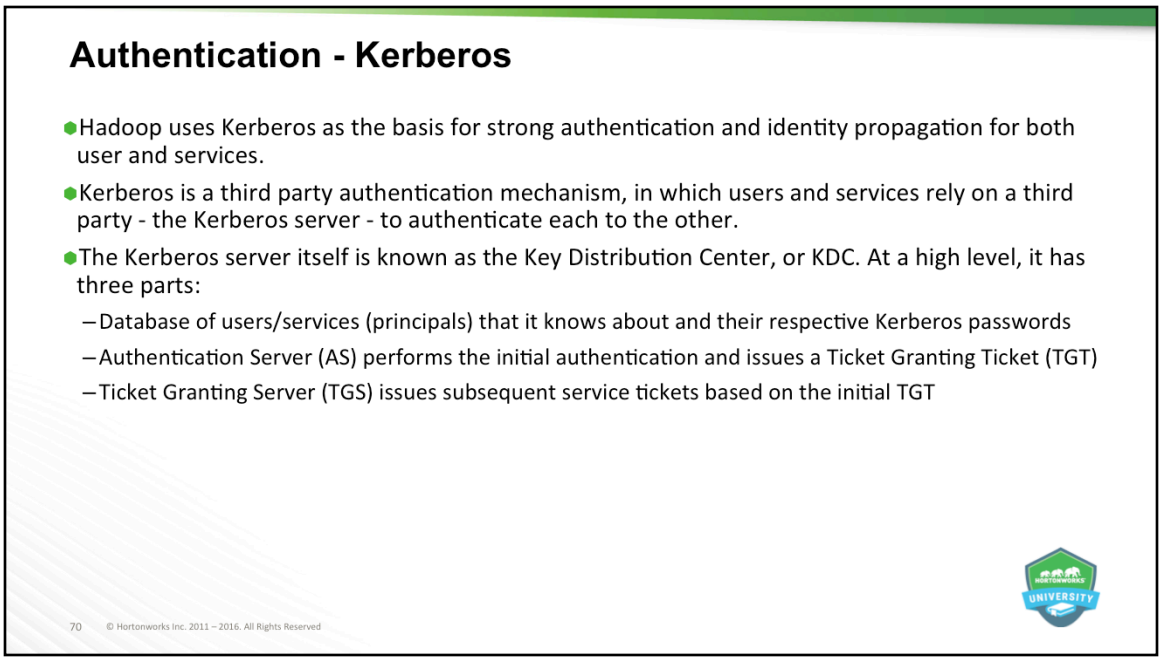

Authentication – Apache Knox Security Flow
Active Directory
Ranger
Original request w/user id/ password
Client gets query result
Apache Knox
Knox runs as proxy user using Hive ST
HiveServer 2
Hive creates map reduce using NN ST
A
B
C
HDFS
Beeline Client
Hive gets Namenode (NN) service ticket
Knox gets service ticket for Hive
KDC
68
© Hortonworks Inc. 2011 – 2016. All Rights Reserved



Kerberos Overview

Strongly authenticating and establishing a user's identity is the basis for secure access in Hadoop. Users need to be able to reliably "identify" themselves and then have that identity propagated throughout the Hadoop cluster. Once this is done, those users can access resources (such as files or directories) or interact with the cluster (like running MapReduce jobs). Besides users, Hadoop cluster resources themselves (such as Hosts and Services) need to authenticate with each other to avoid potential malicious systems or daemon's "posing as" trusted components of the cluster to gain access to data.

Hadoop uses Kerberos as the basis for strong authentication and identity propagation for both user and services. Kerberos is a third party authentication mechanism, in which users and services rely on a third party - the Kerberos server - to authenticate each to the other. The Kerberos server itself is known as the Key Distribution Center, or KDC. At a high level, it has three parts:



Kerberos Overview

Strongly authenticating and establishing a user's identity is the basis for secure access in Hadoop. Users need to be able to reliably "identify" themselves and then have that identity propagated throughout the Hadoop cluster. Once this is done, those users can access resources (such as files or directories) or interact with the cluster (like running MapReduce jobs). Besides users, Hadoop cluster resources themselves (such as Hosts and Services) need to authenticate with each other to avoid potential malicious systems or daemon's "posing as" trusted components of the cluster to gain access to data.

Hadoop uses Kerberos as the basis for strong authentication and identity propagation for both user and services. Kerberos is a third party authentication mechanism, in which users and services rely on a third party - the Kerberos server - to authenticate each to the other. The Kerberos server itself is known as the Key Distribution Center, or KDC. At a high level, it has three parts:

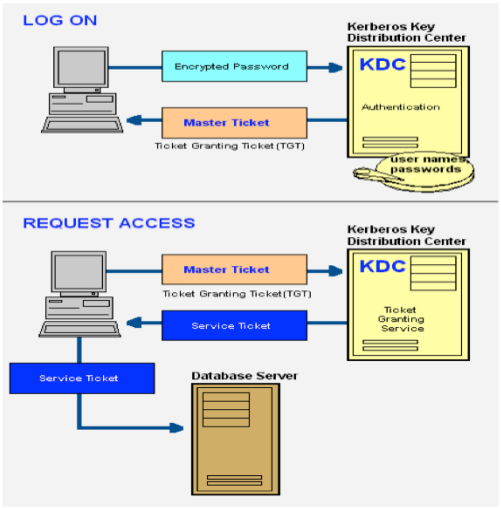
Authentication - Kerberos

- User principal requests authentication from AS
- The AS returns encrypted TGT using the user principal's Kerberos password
- Password known only to the user principal and the AS
- User principal decrypts TGT locally using its Kerberos password
- User principal uses the TGT to get service tickets from the TGS until the ticket expires
- Service tickets allow a principal to access various services
- Cluster resources (hosts/services) cannot provide a password each time to decrypt the TGT, a special file, called a keytab, contains resource principal's authentication credentials
- A Realm has Control over a Set of Hosts/Users/Services

A user principal requests authentication from the AS. The AS returns a TGT that is encrypted using the user principal's Kerberos password, which is known only to the user principal and the AS. The user principal decrypts the TGT locally using its Kerberos password, and from that point forward, until the ticket expires, the user principal can use the TGT to get service tickets from the TGS. Service tickets are what allow a principal to access various services.

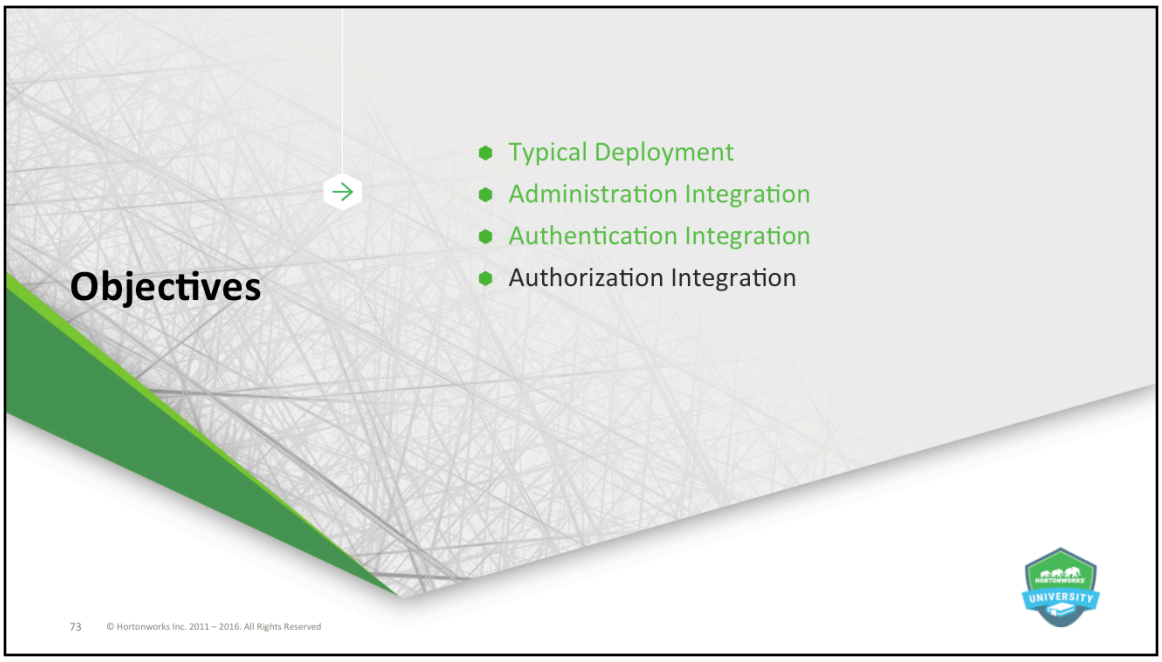
Because cluster resources (hosts or services) cannot provide a password each time to decrypt the TGT, they use a special file, called a keytab, which contains the resource principal's authentication credentials. The set of hosts, users, and services over which the Kerberos server has control is called a realm.

Authentication - Kerberos



A user principal requests authentication from the AS. The AS returns a TGT that is encrypted using the user principal's Kerberos password, which is known only to the user principal and the AS. The user principal decrypts the TGT locally using its Kerberos password, and from that point forward, until the ticket expires, the user principal can use the TGT to get service tickets from the TGS. Service tickets are what allow a principal to access various services.

Because cluster resources (hosts or services) cannot provide a password each time to decrypt the TGT, they use a special file, called a keytab, which contains the resource principal's authentication credentials. The set of hosts, users, and services over which the Kerberos server has control is called a realm.


Objectives
Typical Deployment
Administration Integration
Authentication Integration
Authorization Integration
73
© Hortonworks Inc. 2011 – 2016. All Rights Reserved

Authorization – Apache Ranger

- Supports Fine-grained Authorization for following Apache projects:
- Apache Hadoop/HDFS
  - Provides Plugin for NameNode
  - Make Decision on User Request – Authorized/UnAuthorized
  - Collects Access Request for Auditing
  - Enforce Policies Available in Policy Database
  - Create Policy for Specific Resources – Folders/Files Assign Permission (RWX) to Users/Group
  - Policies Stored in Policy Manager
  - Independent from Native Permission

How does Ranger provide authorization in Apache Hadoop?

Ranger provides a plugin for Apache Hadoop, specifically for the NameNode as part of the authorization method. Ranger' plugin is in the path of the user request and is able to make a decision on whether the user request shoud be authorized. The plugin also collects access request details required for auditing

Ranger will enforce the security policies available in the policy database. Users can create a security policy for a specific set of resources (one or more folders and/or files) and assign specific set of permissions (e.g: read, write, execute) to a specific set of users and/or groups. The security policies are stored our policy manager and are independent from native permissions.

Authorization – Apache Ranger

- Supports Fine-grained Authorization for following Apache projects:
- Apache Hive
  - Provides Plugin for HiveServer2
  - Two Methods of Authorization
    - Storage Based
    - SQL Standard
      - Provide Grant/Revoke at Database and Table Level
      - Commands Familiar to DBA Administration
  - Centralized Interface
    - Granular Access Control at Column Level
    - Use Wildcard in Resource Names

How does Ranger provide authorization in Apache Hive?

Ranger plugin is enabled in Hiveserver2 as part of the authorization

How does Ranger authorization compare to SQL standard authorization?

Apache Hive currently provides two methods of authorization, Storage based authorization and SQL standard authorization, which was introduced in Hive 13. SQL standard authorization provides grant/revoke functionality at database, table level. The commands would be familiar to a DBA admin. Ranger provides a centralized authorization interface for Hive and provides more granular access control at column level through the Hive plugin. Ranger also provides ability to use wildcard in resource names within the policy.

Authorization – Apache Ranger

- Supports Fine-grained Authorization for following Apache projects:
- Apache Hbase
  - Provides coprocessor and Includes Logic to Perform Authorization Check
- Apache Storm
  - Plugin Acts as Authorizer within Nimbus Server
  - Can Authorize all Incoming Request Based on Policies
  - User Can Define Policies on Topologies
- Apache Knox
  - Provides Service Level Authorization Users/Groups
  - ACL's Stored Locally
  - Plugin to Enable Administration of Policies thru Central UI/REST API's

How does Ranger provide authorization in Apache Hbase?

Ranger provides a coprocessor which added to HBase, and includes the logic to perform authorization check and collect audit data.

How does Ranger provide authorization in Apache Knox?

Apache Knox currently provides a service level authorization for users/groups. These acls are stored locally in a file. Ranger has built a plugin for Knox to enable administration of these policies through central UI/REST APIs as well as detailed auditing of Knox user access.

Authorization – Apache Ranger

- Supports Fine-grained Authorization for following Apache projects:
- Apache Kafka
  - Manage ACL's per Topic by Users/Groups
  - Control who can Write To/Read From Topic
  - Supports IP Address Based Permission to Publish/Subscribe

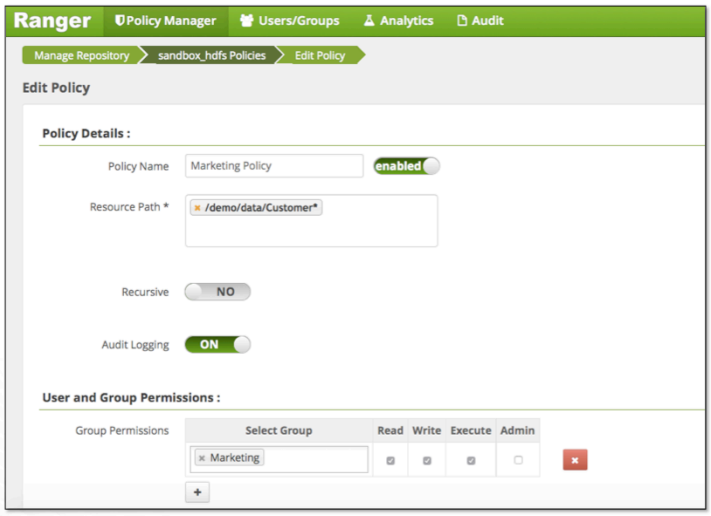
How does Ranger provide authorization in Apache Hbase?

Ranger provides a coprocessor which added to HBase, and includes the logic to perform authorization check and collect audit data.

How does Ranger provide authorization in Apache Knox?

Apache Knox currently provides a service level authorization for users/groups. These acls are stored locally in a file. Ranger has built a plugin for Knox to enable administration of these policies through central UI/REST APIs as well as detailed auditing of Knox user access.

Authorization – Apache Ranger



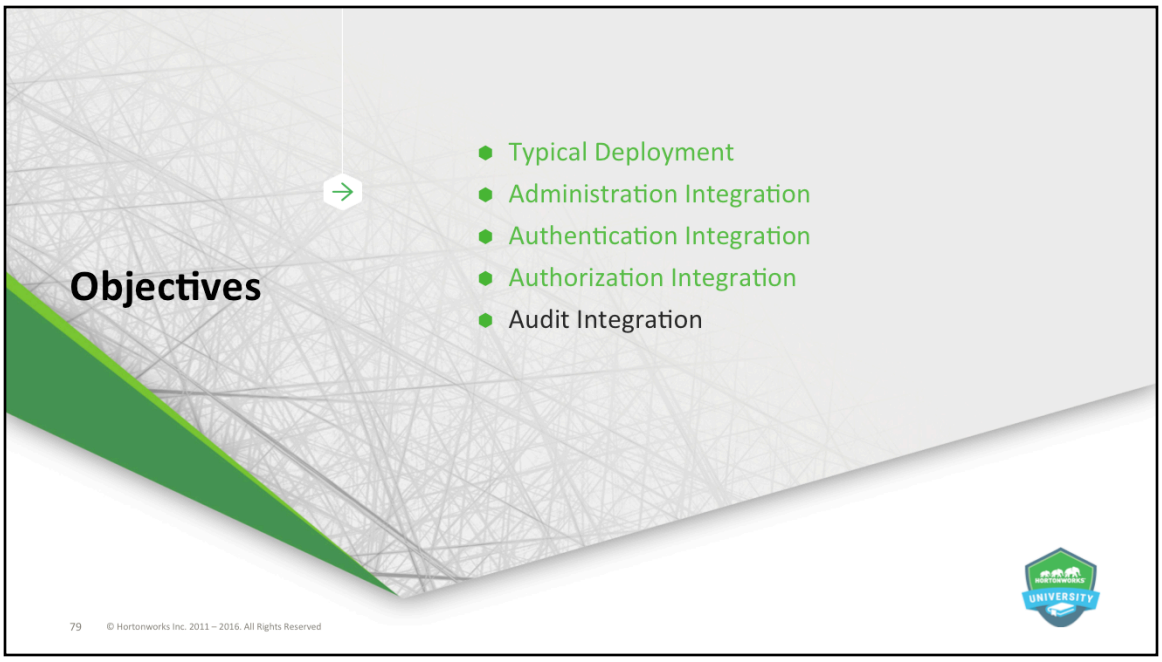
How does Ranger provide authorization in Apache Hadoop?

Ranger provides a plugin for Apache Hadoop, specifically for the NameNode as part of the authorization method. Ranger' plugin is in the path of the user request and is able to make a decision on whether the user request shoud be authorized. The plugin also collects access request details required for auditing

Ranger will enforce the security policies available in the policy database. Users can create a security policy for a specific set of resources (one or more folders and/or files) and assign specific set of permissions (e.g: read, write, execute) to a specific set of users and/or groups. The security policies are stored our policy manager and are independent from native permissions.

How does Ranger provide authorization in Apache Knox?

Apache Knox currently provides a service level authorization for users/groups. These acls are stored locally in a file. Ranger has built a plugin for Knox to enable administration of these policies through central UI/REST APIs as well as detailed auditing of Knox user access.


Objectives
Typical Deployment
Administration Integration
Authentication Integration
Authorization Integration
Audit Integration
79
© Hortonworks Inc. 2011 – 2016. All Rights Reserved

Audit – Apache Ranger

- Supports auditing for following Apache projects:
- Apache Hadoop
  - Provides Plugin for NameNode
  - Plugin Collect Access Request Details
- Apache Hive
- Apache HBase
- Apache Kafka
- Apache YARN
- Apache Storm
- Apache Solr
- Apache Knox
  - Plugin to Enable Administration and Auditing of User Access

How does Ranger provide authorization in Apache Hadoop?

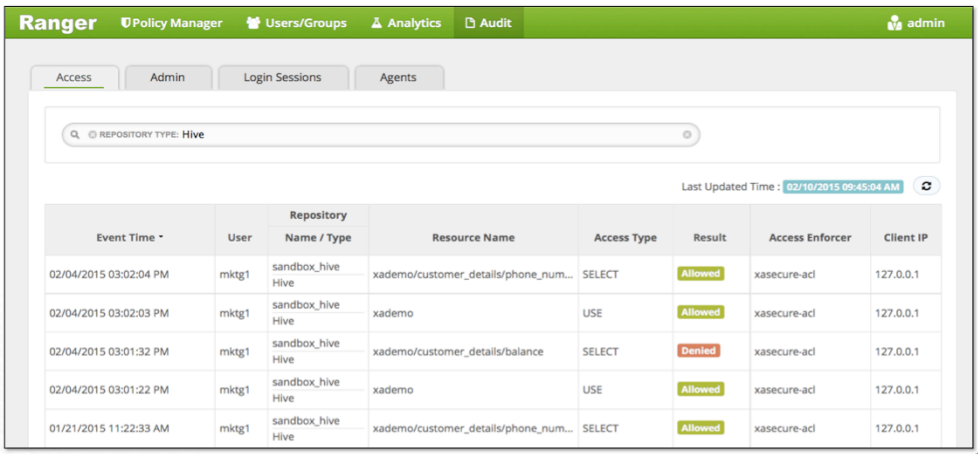
Ranger provides a plugin for Apache Hadoop, specifically for the NameNode as part of the authorization method. Ranger' plugin is in the path of the user request and is able to make a decision on whether the user request shoud be authorized. The plugin also collects access request details required for auditing

Ranger will enforce the security policies available in the policy database. Users can create a security policy for a specific set of resources (one or more folders and/or files) and assign specific set of permissions (e.g: read, write, execute) to a specific set of users and/or groups. The security policies are stored our policy manager and are independent from native permissions.

How does Ranger provide authorization in Apache Knox?

Apache Knox currently provides a service level authorization for users/groups. These acls are stored locally in a file. Ranger has built a plugin for Knox to enable administration of these policies through central UI/REST APIs as well as detailed auditing of Knox user access.

Audit – Apache Ranger



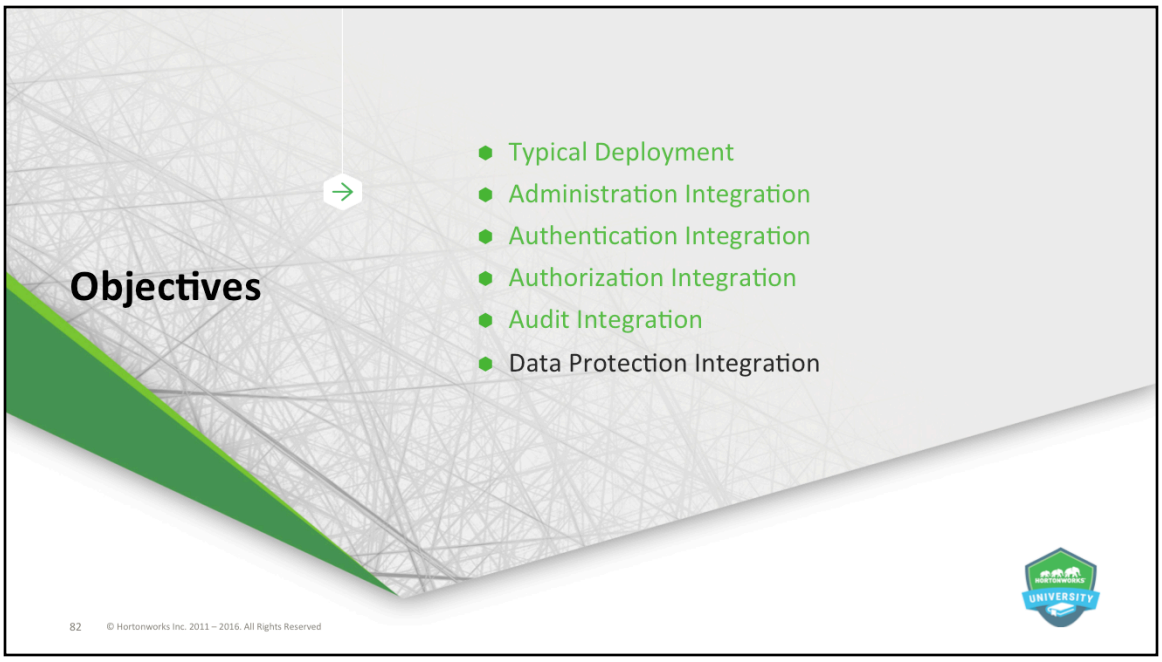
How does Ranger provide authorization in Apache Hadoop?

Ranger provides a plugin for Apache Hadoop, specifically for the NameNode as part of the authorization method. Ranger' plugin is in the path of the user request and is able to make a decision on whether the user request shoud be authorized. The plugin also collects access request details required for auditing

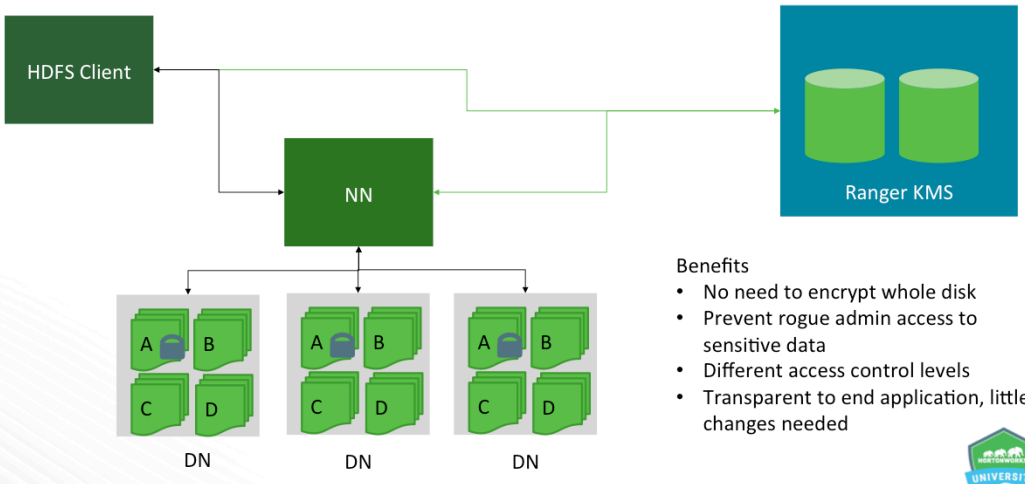
Ranger will enforce the security policies available in the policy database. Users can create a security policy for a specific set of resources (one or more folders and/or files) and assign specific set of permissions (e.g: read, write, execute) to a specific set of users and/or groups. The security policies are stored our policy manager and are independent from native permissions.

How does Ranger provide authorization in Apache Knox?

Apache Knox currently provides a service level authorization for users/groups. These acls are stored locally in a file. Ranger has built a plugin for Knox to enable administration of these policies through central UI/REST APIs as well as detailed auditing of Knox user access.


Objectives
Typical Deployment
Administration Integration
Authentication Integration
Authorization Integration
Audit Integration
Data Protection Integration
82
© Hortonworks Inc. 2011 – 2016. All Rights Reserved

Data Protection – HDFS Encryption



Benefits

- No need to encrypt whole disk
- Prevent rogue admin access to sensitive data
- Different access control levels
- Transparent to end application, little changes needed

Data Protection – HDFS Encryption

- Data At Rest Encryption Implements
  - End-to-End Encryption of Data Read From/Written To HDFS Encryption Zone
  - Data is Encrypted/Decrypted by Client
- HDFS Does Not Have Access to Unencrypted Data/Keys
  - Key Management Server: separate key administrator
- Encryption Involves Several Elements
  - HDFS Encryption Zone (EZ) – Special encrypted HDFS Directory where Data is:
    - Encrypted on Write
    - Decrypted on Read
  - EZ Key – Master Encryption Key associated with all files in an EZ
  - Each EZ Associated with EZ Key when Zone is Created
  - Each File within EZ has Unique Encryption Key – "Data Encryption Key" (DEK)

HDFS data at rest encryption implements end-to-end encryption of data read from and written to HDFS. End-to-end encryption means that data is encrypted and decrypted only by the client. HDFS does not have access to unencrypted data or keys.

HDFS encryption involves several elements:

Encryption key: A new level of permission-based access protection, in addition to standard HDFS permissions.

HDFS encryption zone: A special HDFS directory within which all data is encrypted upon write, and decrypted upon read.

Each encryption zone is associated with an encryption key that is specified when the zone is created.

Each file within an encryption zone has a unique encryption key, called the "data

Data Protection – HDFS Encryption

- HDFS Stores “Encrypted Data Encryption Keys" (EDEKs) as Part of the File's Metadata on the NameNode
- HDFS Does Not Have Access to DEKs (only to EDEK)
- DataNodes See Stream of Encrypted Bytes
- Clients Decrypt an EDEK and Uses the Associated DEK to Encrypt/Decrypt Data During Write/ Read Operations
- Basic Responsibilities of Key Management System (KMS):
  - Provide Access to Stored Encryption Zone Keys
  - Generate/Manage Encryption Zone Keys, and Create EDEKs to be Stored on NameNode
  - Decrypt EDEKs to DEKs for HDFS clients
  - Audit All Access Events

HDFS data at rest encryption implements end-to-end encryption of data read from and written to HDFS. End-to-end encryption means that data is encrypted and decrypted only by the client. HDFS does not have access to unencrypted data or keys.

HDFS encryption involves several elements:

Encryption key: A new level of permission-based access protection, in addition to standard HDFS permissions.

HDFS encryption zone: A special HDFS directory within which all data is encrypted upon write, and decrypted upon read.

Each encryption zone is associated with an encryption key that is specified when the zone is created.

Each file within an encryption zone has a unique encryption key, called the "data

Data Protection – HDFS Encryption

- Encryption Zones - EZ
  - Start as Empty Directory
  - Directory where Contents are Encrypted on Write/Decrypted on Read
  - Encryption is Transparent
  - Cannot be Nested
  - All Files in EZ are Encrypted
- Encryption Zone Keys – EZ Key
  - Unique Key Per Encryption Zone
  - Metadata is Stored in EZ directory
  - Material is Stored in Ranger KMS
  - NameNode has no Access to Key

Data Protection – HDFS Encryption

- Data Encryption Key - DEK
  - Unique Per File
  - Used by Client to Encrypt/Decrypt Data
  - Handled by Client and KMS
  - DataNodes Handle Ciphertext (not plaintext)
- Encrypted Data Encryption Key – EDEK
  - DEK Encrypted with Corresponding EZ Key
  - Generated by Ranger KMS for NameNode
  - Stored in HDFS Metadata as "file xattr"

Data Protection – HDFS Encryption

- Role Separation
  - Access to the Key Encryption/Decryption Process Typically Restricted to End Users
    - Encrypted Keys Safely Stored and Handled by HDFS
    - HDFS Admin User does not have access to the Encrypted Keys
    - So even if HDFS compromised, malicious user only gains access to ciphertext and encrypted keys
  - Recommended: create a separate HDFS admin user for Key Management.
  - Results in Two Types of HDFS Administrator Accounts:
    - HDFS service user: the system-level account associated with HDFS (hdfs by default)
    - HDFS admin user: an account in hdfs supergroup, used by HDFS administrators to configure/manage HDFS
  - For example, to create HDFS admin user called operator1:
    - In Advanced hdfs-site: dfs.cluster.administrators = hdfs,*operator1*
    - In Advanced dbks-site: hadoop.kms.blacklist.DECRYPT_EEK = hdfs,*operator1*

Ranger Key Management Service (Ranger KMS): An open source key management service based on Hadoop's KeyProvider API.

For HDFS encryption, the Ranger KMS has three basic responsibilities:

Provide access to stored encryption zone keys.

Generate and manage encryption zone keys, and create encrypted data keys to be stored in Hadoop.

Audit all access events in Ranger KMS.

Role Separation

Access to the key encryption/decryption process is typically restricted to end users.

Data Protection – HDFS Encryption

In Summary:

- To convert DEK to EDEK (or vice versa): Need EZ Key
- To encrypt or decrypt a file: Need DEK
- EZ key and DEK stored in KMS
- EDEK stored in NameNode
- With HDFS encryption enabled: - Client Needs to:
  - Access HDFS (for data/EDEK)
  - Access KMS (to decrypt EDEK)
  - Access to the DEK (required to read/write files)
  - There are two levels of checks - whether the client has permission to access:
    - The HDFS Directory/File – check performed by NameNode
    - The encryption zone key version – check performed by KMS
- Recommended: Create a separate HDFS admin user for Key Management

Data Protection - Wire Encryption

- Protects Data in Motion
- Hadoop Cluster Typically Communicate Via
  - Remote Procedure Call – RPC -> SASL
  - Hyper Text Transfer Protocol – HTTP -> HTTPS
  - Data Transfer Protocol (UDP) - DTP -> SSL
  - Java Database Connectivity – JDBC -> SASL (QOP)
  - MapReduce Shuffle – HTTP -> HTTPS

Encryption is applied to electronic information in order to ensure its privacy and confidentiality. Wire encryption protects data as it moves into and through Hadoop cluster over RPC, HTTP, Data Transfer Protocol (DTP), and JDBC.

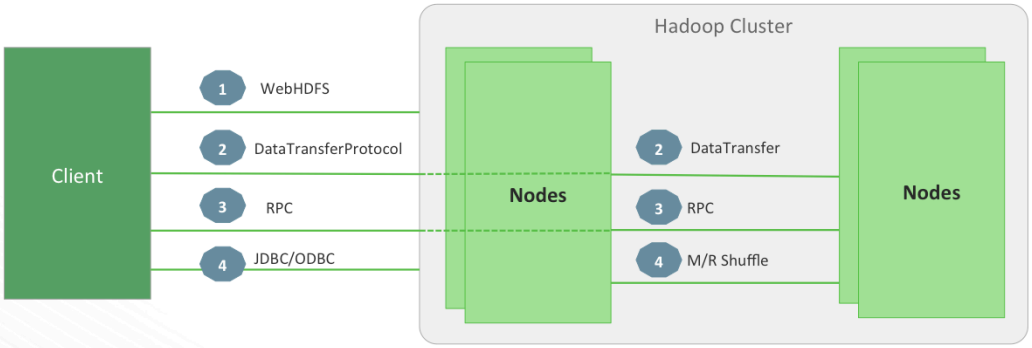
The following describes how the data is protected as it is in motion:

Clients typically communicate directly with the Hadoop cluster and the data can be protected using:

RPC encryption: Clients interacting directly with the Hadoop cluster through RPC. A client uses RPC to connect to the NameNode (NN) to initiate file read and write operations. RPC connections in Hadoop use Java's Simple Authentication & Security Layer (SASL), which supports encryption.

Data Transfer Protocol: The NN gives the client the address of the first DataNode (DN) to read or write the block. The actual data transfer between the client and a DN

Data Protection - Points of Communication



Data Protection - Wire Encryption

- Remote Procedure Call
  - Used to Connect/Communicate to Cluster
  - Connections in Hadoop use Java's Simple Authentication & Security Layer (SASL)
  - SASL Supports Encryption
- Data Transfer Protocol
  - Supports Encryption
  - Utilizes 3DES Algorithm for Encryption - Default
  - Supports rc4 Algorithm
- Hyper Text Transfer Protocol – HTTP
  - Users Typically Interact with Hadoop
  - Application use REST API's/Thrift
  - Encryption Implemented via Secure Sockets Layer - SSL

Encryption is applied to electronic information in order to ensure its privacy and confidentiality. Wire encryption protects data as it moves into and through Hadoop cluster over RPC, HTTP, Data Transfer Protocol (DTP), and JDBC.

The following describes how the data is protected as it is in motion:

Clients typically communicate directly with the Hadoop cluster and the data can be protected using:

RPC encryption: Clients interacting directly with the Hadoop cluster through RPC. A client uses RPC to connect to the NameNode (NN) to initiate file read and write operations. RPC connections in Hadoop use Java's Simple Authentication & Security Layer (SASL), which supports encryption.

Data Transfer Protocol: The NN gives the client the address of the first DataNode (DN) to read or write the block. The actual data transfer between the client and a DN

Data Protection - Wire Encryption

- Java Database Connectivity – JDBC
  - HiveServer2 Implements with Java SASL Protocol's Quality of Protection – QOP
  - Utilizing QOP Setting Data Moving Between JDBC Server & JDBC Client Can Be Encrypted
- MapReduce Shuffle
  - Available Since HDP 2.0
  - HTTPS Utilized During Shuffle Phase
  - Reducer Initiates Connection to Mapper Acting as SSL Client
  - Shuffle Data Encrypted

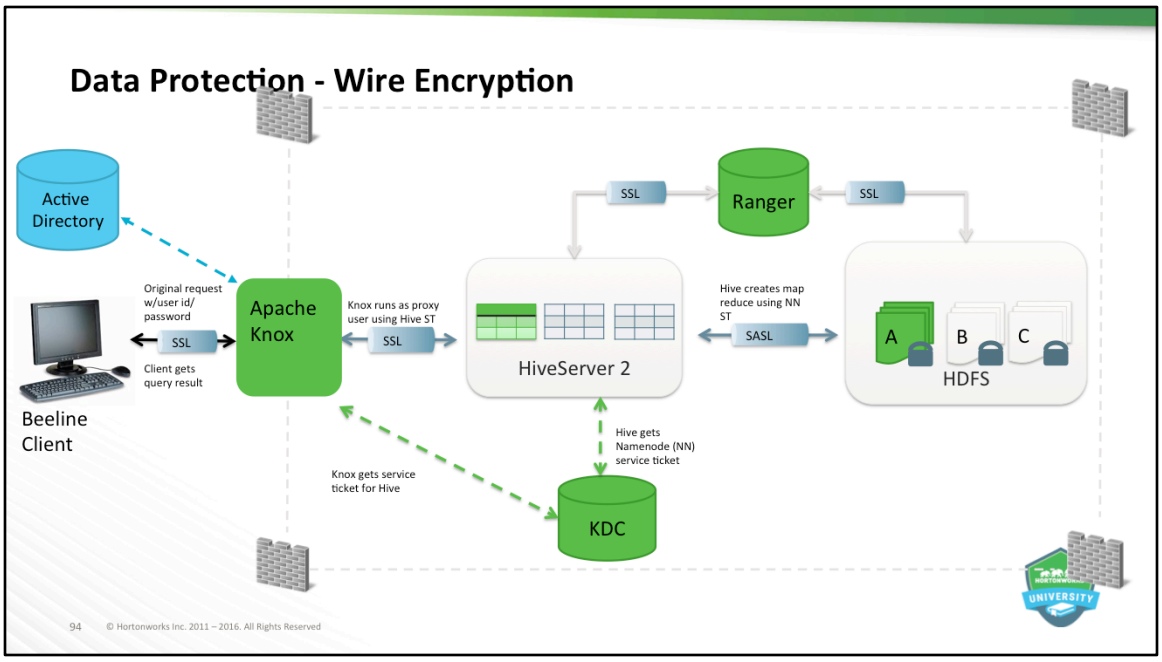
Encryption is applied to electronic information in order to ensure its privacy and confidentiality. Wire encryption protects data as it moves into and through Hadoop cluster over RPC, HTTP, Data Transfer Protocol (DTP), and JDBC.

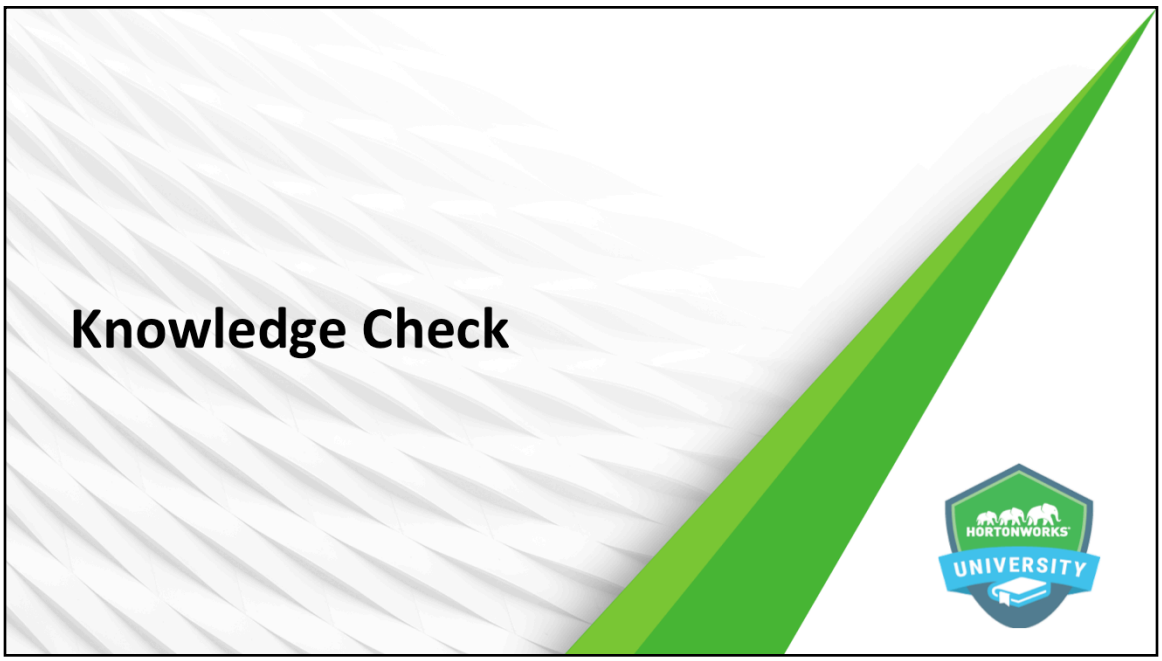
The following describes how the data is protected as it is in motion:

Clients typically communicate directly with the Hadoop cluster and the data can be protected using:

RPC encryption: Clients interacting directly with the Hadoop cluster through RPC. A client uses RPC to connect to the NameNode (NN) to initiate file read and write operations. RPC connections in Hadoop use Java's Simple Authentication & Security Layer (SASL), which supports encryption.

Data Transfer Protocol: The NN gives the client the address of the first DataNode (DN) to read or write the block. The actual data transfer between the client and a DN


Data Protection - Wire Encryption
Active Directory
Ranger
SSL
SSL
Original request w/user id/ password
Apache Knox
Knox runs as proxy user using Hive ST
SSL
SSL
HiveServer 2
Hive creates map reduce using NN ST
SASL
A
B
C
HDFS
Client gets query result
Beeline Client
Hive gets Namenode (NN) service ticket
Knox gets service ticket for Hive
KDC
94
© Hortonworks Inc. 2011 – 2016. All Rights Reserved


Knowledge Check
HORTONWORKS
UNIVERSITY

Questions

1. What component provides central administration for security policy?
2. What component extends the reach of Hadoop to outside users and simplifies security?
3. True/False – Apache Knox can be used with secure and unsecure Hadoop clusters?
4. True/False – NameNode has access to encryption keys?
5. True/False – End-to-end encryption means data is encrypted and decrypted by the NameNode?
6. True/False – HDFS does not have access to unencrypted data or keys?

1. Apache Ranger
2. Apache Knox
3. TRUE
4. FALSE
5. FALSE - See previous question
6. TRUE

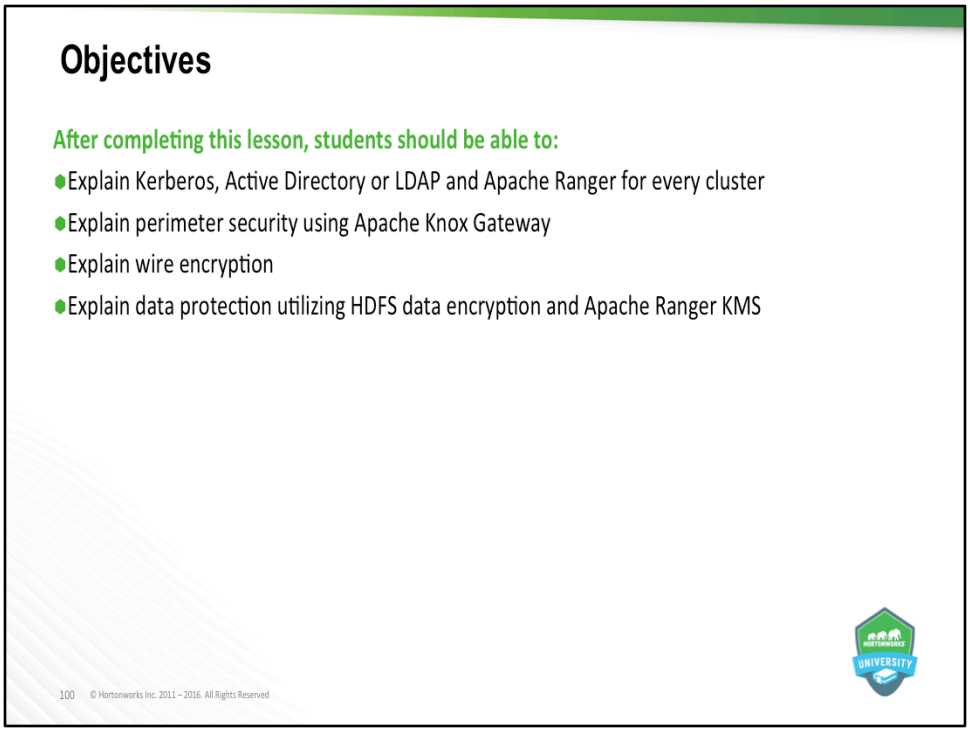

Summary
HORTONWORKS
UNIVERSITY

Summary

- Apache Ranger provides central administration ensuring consistent security policy across the Hadoop Stack.
- Apache Ranger supports fine-grained authorization for multiple Apache projects.
- Apache Knox extends the reach of Apache Hadoop to outside users by simplifying security.
- Apache Knox can be used with secure and unsecure clusters
- Hadoop uses Kerberos as the basis for strong authentication and identity propagation for both users and services.
- HDFS data at rest encryption implements end-to-end encryption of data read from and written to HDFS.
- End-to-end encryption means that data is encrypted and decrypted only by the client and HDFS does not have access to unencrypted data or keys.

This summary page lists some of the main points from this lesson.


What Security Tool for
Each Use Case
HORTONWORKS
UNIVERSITY



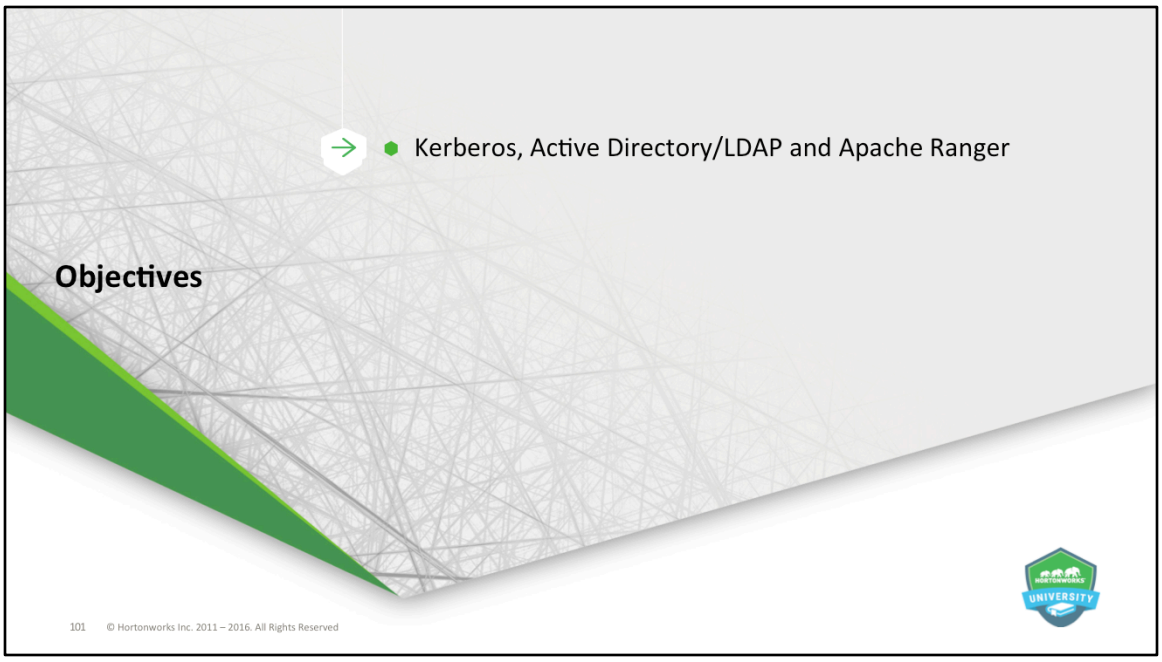
After completing this lesson, students should be able to:

Explain Kerberos, Active Directory or LDAP and Apache Ranger for every cluster

Explain perimeter security using Apache Knox Gateway

Explain wire encryption

Explain data protection utilizing HDFS data encryption and Apache Ranger KMS

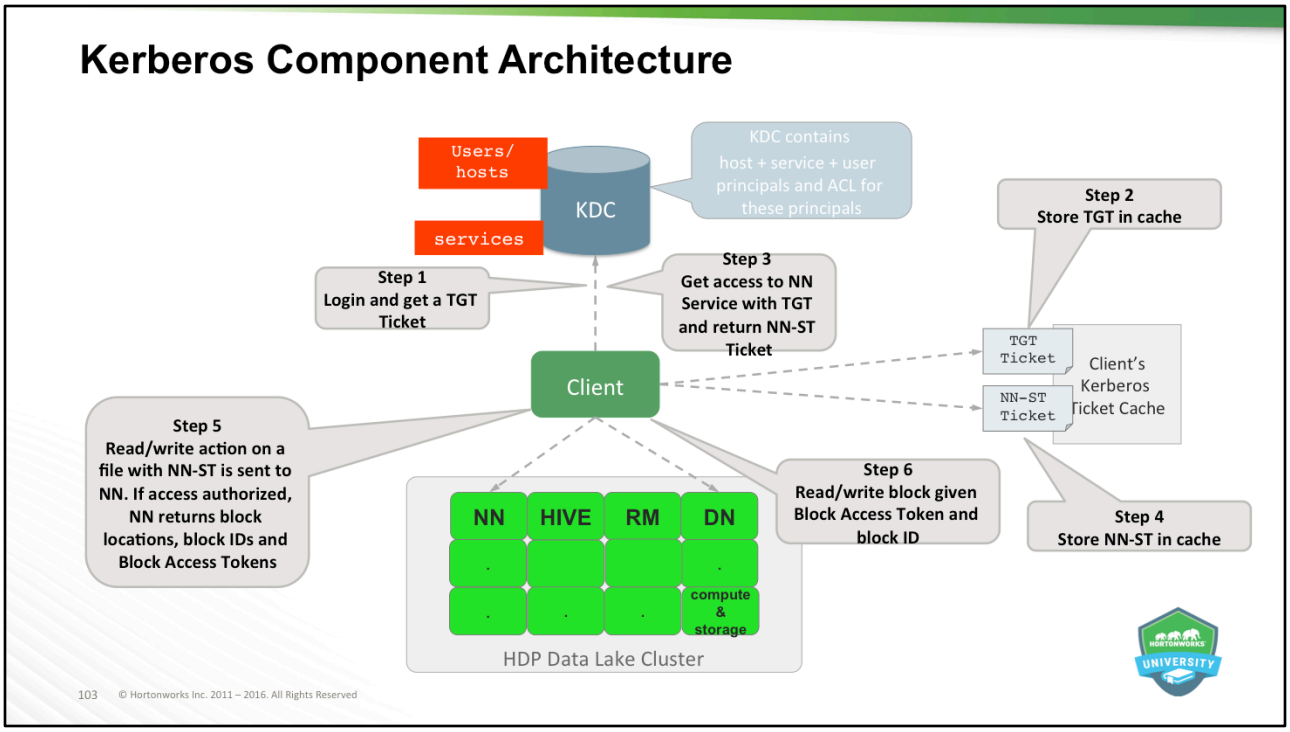

Kerberos, Active Directory/LDAP and Apache Ranger
Objectives
101
© Hortonworks Inc. 2011 – 2016. All Rights Reserved

Kerberos

- Provides strong authentication
- Establishes identity for users, services and host
- Prevents impersonation on unauthorized account
- Supports token delegation model
- Works with existing directory services
- Basis for authorization

Token delegation = TGT is token delegated to user and cached – minimizes the number of times users connect to KDC

Strong Authentication = Password never sent over the wire



SSH in as user

If you directly try hdfs fs –ls to access HDFS …it won't work

Kinit to authenticate from KDC using user password

TGT generated and stored in cache

Now user can access HDFS

Which will generate NN_ST and gets cached

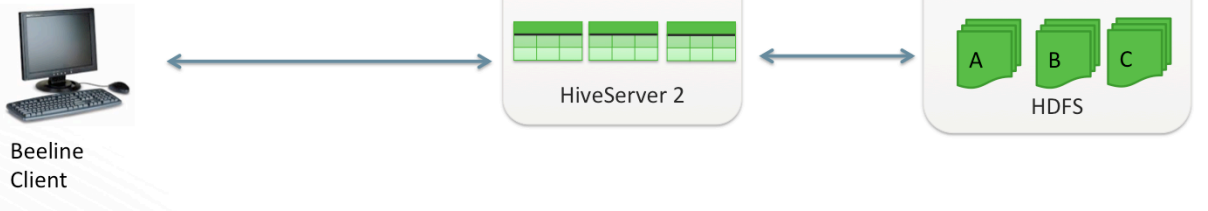
Give NN the ST and checks users has permissions for file/sir

Send back block ids, DN list

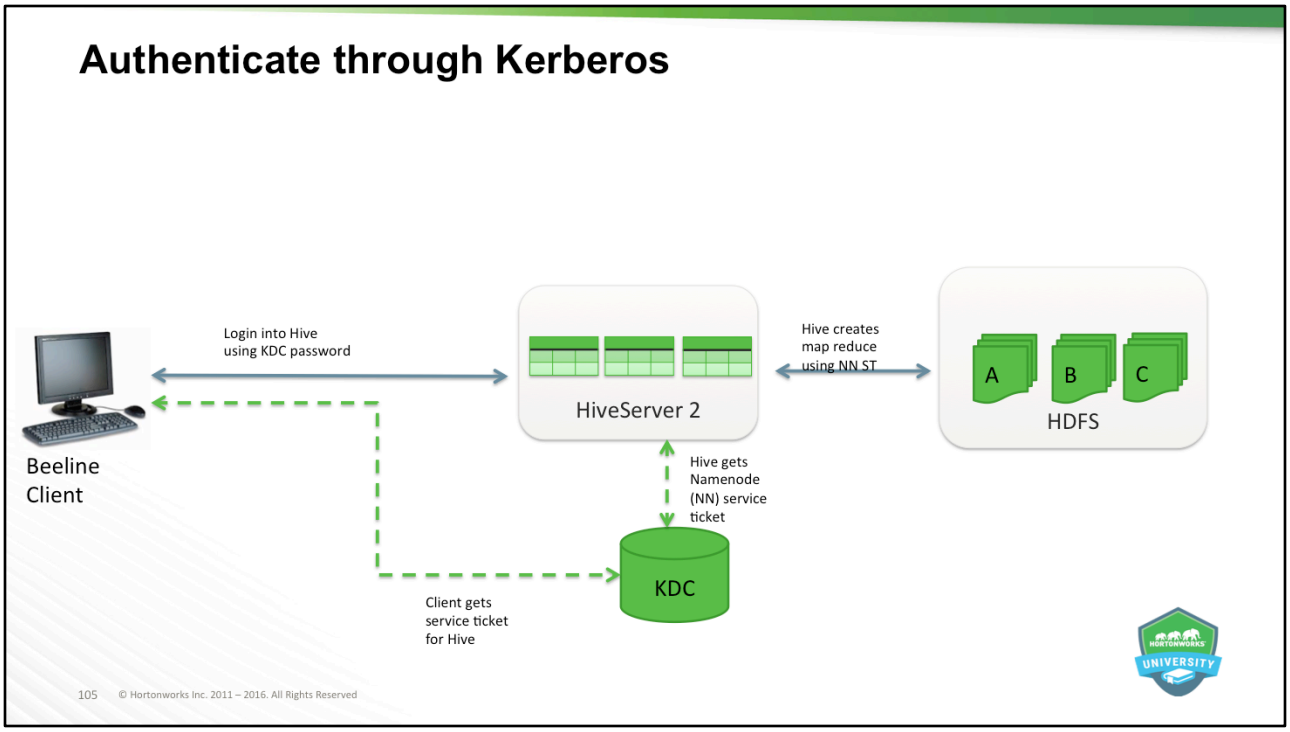
Client requests ST for DNs

Access Bocks in DN

SQL Access through Beeline client



No security

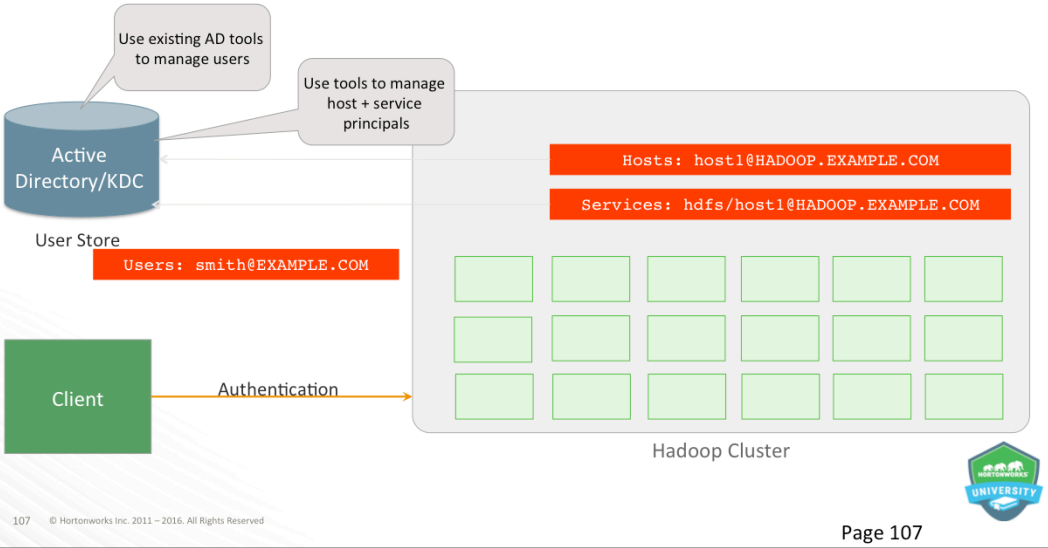


Add Kerberos

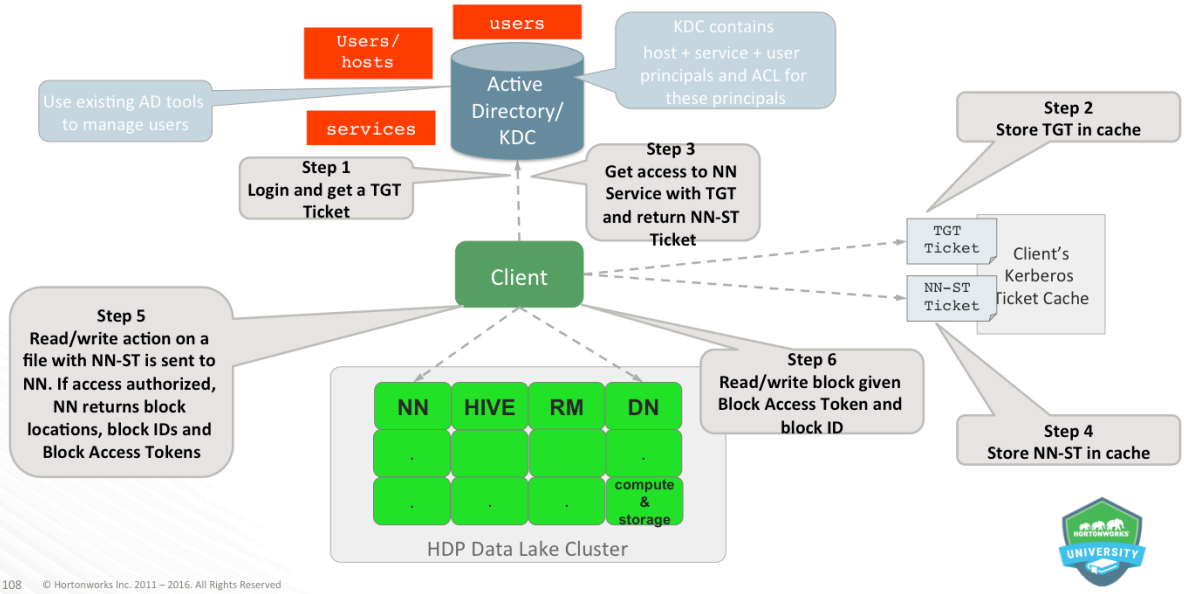
Active Directory

- Leverage existing infrastructure
  - Obtain dedicated AD replica
- Utilize existing tools to manage users
- (Optional) establish one-way cross realm trust from Kerberos KDC
  - For customers who don't want Hadoop principals created in their AD

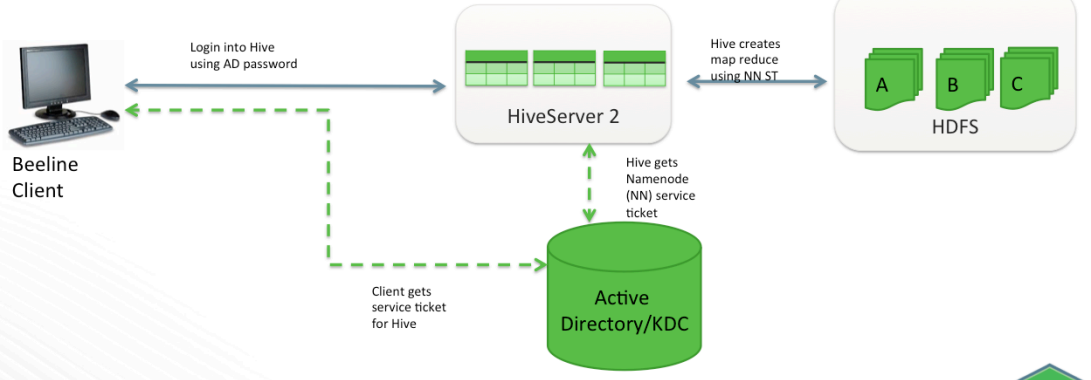
Option 1: Authenticate Directly to Active Directory



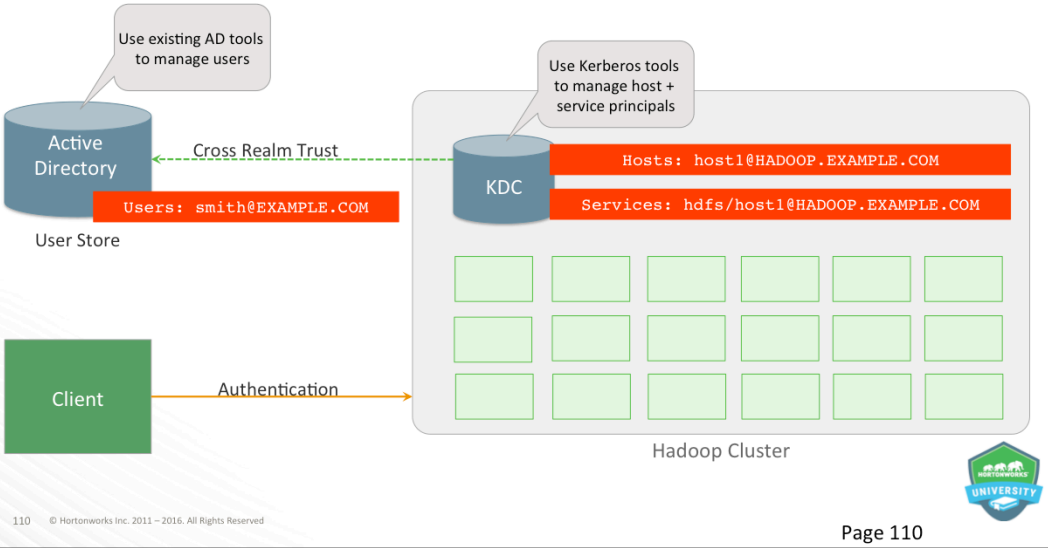
Option 1: Authenticate Directly to Active Directory



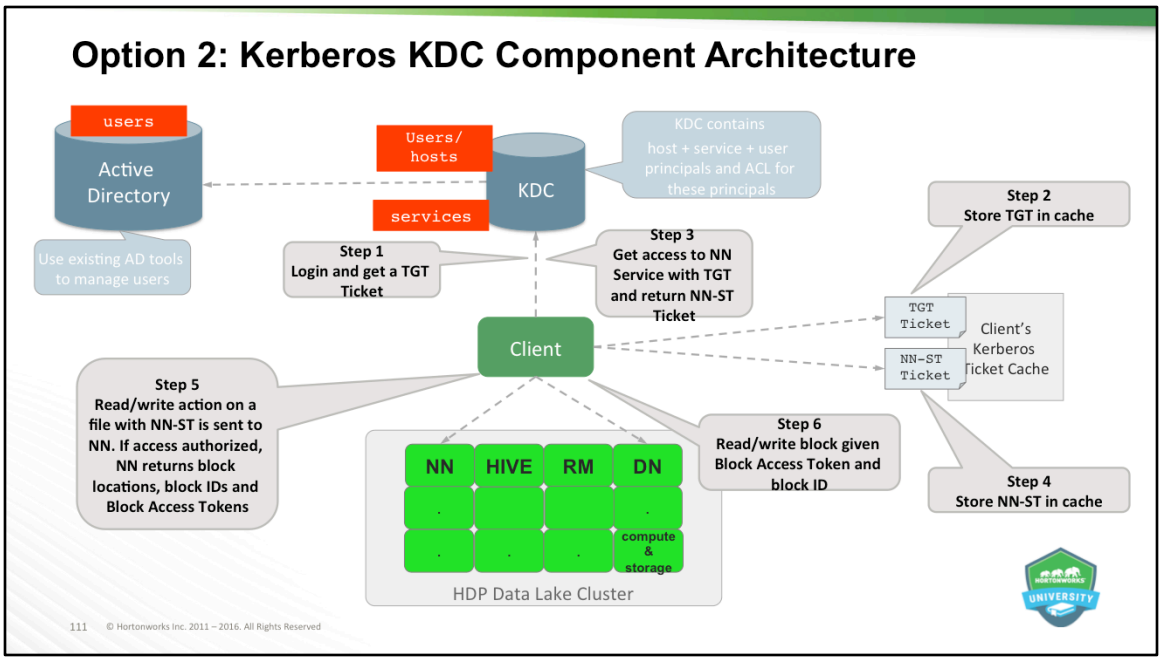
Option 1: Authenticate Directly to Active Directory



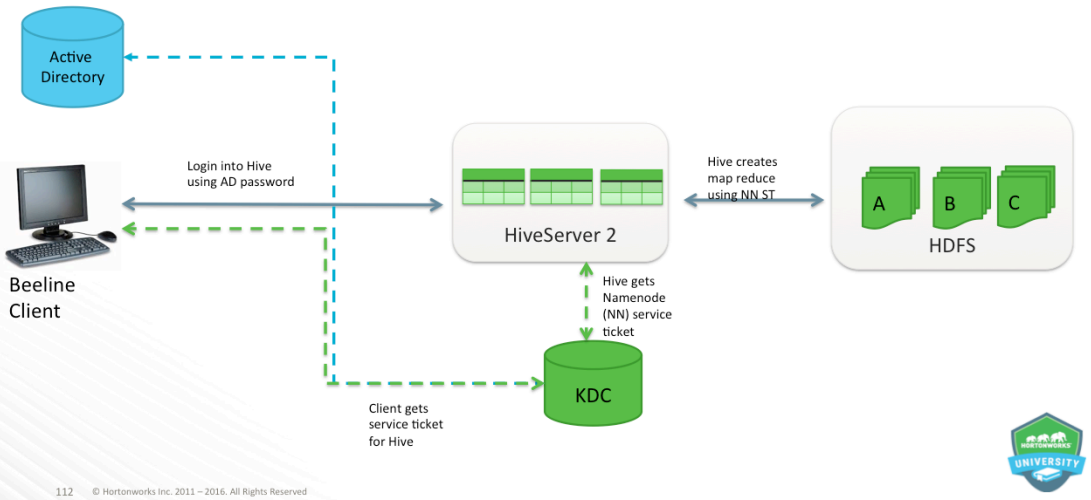
Option 2: Kerberos KDC + Active Directory



Option 2: Kerberos KDC Component Architecture

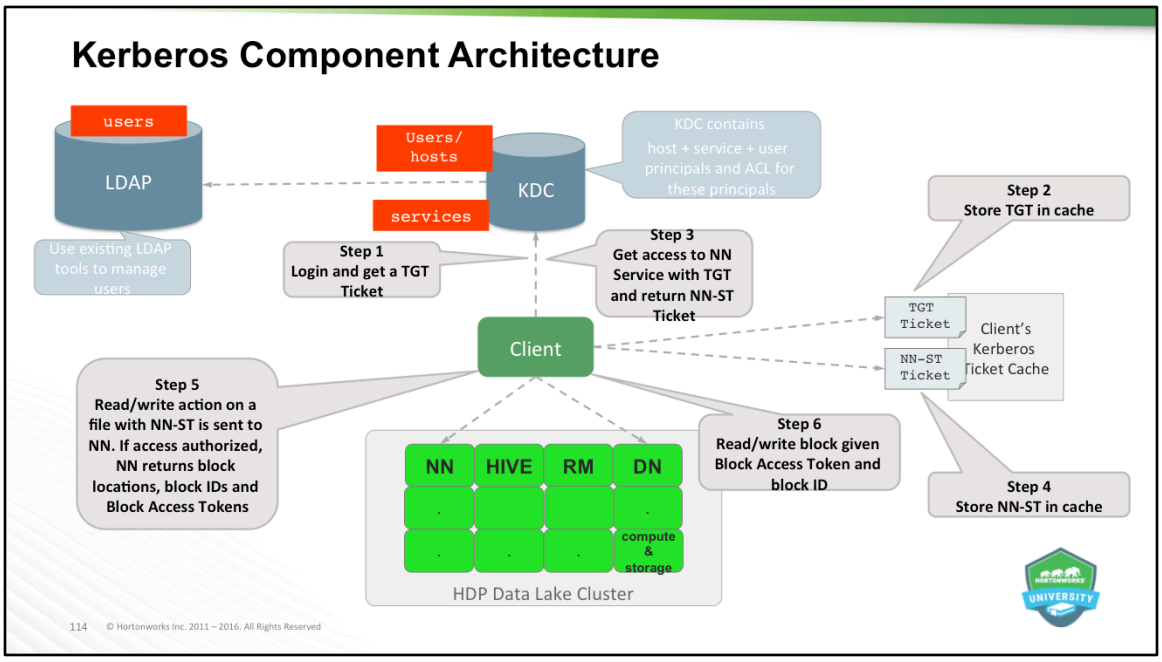


Option 2: Authenticate through Kerberos KDC/AD

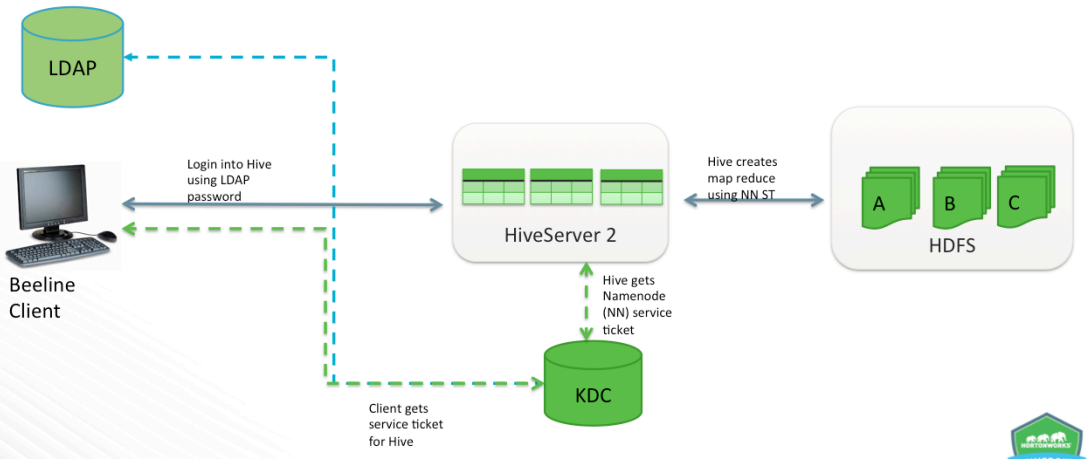


LDAP

- Leverage existing infrastructure
- Utilize existing tools to manage users


Kerberos Component Architecture
users
LDAP
Use existing LDAP tools to manage users
Users/ hosts
services
KDC
KDC contains host + service + user principals and ACL for these principals
Step 1 Login and get a TGT Ticket
Step 3 Get access to NN Service with TGT and return NN-ST Ticket
Step 2 Store TGT in cache
TGT Ticket
NN-ST Ticket
Client's Kerberos Ticket Cache
Client
Step 5 Read/write action on a file with NN-ST is sent to NN. If access authorized, NN returns block locations, block IDs and Block Access Tokens
Step 6 Read/write block given Block Access Token and block ID
Step 4 Store NN-ST in cache
NN
HIVE
RM
DN
compute & storage
HDP Data Lake Cluster
114
© Hortonworks Inc. 2011 – 2016. All Rights Reserved

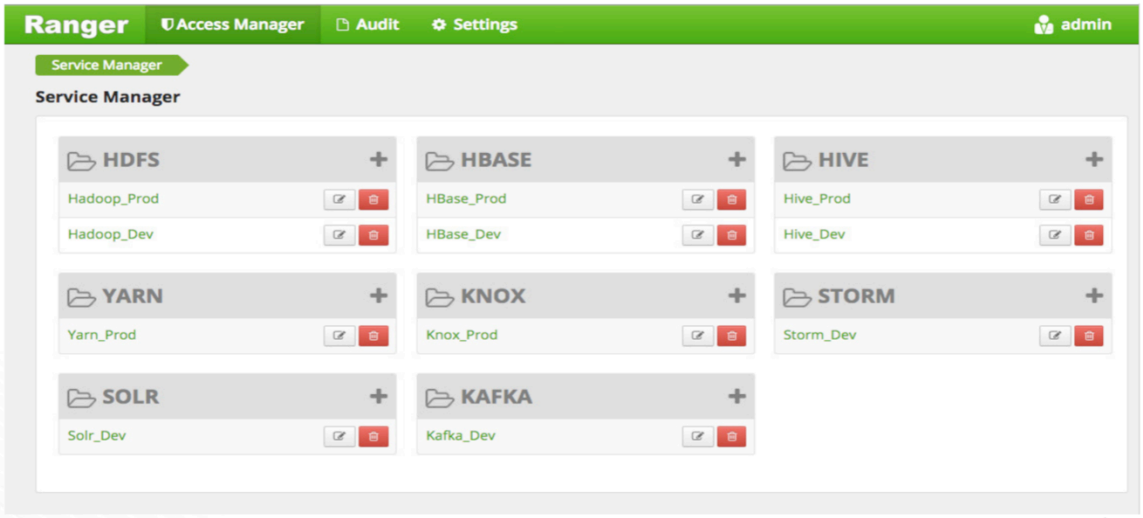
Authenticate through Kerberos/LDAP



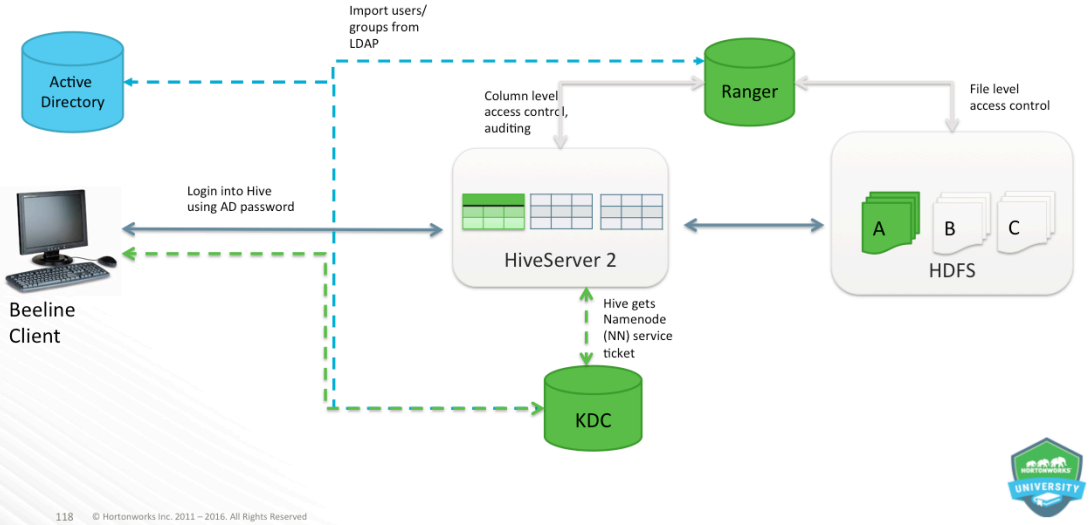
Apache Ranger

- Centralized administration, authorization and auditing
- Single pane of glass for security administrator
- Console ensures consistent security policy coverage across entire Hadoop stack

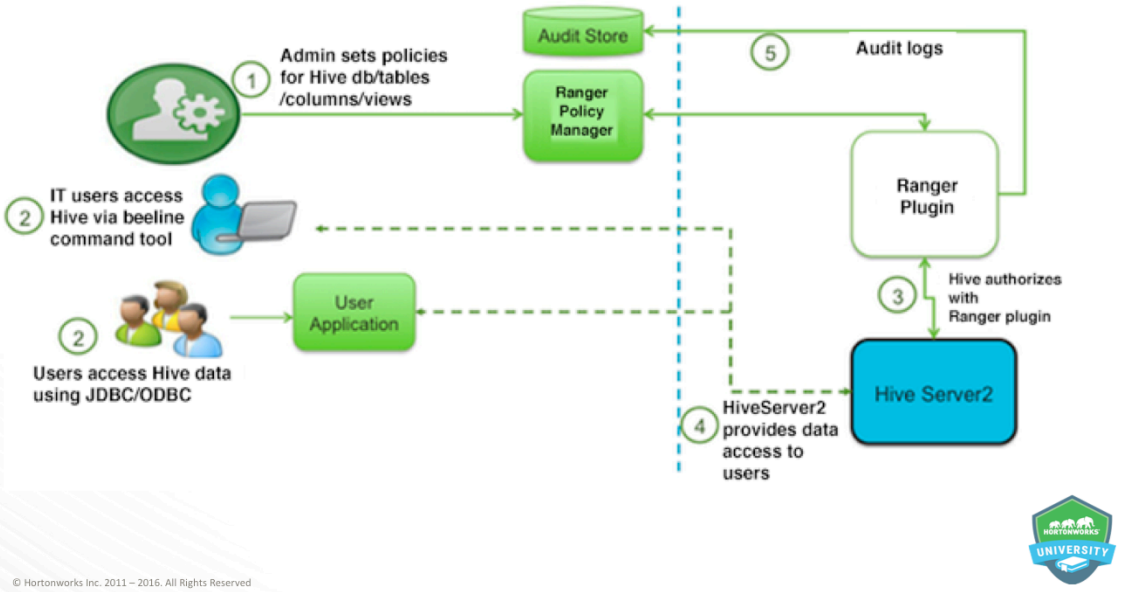
Apache Ranger



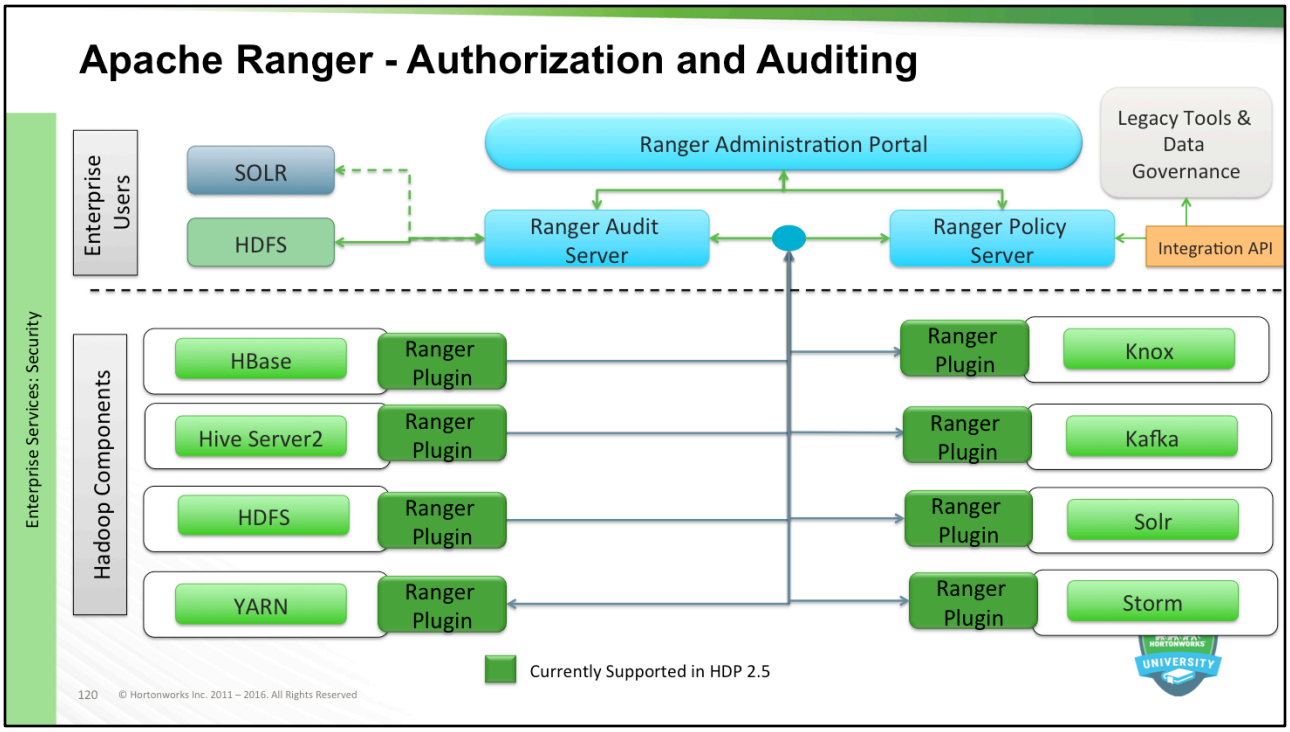
Add Authorization through Ranger



Apache Ranger



Lets take a min to step through how Hive plugin would work from end user perspective
(then talk through each step)



Architectural overview of how it work:

Similar to Hive plugin in earlier slide, plugins for other Hadoop components would work the same way.

Plugins are embedded with the component, so if Admin goes down, policies are still in effect (may not have latest policies downloaded that's all)

No OS administrative overhead (i.e. no new daemons spawned) for each plugin

Objectives

- Kerberos, Active Directory/LDAP and Apache Ranger
- Apache Knox Gateway

Apache Knox Gateway

Enhanced Security

- Protect network details
- SSL for non-SSL services
- WebApp vulnerability filter

Centralized Control

- Central REST API auditing
- Service-level authorization
- Alternative to SSH "edge node"

Simplified Access

- Kerberos encapsulation
- Extends API reach
- Single access point
- Multi-cluster support
- Single SSL certificate

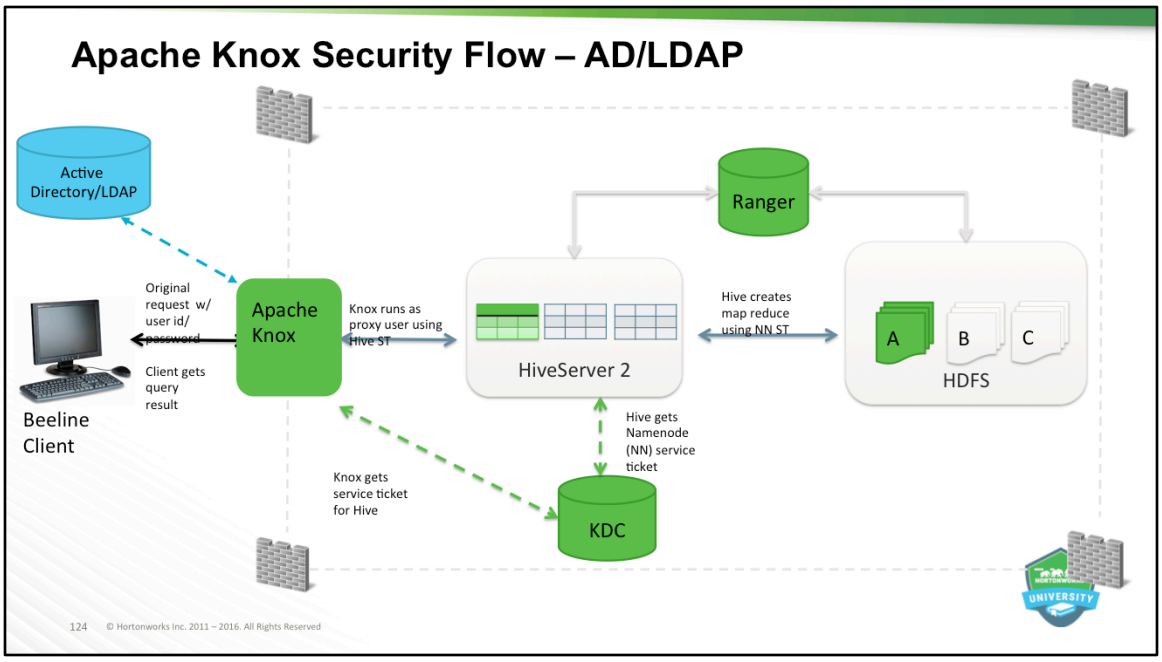
Enterprise Integration

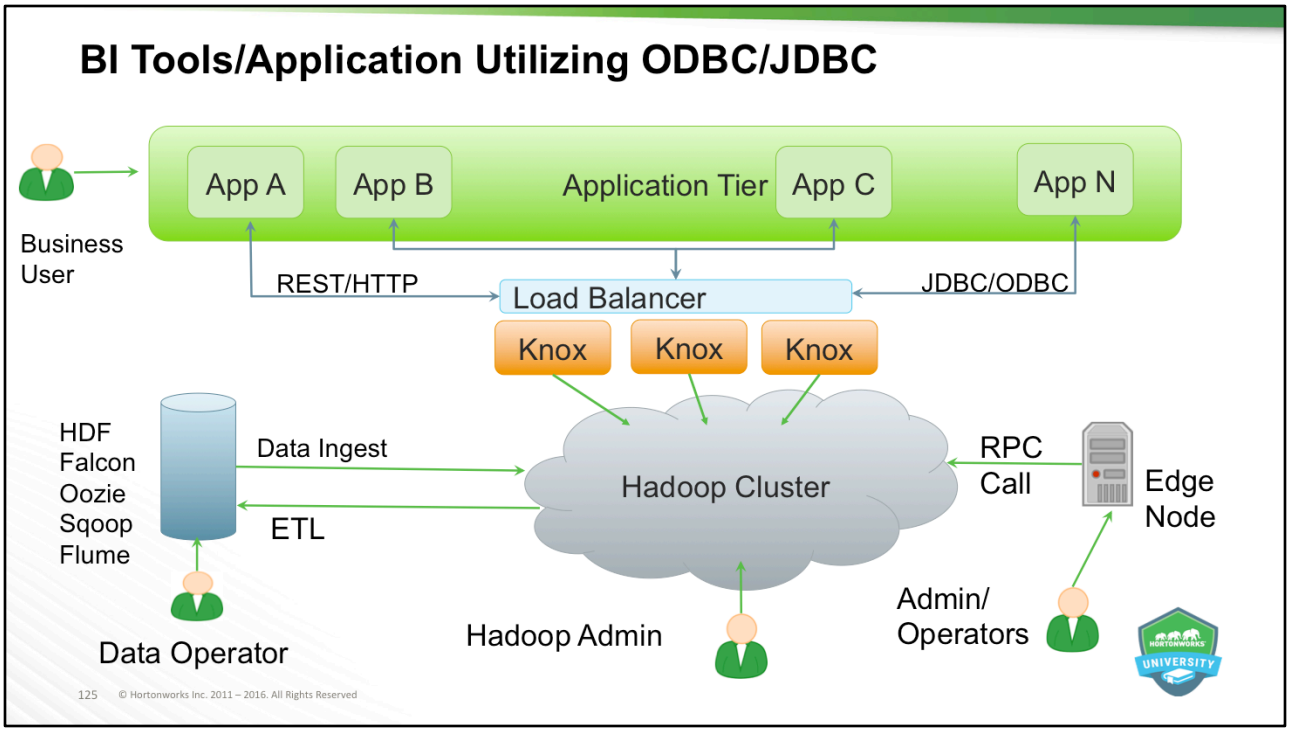
- LDAP integration
- Active Directory integration
- SSO integration
- Apache Shiro extensibility
- Custom extensibility

Simplified access - Extend Hadoop's REST/HTTP services by encapsulating Kerberos within the cluster
Enhanced security - Expose Hadoop's REST/HTTP services without revealing network details, with SSL provided out of box
Centralized control - Centrally enforce REST API security and route requests to multiple Hadoop clusters
Enterprise integration - Support LDAP and Active Directory

Apache Knox Gateway – Key Points

- Streamlining authentication for data access
- Hide internal hdp ports from end points
- Hide Kerberos complexities for end points accessing data via RESTAPIs
- Enable SSL for end point access
- Potentially provide web single sign on capabilities


Apache Knox Security Flow – AD/LDAP
Active Directory/LDAP
Ranger
Original request w/ user id/ password
Apache Knox
Knox runs as proxy user using Hive ST
HiveServer 2
Hive creates map reduce using NN ST
A
B
C
HDFS
Client gets query result
Beeline Client
Hive gets Namenode (NN) service ticket
Knox gets service ticket for Hive
KDC
124
© Hortonworks Inc. 2011 – 2016. All Rights Reserved



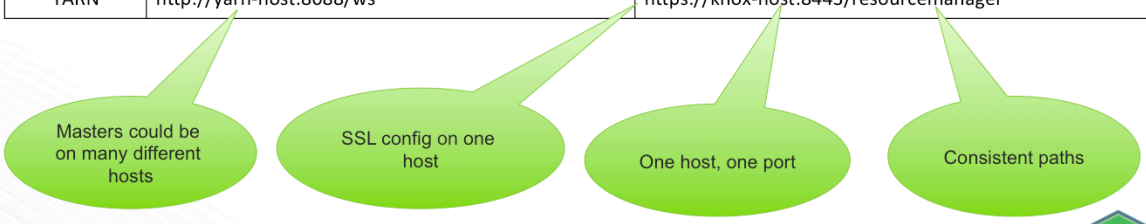
Applications can interact with Knox via REST/HTTP or JDBC/ODBC

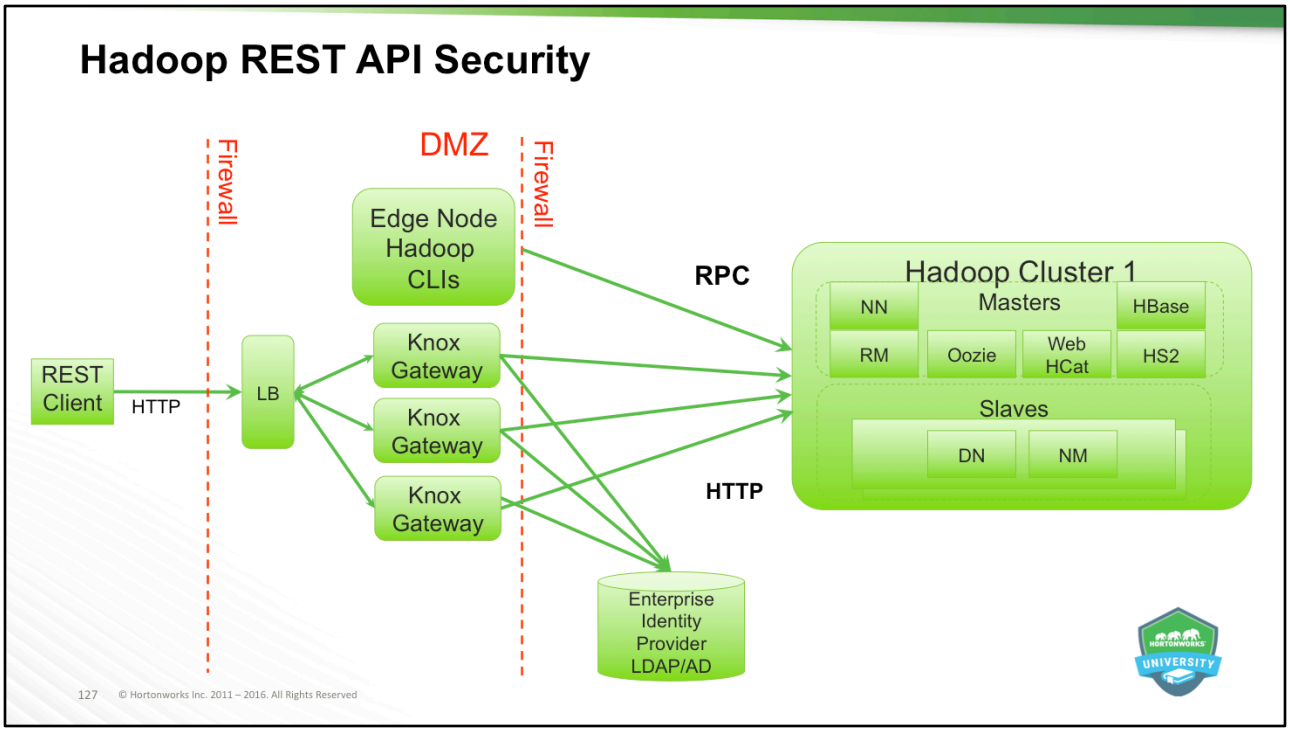
Can run multiple Knox gateways via LoadBalancers

Knox is not meant to bulk import/export of data (would be like drinking water from pool using a straw)
Bulk operations should happen within cluster via HDF, Sqoop etc

Hadoop REST API with Knox

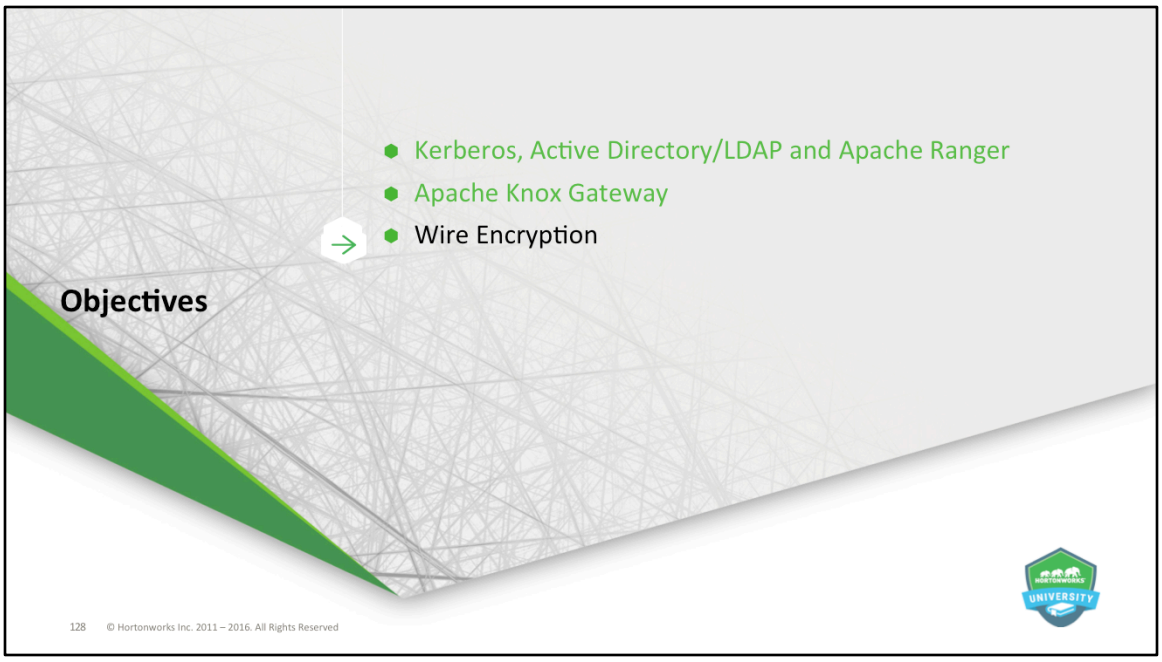
Service	Direct URL	Knox URL
WebHDFS	http://namenode-host:50070/webhdfs	https://knox-host:8443/webhdfs
WebHCat	http://webhcat-host:50111/templeton	https://knox-host:8443/templeton
Oozie	http://oozie-host:11000/oozie	https://knox-host:8443/oozie
HBase	http://hbase-host:60080	https://knox-host:8443/hbase
Hive	http://hive-host:10001/cliservice	https://knox-host:8443/hive
YARN	http://yarn-host:8088/ws	https://knox-host:8443/resourcemanager





Typical topology would look like this where you have 3 layers:
1. Presentation
2. Application
3. Data

Knox can run in DMZ in between firewalls so your cluster and AD remains hidden from outside world
With Knox you can optionally expose data from multiple clusters (without end users knowing where data came from)


Kerberos, Active Directory/LDAP and Apache Ranger
Apache Knox Gateway
Wire Encryption
Objectives
128
© Hortonworks Inc. 2011 – 2016. All Rights Reserved

Wire Encryption

Protects data in motion

- Hadoop clusters typically communicate via:
  - Encrypted Remote Procedure Call – RPC using Java's Simple Authentication & Security Layer (SASL)
  - Hyper Text Transfer Protocol Secure - HTTP Over SSL
  - Data Transfer Protocol – Encrypted DTP using 3DES
  - Java Database Connectivity – JDBC SASL Protocol's Quality of Protection QOP
  - MapReduce Shuffle – Utilizes HTTPS
- Performance overhead 2x degradation

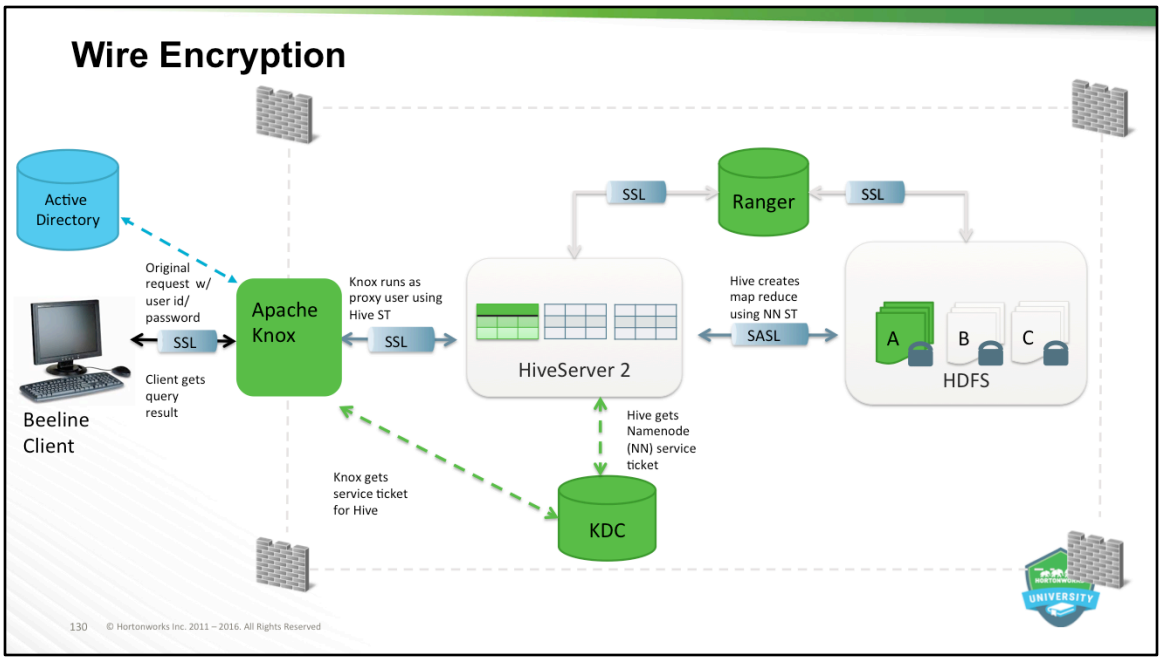
Encryption is applied to electronic information in order to ensure its privacy and confidentiality. Wire encryption protects data as it moves into and through Hadoop cluster over RPC, HTTP, Data Transfer Protocol (DTP), and JDBC.

The following describes how the data is protected as it is in motion:

Clients typically communicate directly with the Hadoop cluster and the data can be protected using:

RPC encryption: Clients interacting directly with the Hadoop cluster through RPC. A client uses RPC to connect to the NameNode (NN) to initiate file read and write operations. RPC connections in Hadoop use Java's Simple Authentication & Security Layer (SASL), which supports encryption.

Data Transfer Protocol: The NN gives the client the address of the first DataNode (DN) to read or write the block. The actual data transfer between the client and a DN


Wire Encryption
Active Directory
Original request w/ user id/ password
SSL
Client gets query result
Beeline Client
Apache Knox
Knox runs as proxy user using Hive ST
SSL
SSL
Ranger
SSL
HiveServer 2
Hive creates map reduce using NN ST
SASL
A
B
C
HDFS
Hive gets Namenode (NN) service ticket
Knox gets service ticket for Hive
KDC
130
© Hortonworks Inc. 2011 – 2016. All Rights Reserved

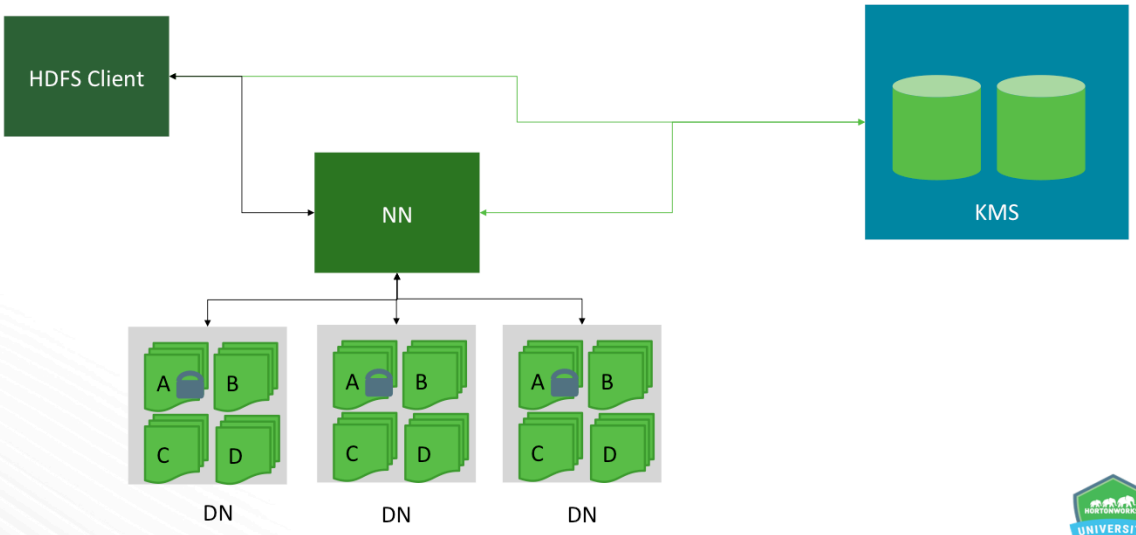

Kerberos, Active Directory/LDAP and Apache Ranger
Apache Knox Gateway
Wire Encryption
HDFS Data at Rest Encryption
Objectives
131
© Hortonworks Inc. 2011 – 2016. All Rights Reserved

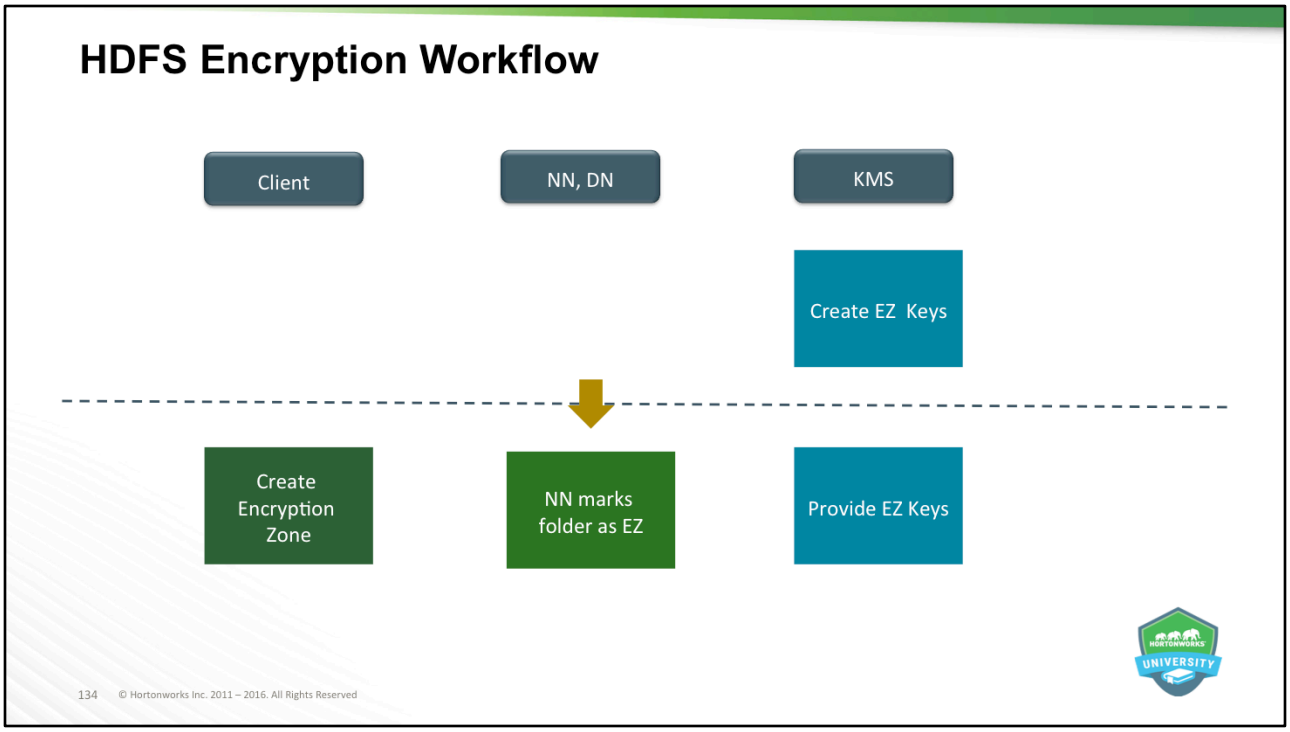
HDFS Encryption Solution

Protects data at rest

- Benefits:
  - No need to encrypt whole disk
  - Prevent rogue admin access to sensitive data
  - Different access control levels
  - Transparent to end application, little changes needed
- Performance overhead around 10-15%

HDFS Encryption Solution

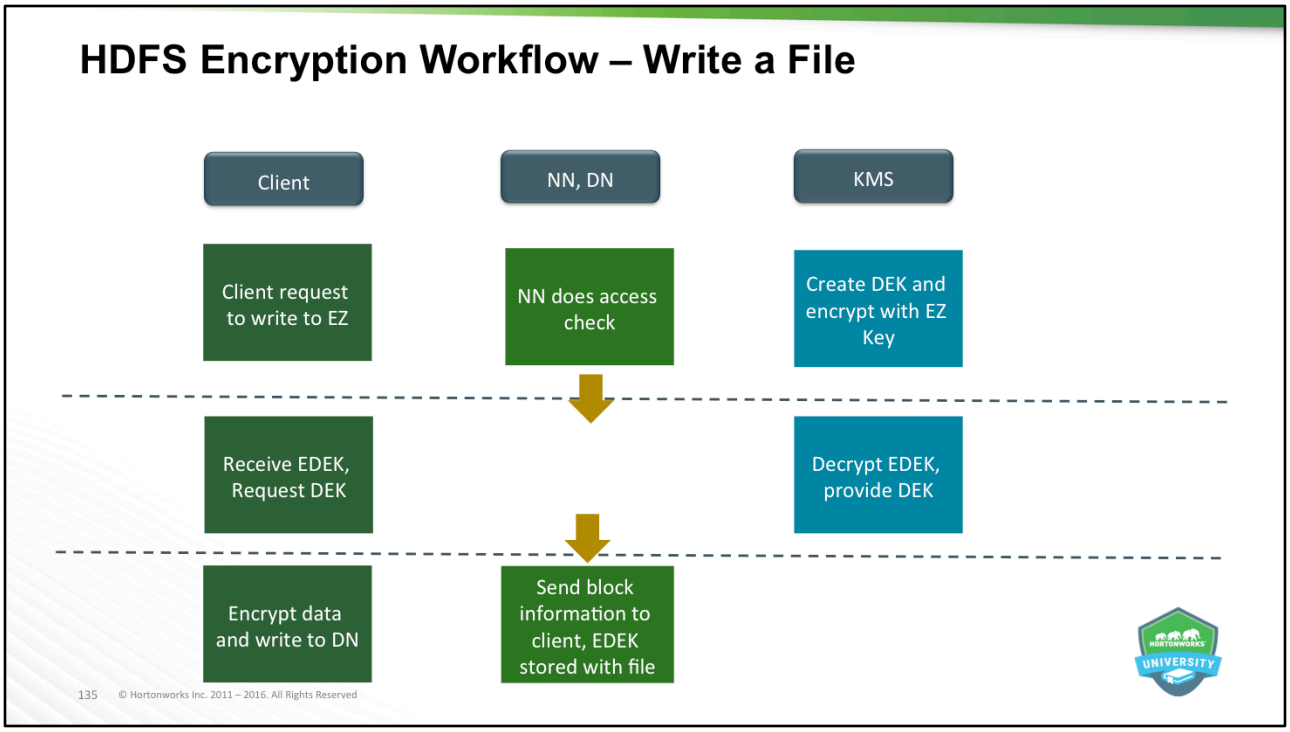




For transparent encryption, we introduce a new abstraction to HDFS: the encryption zone. An encryption zone is a special directory whose contents will be transparently encrypted upon write and transparently decrypted upon read. Each encryption zone is associated with a single encryption zone key which is specified when the zone is created.

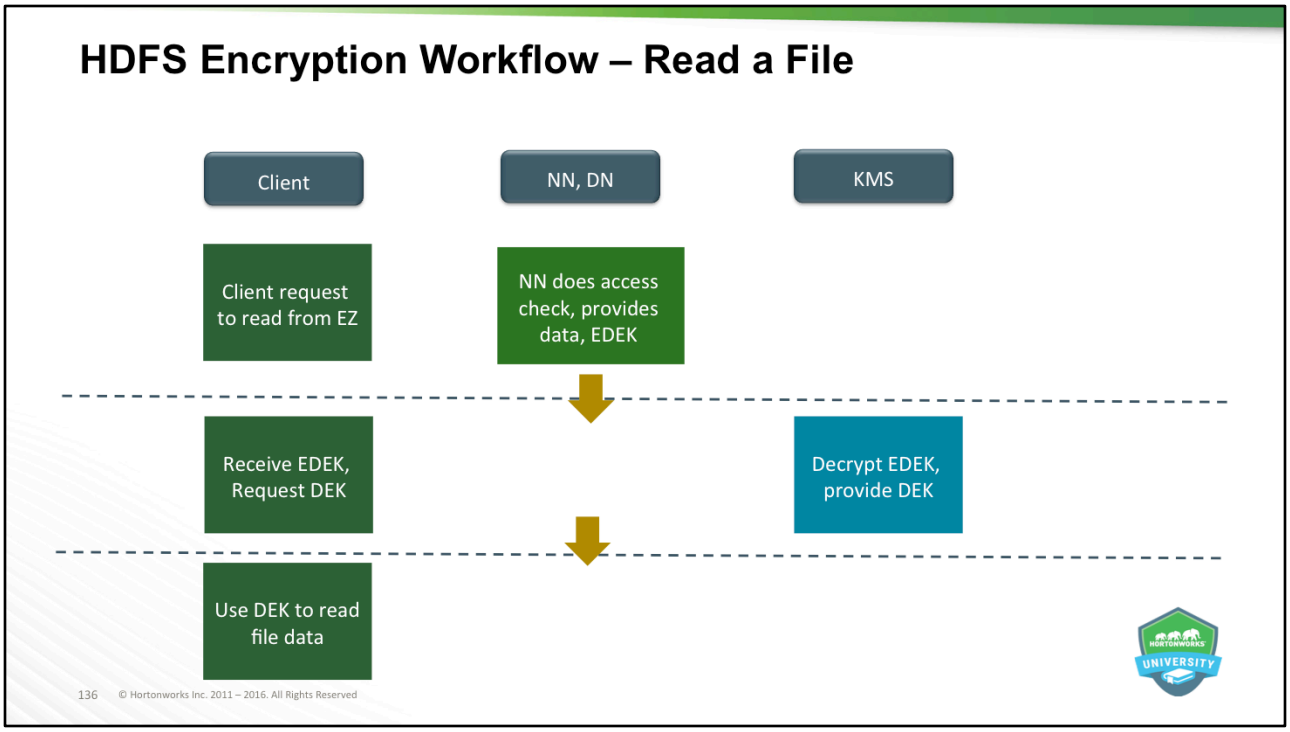
KMS performs three basic responsibilities:

- Providing access to stored encryption zone keys
- Generating new encrypted data encryption keys for storage on the NameNode
- Decrypting encrypted data encryption keys for use by HDFS clients



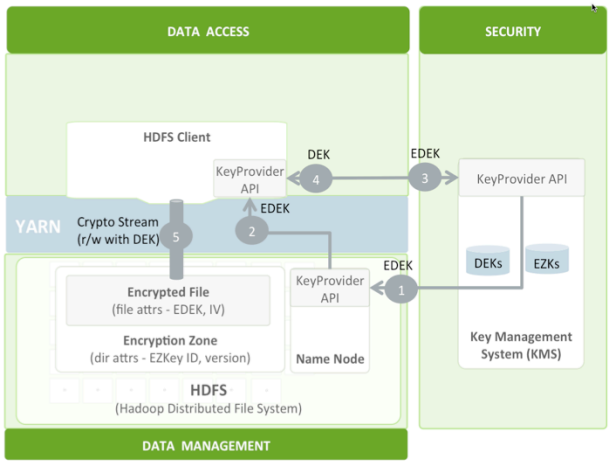
Each file within an encryption zone has its own unique data encryption key (DEK). DEKs are never handled directly by HDFS. Instead, HDFS only ever handles an encrypted data encryption key (EDEK). Clients decrypt an EDEK, and then use the subsequent DEK to read and write data. HDFS DataNodes simply see a stream of encrypted bytes.

When creating a new file in an encryption zone, the NameNode asks the KMS to generate a new EDEK encrypted with the encryption zone's key. The EDEK is then stored persistently as part of the file's metadata on the NameNode.



When reading a file within an encryption zone, the NameNode provides the client with the file's EDEK and the encryption zone key version used to encrypt the EDEK. The client then asks the KMS to decrypt the EDEK, which involves checking that the client has permission to access the encryption zone key version. Assuming that is successful, the client uses the DEK to decrypt the file's contents.

Apache Ranger KMS



Acronym	Description
EZ	Encryption Zone (an HDFS directory)
EZK	Encryption Zone Key; master key associated with all files in an EZ
DEK	Data Encryption Key, unique key associated with each file. EZ Key used to generate DEK
EDEK	Encrypted DEK, Name Node only has access to encrypted DEK.
IV	Initialization Vector

Process for encryption in HDFS

1. KMS admin creates a Encryption Zone (EZ) Key (hadoop key create mykey)

2. HDFS admin creates an Encryption Zone using the EZKey (hdfs crypto -createZone -keyName myKey -path /home/me/zone1). Assuming that /home/me/zone1 already exists.

3. When user creates a file in the EZ with appropriate access permissions,

- Name Node uses EZ Key ID and version to get a EDEK (encrypted data encryption key) from KMS
- Name Node returns EDEK to HDFS Client
- HDFS client communicates with KMS to decrypt EDEK
- HDFS client uses DEK and Hadoop Cryptographic File System (CryptoOutputStream) to write an encrypted file in the HDFS EZ

4. When user reads a file in the EZ, check for appropriate permissions and KMS authorization

- Name Node passes EDEK to KMS

Knowledge Check

Knowledge Check

1. What is Kerberos used for?
2. True/False - Every Hadoop cluster requires Kerberos as part of security.
3. What HDP component provides consistent security policy across the Hadoop stack?
4. True/False - Apache Ranger utilizes additional daemon processes to interface with the Hadoop stack.
5. What HDP component hides the Hadoop cluster's internal ports and hosts from end users and provide a single SSL based URL?
6. True/False - Wire Encryption protects data at rest.
7. What protection does HDFS Encryption provide?

1. Authentication and Authorization
2. TRUE
3. Apache Ranger
4. FALSE - Ranger uses plugins
5. Apache Knox
6. FALSE
7. Protects Data at Rest

Summary

Summary

- Every Hadoop Cluster needs Kerberos for Authentication and Authorization
- User Authentication is accomplished by leveraging existing Active Directory or LDAP infrastructure
- Apache Ranger provides central administration ensuring consistent security policy across the Hadoop Stack utilizing Ranger Plugins.
- Apache Knox will streamline authentication, hide the cluster's internal ports and the complexities of Kerberos from the end points accessing data.
- Apache Knox provides a single SSL based URL to access the Hadoop services it supports.
- Wire Encryption protects the data in motion.
- HDFS Encryption protects the data at rest.

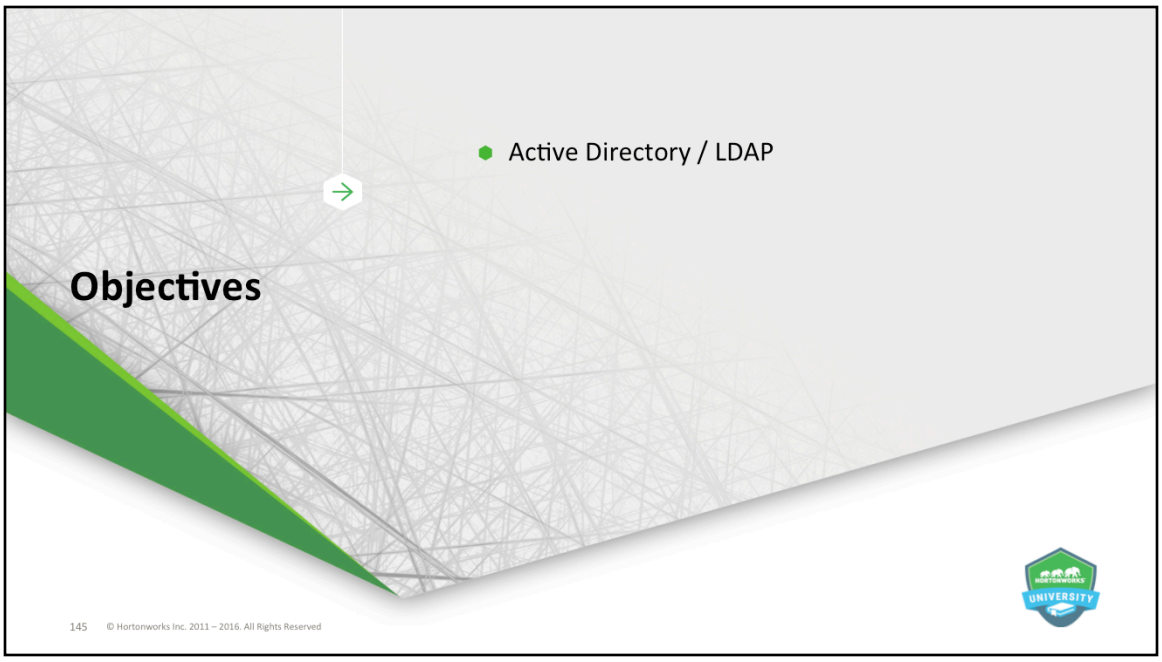
Lab: Accessing Your Cluster/Setting up the Environment

This lesson covers the basics of security in an HDP environment.

Objectives

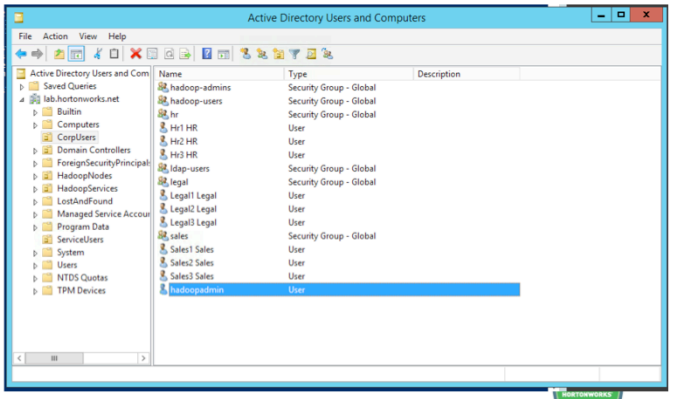
After completing this lesson, students should be able to:

- Understand authentication options – *Active Directory or LDAP*
- Discuss KDC implementation options – *MIT KDC or Active Directory*
- Install an HDP cluster managed by Ambari – *fully configured & tested*

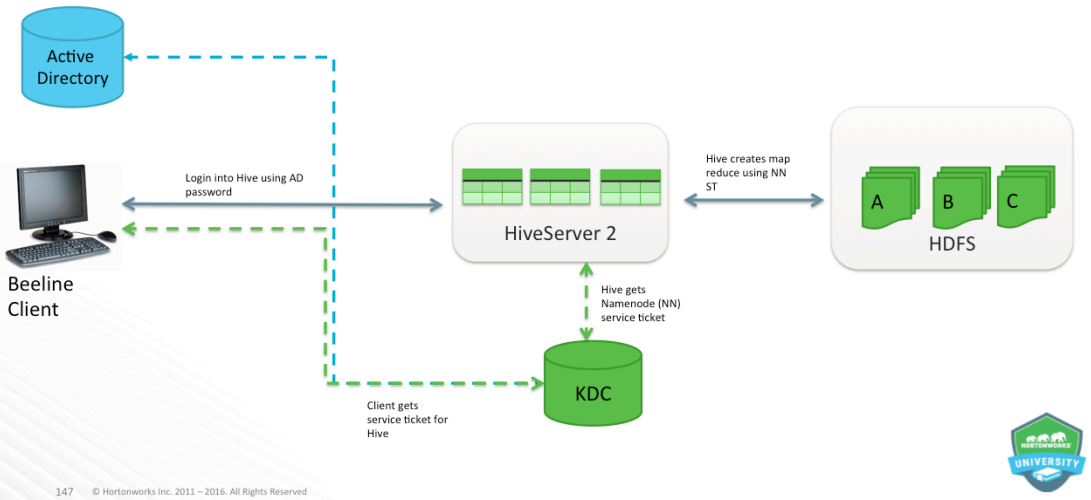

Active Directory / LDAP
Objectives
145
© Hortonworks Inc. 2011 – 2016. All Rights Reserved

Active Directory/LDAP

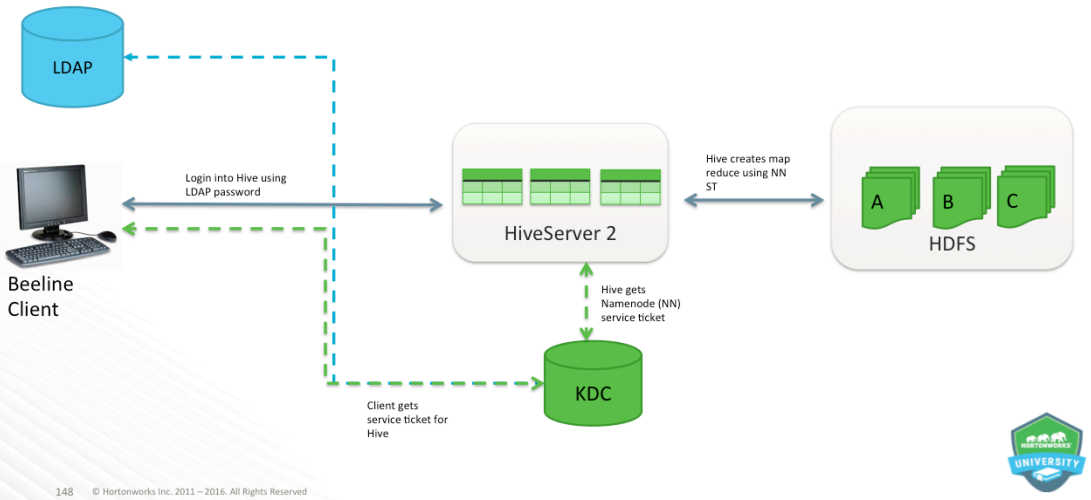
- Most Environments Have One
- Leverage Existing Infrastructure
- Used to Authenticate Users
- Provide Single Sign On
- Utilize Existing Tools to Manage

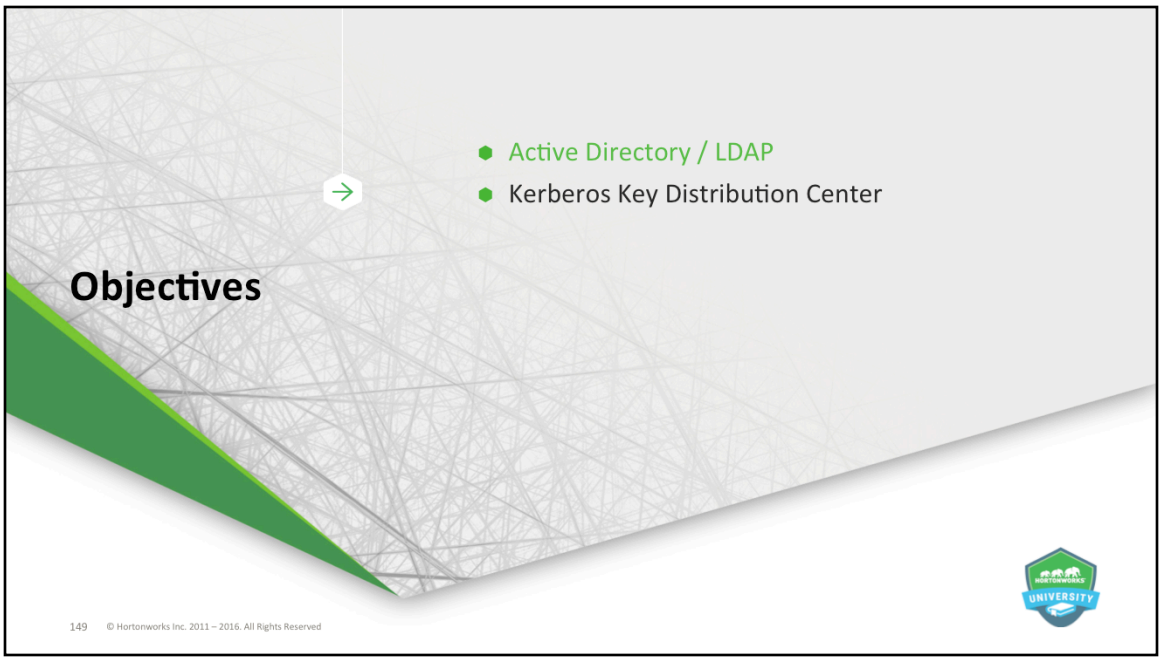


Authenticate through Kerberos/AD



Authenticate through Kerberos/LDAP




Objectives
Active Directory / LDAP
Kerberos Key Distribution Center
149
© Hortonworks Inc. 2011 – 2016. All Rights Reserved

Kerberos KDC

- MIT Kerberos Server
  - The Original Key Distribution Center
  - Open Source and Packaged with Linux
  - Dedicated to HDP Cluster
- Microsoft Active Directory
  - Infrastructure Usually Exists
  - Widely Used

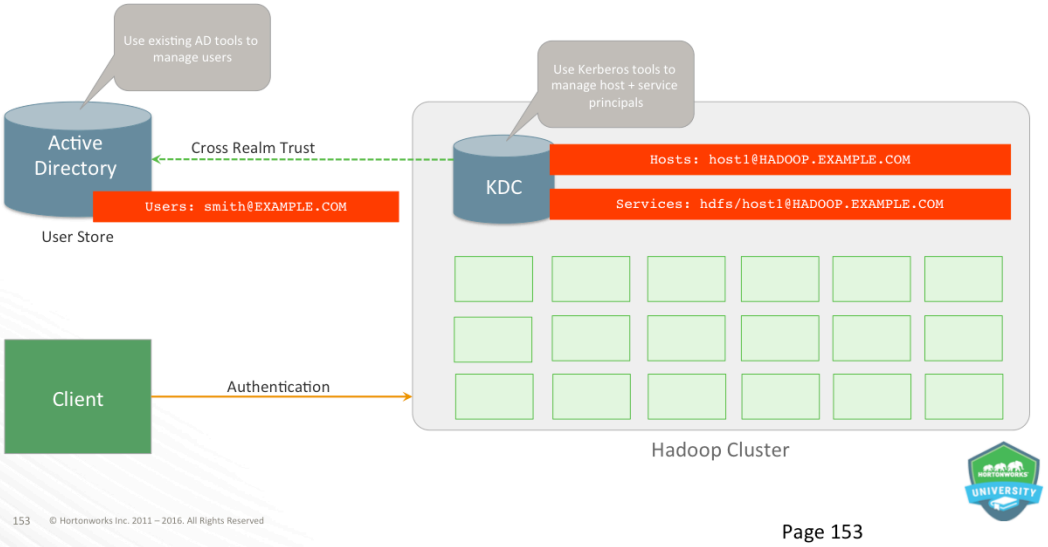
Two Implementation Options

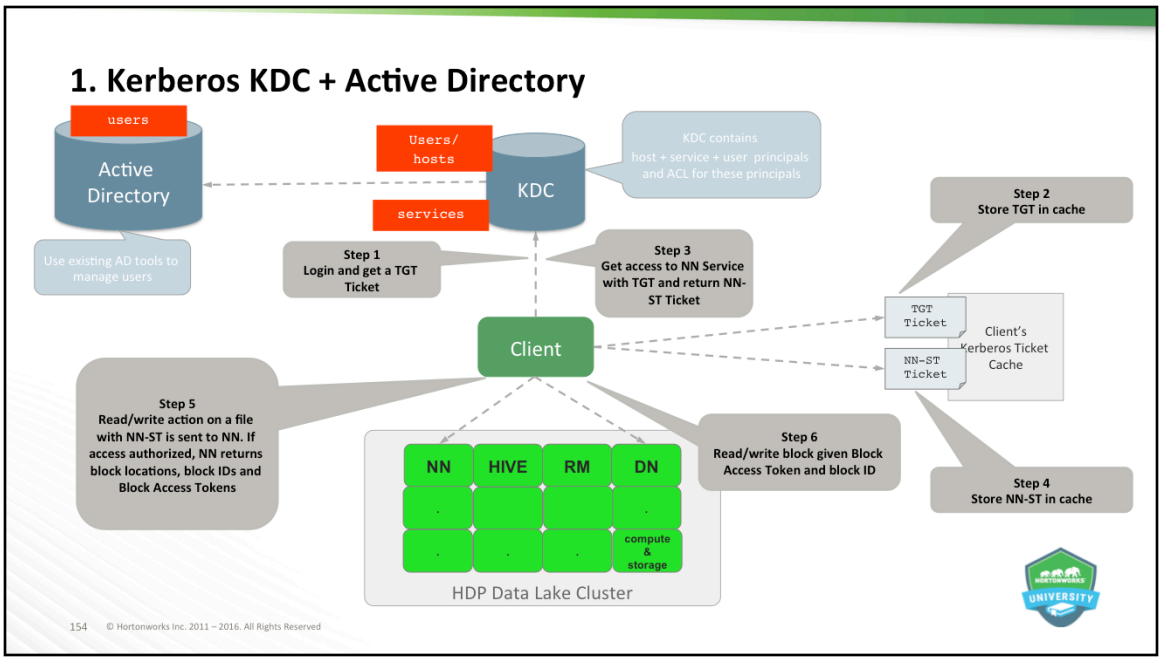
- 1. MIT Kerberos KDC Server with Established Trust to AD
  - KDC is Dedicated to HDP Cluster
  - One-Way Cross Realm Trust From Kerberos KDC realm to AD domain
  - User Authentication in Active Directory
  - Service Principals in Local KDC
- 2. Utilize Existing Microsoft Active Directory Server as KDC Server - Recommended
  - Obtain Dedicated AD Replica
    - HDP Requires Many Service Principal Names – SPN
    - Cluster Can Overload Domain Controller at ~100 Nodes
  - Need AD Administrator or Privilege
  - Dedicated OU for Hadoop principals

1. MIT KDC Implementation

- Install MIT Kerberos KDC Primary and Secondary Server
- Install Kerberos Client Package on All HDP Cluster Nodes
- Configure Kerberos KDC Servers/Clients
- Establish One-way Trust to Active Directory
- Configure Kerberos Hadoop Realm on the AD Domain Controller
- Configure AD Domain on the KDC and HDP Cluster Nodes

1. Kerberos KDC + Active Directory

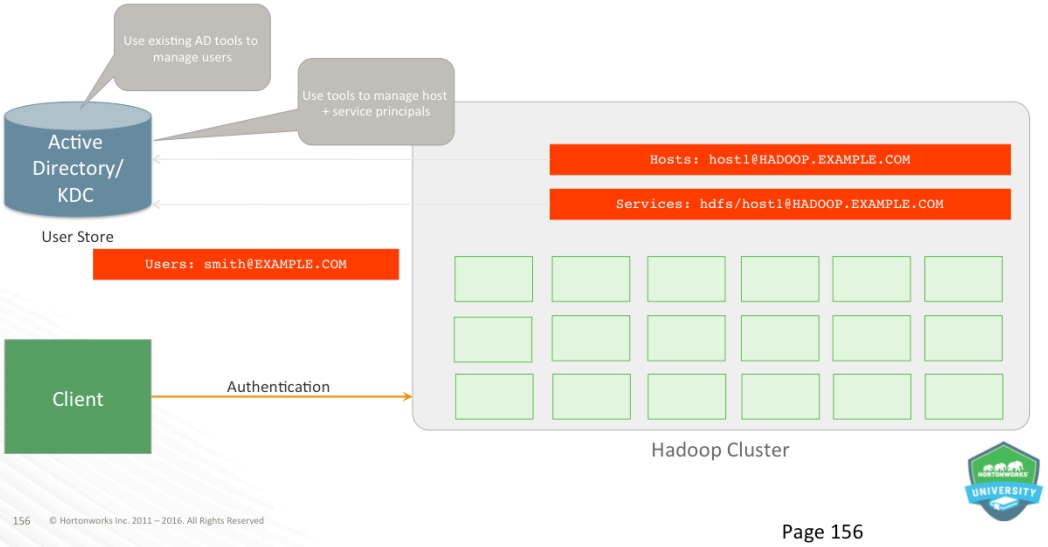



1. Kerberos KDC + Active Directory
users
Active Directory
Use existing AD tools to manage users
Users/ hosts
services
KDC
KDC contains host + service + user principals and ACL for these principals
Step 1 Login and get a TGT Ticket
Step 2 Store TGT in cache
Step 3 Get access to NN Service with TGT and return NN-ST Ticket
Client
TGT Ticket
NN-ST Ticket
Client's Kerberos Ticket Cache
Step 4 Store NN-ST in cache
Step 5 Read/write action on a file with NN-ST is sent to NN. If access authorized, NN returns block locations, block IDs and Block Access Tokens
Step 6 Read/write block given Block Access Token and block ID
NN
HIVE
RM
DN
compute & storage
HDP Data Lake Cluster
154
© Hortonworks Inc. 2011 – 2016. All Rights Reserved

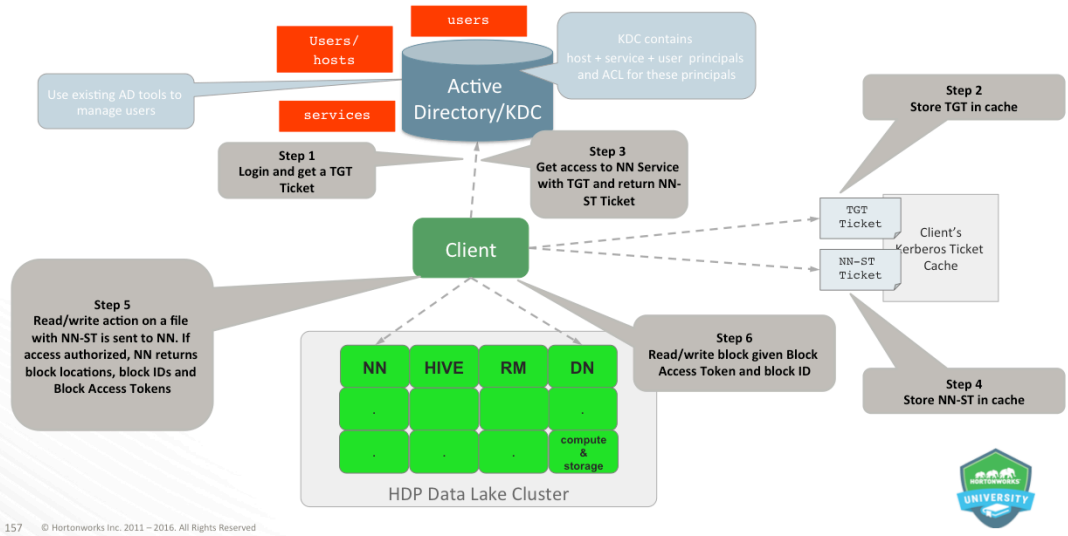
2. Existing Active Directory Implementation

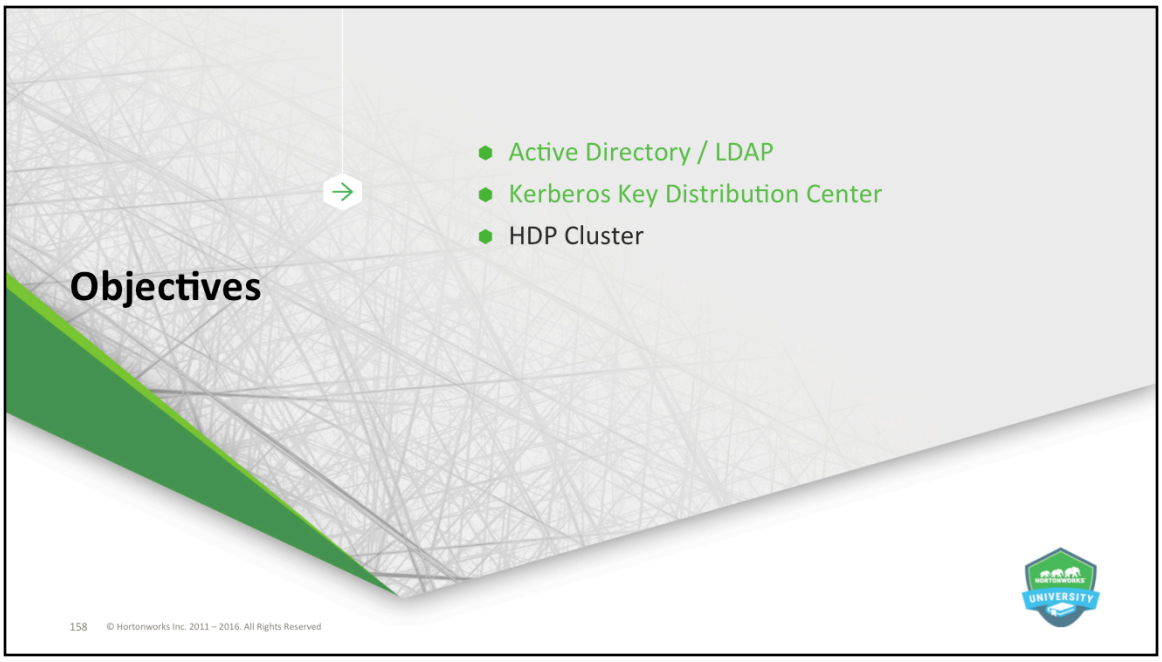
- Obtain Active Directory Replica Server
- Basic structure of Organizational Units have been pre-created
- Ambari Server and Hadoop Cluster Nodes have
  - Network Access to Domain Controllers
  - Able to Resolve the DNS names of Domain Controllers
- AD Secure LDAP (LDAPS) connectivity has been configured
- AD User Container for principals has been created and Available
- AD administrative credentials with delegated control of "Create, Delete, and Manage User Accounts" on the previously mentioned User Container are Available

2. Existing Active Directory Implementation



2. Existing Active Directory Implementation

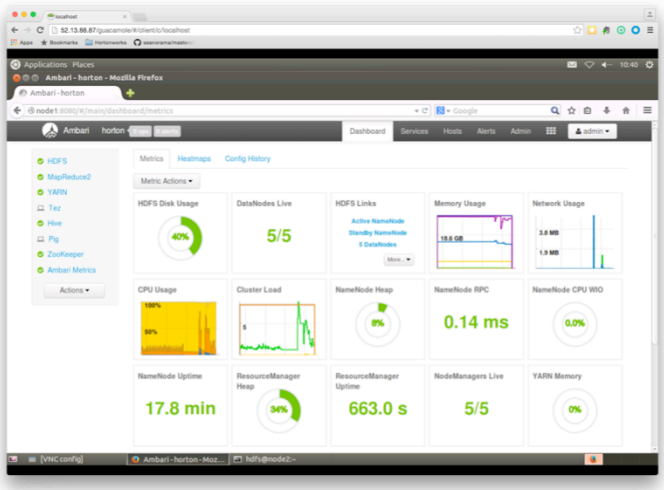



Objectives
Active Directory / LDAP
Kerberos Key Distribution Center
HDP Cluster
158
© Hortonworks Inc. 2011 – 2016. All Rights Reserved

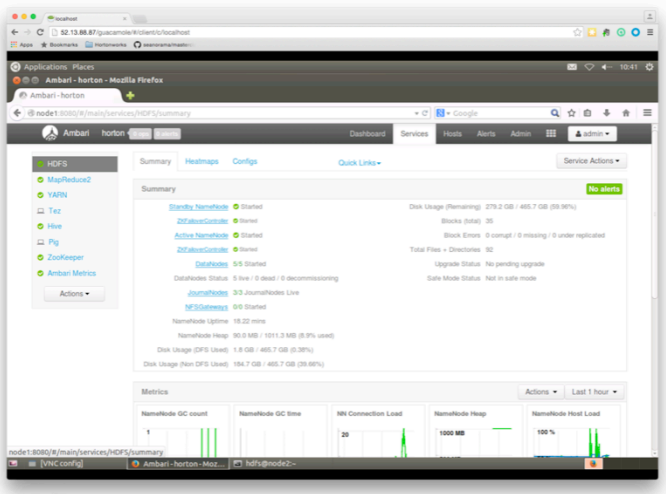
HDP Cluster

- Install HDP Cluster Utilizing Ambari
- Configure HDP Cluster and All Desired EcoSystems Components
- Test HDP Cluster and All Installed EcoSystem Components
- Ensure HDP Cluster Connectivity to Kerberos KDC Server
- Configure Ambari for LDAP Sync Utilizing AD
- Setup AD/OS Integration Via SSSD

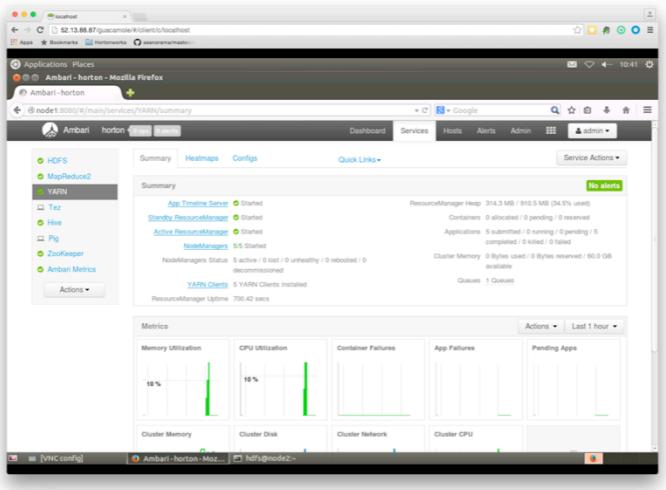
Installed HDP Cluster



HDP Cluster – HA NameNode



HDP Cluster – HA ResourceManager



HDP Cluster - OS/AD Integration

- What does OS/AD integration mean?
  - $ kinit
  - $ groups sales1
  - *sales1 : domain_users sales hadoop-users*
- What is it needed for:
  - Group mappings
  - Ability to submit YARN jobs
  - Easier access/administration
- How?
  - Manually add users/groups (don't do this!)
  - PAM, winbind, NSLCD, …
  - SSSD with AD/Kerberos (commonly used)

SSSD is a system daemon. Its primary function is to provide access to identity and authentication remote resource through a common framework that can provide caching and offline support to the system. It provides PAM and NSS modules, and in the future will D-BUS based interfaces for extended user information. It provides also a better database to store local users as well as extended user data.

HDP Cluster - SSSD OS/AD Integration

- SSSD is a System Security Services Daemon
  - Primary Function to Provide Access to Identity and Authentication Resource
  - Utilizes a common framework that provides caching and offline support to the system
  - Provides Pluggable Authentication Module - PAM Modules
  - Provides Network Security Services - NSS Modules
- Needed to Allow Hadoop Nodes to Recognize Users/Group Defined in AD
- Each Hadoop Node Needs to Join AD Domain
- Configure SSSD "Identity & Authentication" on edge nodes
  - Allows users to login and automatically get kerberos ticket
- Configure SSSD "Identity Provider" on master/data nodes
  - To prevent users from logging in to these nodes

SSSD is a system daemon. Its primary function is to provide access to identity and authentication remote resource through a common framework that can provide caching and offline support to the system. It provides PAM and NSS modules, and in the future will D-BUS based interfaces for extended user information. It provides also a better database to store local users as well as extended user data.


Summary
HORTONWORKS
UNIVERSITY

Summary

- Most environments implementing a Hadoop cluster have an existing Active Directory or LDAP infrastructure to authenticate users
- Active Directory can be used for your Kerberos KDC
- Utilizing MIT Kerberos KDC allows for dedication to the Hadoop cluster
- Recommendation is to utilize an Active Directory Replica Domain Server for the KDC
- A fully configured and tested Hadoop cluster is required prior to enabling security

This summary page lists some of the main points from this lesson.

Lab: Configure AD Resolution and Certificate


Ambari Server Security
HORTONWORKS
UNIVERSITY

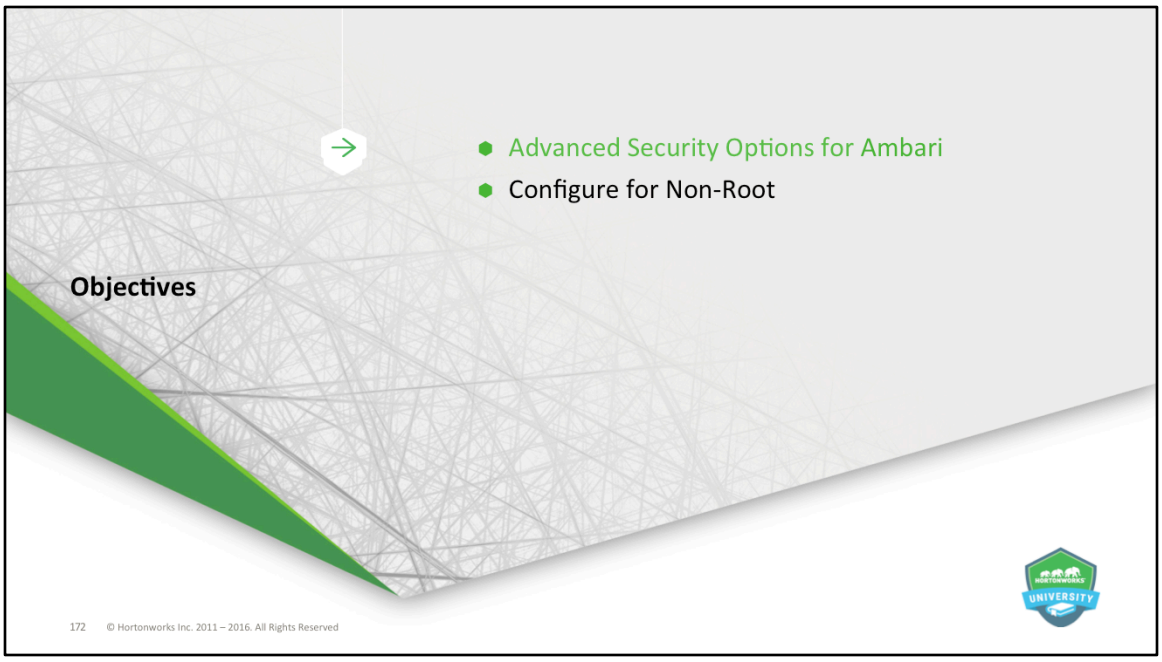
Lesson Objectives

- Configure Ambari Server for Security
- Configure Ambari Server/Agent for Non-Root
- Encrypt the Ambari Database and Passwords
- Configure Ambari for LDAP or Active Directory Authentication
- Setup HTTPS/SSL Server
- Setup Two-Way SSL Between Ambari Server and Agents
- Enable SPNEGO Authentication for Hadoop


Advanced Security Options for Ambari
Objectives
170
© Hortonworks Inc. 2011 – 2016. All Rights Reserved

Advanced Security Options for Ambari

- Configure Ambari Server for Non-Root
- Configure Ambari Agent for Non-Root
- Encrypt Database and Passwords
- Enable LDAP/Active Directory Authentication
- Enable HTTPS/SSL Server
- Enable Two-Way SSL
- Enable SPNEGO Authentication for Hadoop


Advanced Security Options for Ambari
Configure for Non-Root
Objectives
172
© Hortonworks Inc. 2011 – 2016. All Rights Reserved

Configure Ambari Server for Non-Root

- Secure Environments Require:
  - Restricting Access to Root
  - Limiting Services that Run as Root
- Server can be Configured to Operate Without Direct Root Access

- During "ambari-server setup" Process
- Prompt Provided to "Customize user account for ambari-server daemon?": Choose "y"
- Setup Prompts for Appropriate Non-Root User for Ambari Server
- Choose Non-Root User to Run Ambari-Server – "ambari"
- User Must Be Part of Hadoop Group – Default Group is "hadoop"

Configure Ambari Agent for Non-Root

- Agent can be Configured to Operate Without Direct Root Access
- User Requires sudo Access to "su" to:
  - Hadoop Service Accounts
  - Perform Specific Privileged Commands
  - Follow Sudoer Configuration
- Edit "run_as_user" Property in /etc/ambari-agent/conf/ambari-agent.ini File
  run_as_user=ambari
- Restart Ambari Agent
- Must Be Done on Every Node

Sudoer Configuration

- Sudo Needs to be Configured to Enable Ambari to Run as Non-Root
- Customizable Users Section Contains "su" Commands & Corresponding Service Accounts

```
# Ambari Customizable Users
ambari ALL=(ALL) NOPASSWD:SETENV: /bin/su hdfs *,/bin/su ambari-qa *,/bin/su ranger *,/bin/su zookeeper *,/bin/su knox *,/
bin/su falcon *,/bin/su ams *, /bin/su flume *,/bin/su hbase *,/bin/su spark *,/bin/su accumulo *,/bin/su hive *,/bin/su hcat *,/
bin/su kafka *,/bin/su mapred *,/bin/su oozie *,/bin/su sqoop *,/bin/su storm *,/bin/su tez *,/bin/su atlas *,/bin/su yarn *,/bin/su
kms *
```

- Non-Customizable User Section Contains "su" Commands for System Account
  - Cannot be Modified
  - Only Required if Using Ambari Installed/Managed MySQL Instance for Hive Metastore

```
# Ambari Non-Customizable Users
ambari ALL=(ALL) NOPASSWD:SETENV: /bin/su mysql *
```

Sudoer Configuration

- Commands Section Contains Specific Commands Issued for Standard Agent Operations

  # Ambari Commands
  ambari ALL=(ALL) NOPASSWD:SETENV: /usr/bin/yum,/usr/bin/zypper,/usr/bin/apt-get, /bin/mkdir, /usr/bin/test, /bin/ln, /bin/chown, /bin/chmod, /bin/chgrp, /usr/sbin/groupadd, /usr/sbin/groupmod, /usr/sbin/useradd, /usr/sbin/usermod, /bin/cp, /usr/sbin/setenforce, /usr/bin/test, /usr/bin/stat, /bin/mv, /bin/sed, /bin/rm, /bin/kill, /bin/readlink, /usr/bin/pgrep, /bin/cat, /usr/bin/unzip, /bin/tar, /usr/bin/tee, /bin/touch, … ***Trimmed See Document for Complete Command***

  # Ambari Ranger Commands
  ambari ALL=(ALL) NOPASSWD:SETENV: /usr/hdp/*/ranger-usersync/setup.sh, /usr/bin/ranger-usersync-stop, /usr/bin/ranger-usersync-start, /usr/hdp/*/ranger-admin/setup.sh *, /usr/hdp/*/ranger-knox-plugin/disable-knox-plugin.sh *, /usr/hdp/*/ranger-storm-plugin/disable-storm-plugin.sh *, /usr/hdp/*/ranger-hbase-plugin/disable-hbase-plugin.sh *, /usr/hdp/*/ranger-hdfs-plugin/disable-hdfs-plugin.sh *, /usr/hdp/current/ranger-admin/ranger_credential_helper.py, /usr/hdp/current/ranger-kms/ranger_credential_helper.py

- Sudo Defaults Section Contains Commands to Override Non-Interactive Shell
  - Agents needs to Run Commands Non-Interactively

    Defaults exempt_group = ambari
    Defaults !env_reset,env_delete-=PATH
    Defaults: ambari !requiretty

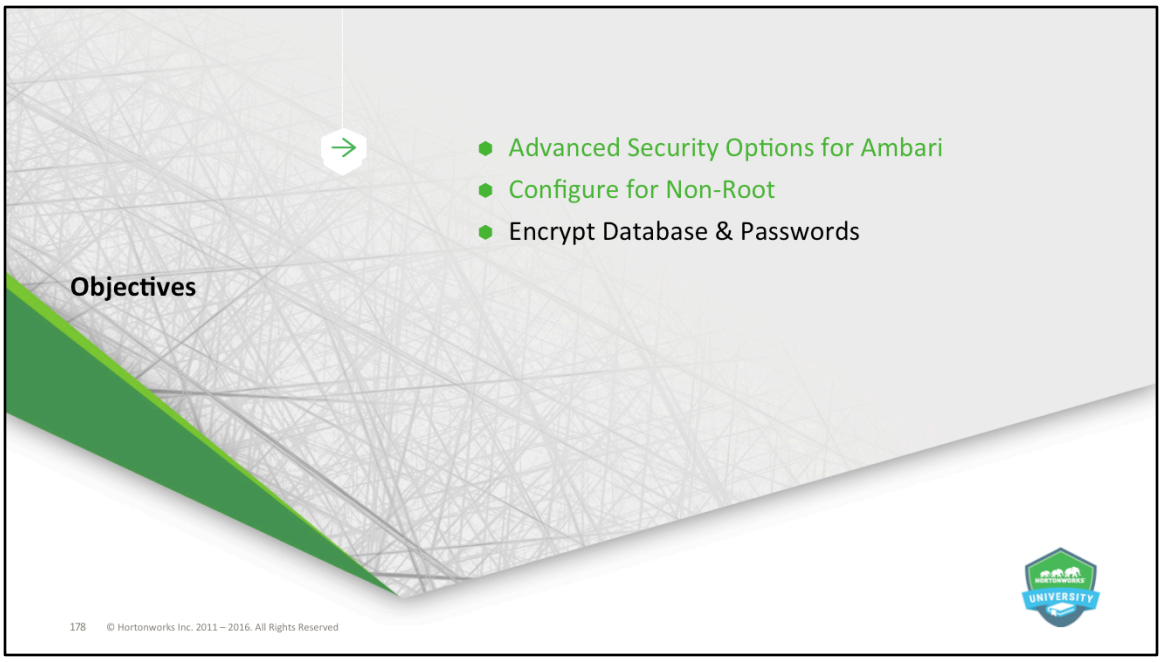
Ambari Commands

ambari ALL=(ALL) NOPASSWD:SETENV: /usr/bin/yum,/usr/bin/zypper,/usr/bin/apt-get, /bin/mkdir, /usr/bin/test, /bin/ln, /bin/chown, /bin/chmod, /bin/chgrp, /usr/sbin/groupadd, /usr/sbin/groupmod, /usr/sbin/useradd, /usr/sbin/usermod, /bin/cp, /usr/sbin/setenforce, /usr/bin/test, /usr/bin/stat, /bin/mv, /bin/sed, /bin/rm, /bin/kill, /bin/readlink, /usr/bin/pgrep, /bin/cat, /usr/bin/unzip, /bin/tar, /usr/bin/tee, /bin/touch, /usr/bin/hdp-select, /usr/bin/conf-select, /usr/hdp/current/hadoop-client/sbin/hadoop-daemon.sh, /usr/lib/hadoop/bin/hadoop-daemon.sh, /usr/lib/hadoop/sbin/hadoop-daemon.sh, /sbin/chkconfig gmond off, /sbin/chkconfig gmetad off, /etc/init.d/httpd *, /sbin/service hdp-gmetad start, /sbin/service hdp-gmond start, /usr/sbin/gmond, /usr/sbin/update-rc.d ganglia-monitor *, /usr/sbin/update-rc.d gmetad *, /etc/init.d/apache2 *, /usr/sbin/service hdp-gmond *, /usr/sbin/service hdp-gmetad *, /sbin/service mysqld *, /usr/bin/python2.6 /var/lib/ambari-agent/data/tmp/validateKnoxStatus.py *, /usr/hdp/current/knox-server/bin/knoxcli.sh *

Ambari Ranger Commands

Sudoer Configuration

- Configuration Must Be Done on Every Node
- Verify Configuration “su ambari” Execute Command “sudo –l”
- Configuration Output Matches Entries Applied

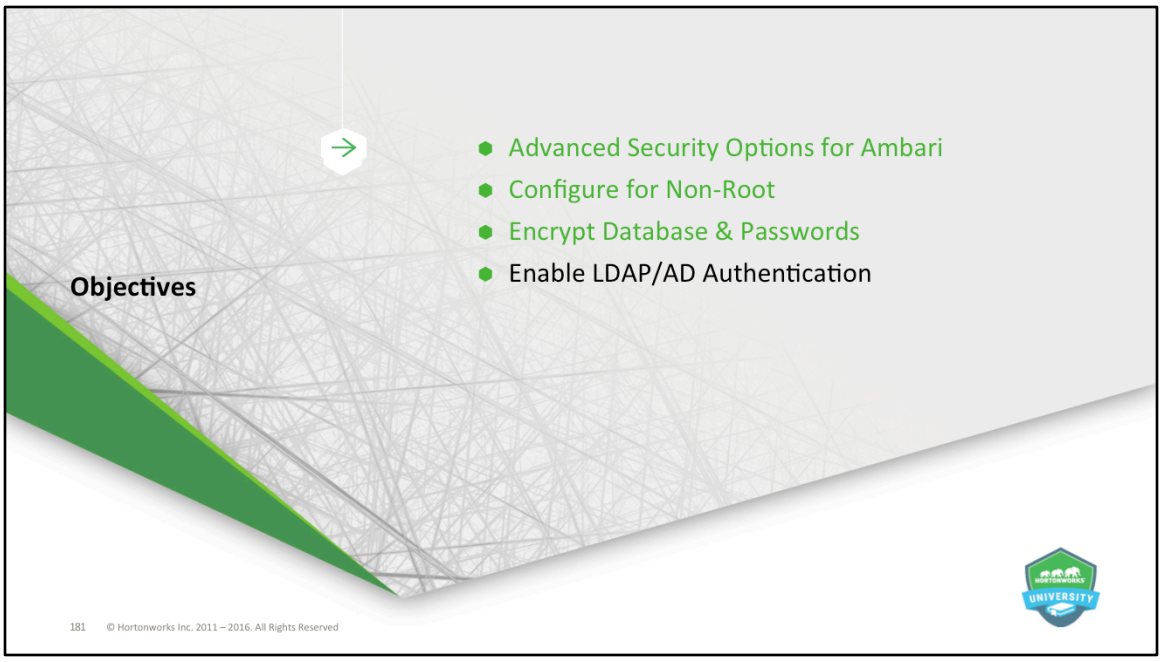

Objectives
Advanced Security Options for Ambari
Configure for Non-Root
Encrypt Database & Passwords
178
© Hortonworks Inc. 2011 – 2016. All Rights Reserved

Encrypt Database and LDAP Passwords

- By Default Passwords to Access Ambari Database/LDAP Server Stored Plain Text
- To Encrypt Passwords in Configuration File – Run Special Setup Command
- On Ambari Server
  - Stop Ambari Server
  - Run "ambari-server setup-security"
  - Select Option 2 - "[2] Encrypt passwords stored in ambari.properties file."
  - Provide Master Key for Encrypting Passwords and Confirm
  - Three Options for Maintaining Master Key
    - Persist it to a file on the server by pressing "y" at the prompt
    - Create an environment variable AMBARI_SECURITY_MASTER_KEY and set it to the key
    - Provide the key manually at the prompt on server start up
  - Start Ambari Server

Encrypt Database and LDAP Passwords

- To Change the Current Master Key
- On Ambari Server
  - Stop Ambari Server
  - Run “ambari-server setup-security”
  - Select Option 2 - “[2] Encrypt passwords stored in ambari.properties file.”
  - Provide Master Key When Prompted
  - “Do you want to reset Master Key?” – Enter “yes”
  - Provide New Master Key for Encrypting Passwords and Confirm
  - Start Ambari Server


Objectives
Advanced Security Options for Ambari
Configure for Non-Root
Encrypt Database & Passwords
Enable LDAP/AD Authentication
181
© Hortonworks Inc. 2011 – 2016. All Rights Reserved

Configure Ambari for LDAP/AD Authentication

- By Default Ambari Uses Internal Database as User Store for Authentication/Authorization
- Configure External Authentication with LDAP/Active Directory (AD)
- Must Synchronize LDAP Users/Groups into Ambari DB
- Then Manage Authorization and Permission Against Users/Groups
- LDAP Properties/Values Needed for LDAP Authentication:

NOTE: When synchronizing LDAP users and groups, Ambari uses LDAP results paging controls to synchronize large numbers of LDAP objects. Most modern LDAP servers support these control, but for those that do not, such as Oracle Directory Server Enterprise Edition 11g, Ambari introduces a configuration parameter to disable pagination. The authentication.ldap.pagination.enabled property can be set to false in the /etc/ambari-server/conf/ambari-properties file to disable result paging controls. This will limit the maximum number of entities that can be imported at any given time to the maximum result limit of the LDAP server. To work around this, import sets of users or groups using the -users and -groups options covered in section 3.1.4 - Specific Set of Users and Groups.

Configure Ambari for LDAP/AD Authentication

Property	Values	Description
authentication.ldap.primaryUrl	Server:Port	Hostname/Port for LDAP/AD Server
authentication.ldap.secondaryUrl	Server:Port	Hostname/Port for LDAP/AD Server – Secondary
authentication.ldap.useSSL	true or false	If true, use SSL when connecting to the LDAP or AD server.
authentication.ldap.usernameAttribute	[LDAP attribute]	The attribute for username. Example: uid
authentication.ldap.baseDn	[Distinguished Name]	The root Distinguished Name to search in the directory for users. Example: ou=people,dc=hadoop,dc=apache,dc=org
authentication.ldap.referral	[Referral method]	Determines if LDAP referrals should be followed, or ignored.
authentication.ldap.bindAnonymously	true or false	If true, bind to the LDAP or AD server anonymously
authentication.ldap.managerDn	[Full Distinguished Name]	If Bind anonymous is set to false, the Distinguished Name ("DN") for the manager. Example: uid=hdfs,ou=people,dc=hadoop,dc=apache,dc=org
authentication.ldap.managerPassword	[password]	If Bind anonymous is set to false, the password for the manager
authentication.ldap.userObjectClass	[LDAP Object Class]	The object class that is used for users. Example: organizationalPerson
authentication.ldap.groupObjectClass	[LDAP Object Class]	The object class that is used for groups. Example: groupOfUniqueNames
authentication.ldap.groupMembershipAttr	[LDAP attribute]	The attribute for group membership. Example: uniqueMember
authentication.ldap.groupNamingAttr	[LDAP attribute]	The attribute for group name.

NOTE: If you are going to set bindAnonymously to false (the default), you need to make sure you have an LDAP Manager name and password set up. If you are going to use SSL, you need to make sure you have already set up your certificate and keys.

Configure Ambari for LDAP/AD Authentication

- Configure Ambari to Use LDAP Server
- If Using LDAPS and Server is Self-Signed or Internal Certificate Authority
- Then Import Certificate and Create Keystore file As Follows:
  - Make Keys Directory - /etc/ambari-server/keys
  - Import Certificates Using Keytool

    $ JAVA_HOME/bin/keytool -import -trustcacerts -alias root -file $PATH_TO_YOUR_LDAPS_CERT -keystore /etc/ambari-server/keys/ldaps-keystore.jks
  - Set Password when Prompted

Note: Only if you are using LDAPS, and the LDAPS server certificate is signed by a trusted Certificate Authority, there is no need to import the certificate into Ambari so this section does not apply to you. If the LDAPS server certificate is self-signed, or is signed by an unrecognized certificate authority such as an internal certificate authority, you must import the certificate and create a keystore file. The following example creates a keystore file at /keys/ldaps-keystore.jks, but you can create it anywhere in the file system:

Run the LDAP setup command on the Ambari server and answer the prompts, using the information you collected above:

Configure Ambari for LDAP/AD Authentication

- Configure Ambari to Use LDAP Server
- Execute the LDAP Setup Command on Ambari Server
- Answer Prompts with Properties Information Collected – 15 Values

  ambari-server setup-ldap
- If "SSL*=true" In Step 3 a Prompt Appears

  "Do you want to provide custom TrustStore for Ambari?"

  – More Secure Option - Enter "y" Follow Addition Prompts

  – Less Secure Option - Enter "n"
- Review Settings and Confirm "y"
- Restart Ambari Server

If you set Use SSL* = true in step 3, the following prompt appears: Do you want to provide custom TrustStore for Ambari?

Consider the following options and respond as appropriate.

More secure option: If using a self-signed certificate that you do not want imported to the existing JDK keystore, enter y.

For example, you want this certificate used only by Ambari, not by any other applications run by JDK on the same host.

If you choose this option, additional prompts appear. Respond to the additional prompts as follows:

At the TrustStore type prompt, enter jks.

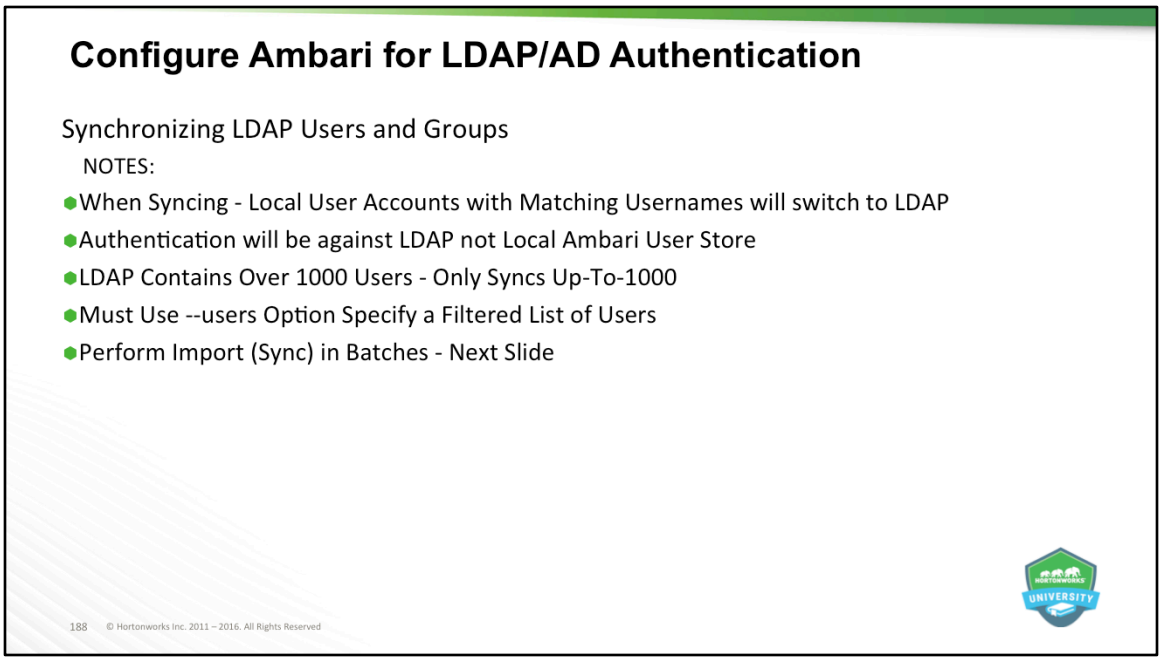
Configure Ambari for LDAP/AD Authentication

- Configure Ambari to Use LDAP Server
- The Users Imported are Initially Granted Ambari User Privilege
- Ambari Users can Read Metrics, View Service Status/Configuration, Browse Job Information
- These New Users Need to be Admins to Start/Stop Services, Modify Configurations, and Run Smoke Tests

Configure Ambari for LDAP/AD Authentication

Synchronizing LDAP Users and Groups

- Run LDAP Synchronize Command
  ambari-server sync-ldap
- Ambari Server Up and Running
- Prompted for Ambari Admin Credentials
- Utility Provides Three Options for Synchronization
  - Specific Set of User and Group
  - Synchronize Existing User and Group with LDAP
  - All User and Groups
- Review Log Files - Provide Information for Failed Synchronization Attempts
- /var/log/ambari-server/ambari-server.log



To perform this operation, your Ambari Server must be running.

When prompted, you must provide credentials for an Ambari Admin.

When syncing ldap, Local user accounts with matching username will switch to LDAP type, which means their authentication will be against the external LDAP and not against the Local Ambari user store.

LDAP sync only syncs up-to-1000 users. If your LDAP contains over 1000 users and you plan to import over 1000 users, you must use the --users option when syncing and specify a filtered list of users to perform import in batches.

When synchronizing LDAP users and groups, Ambari uses LDAP results paging controls to synchronize large numbers of LDAP objects. Most modern LDAP servers support these control, but for those that do not, such as Oracle Directory Server Enterprise Edition 11g, Ambari introduces a configuration parameter to disable

Configure Ambari for LDAP/AD Authentication

Sync'ing Specific Set of Users/Groups

- Use Option to Synchronize a Specific Set of Users/Groups
- Provide Command With a Text File of Comma-Separated Users and/or Groups
- The comma separated entries in each of these files should be based off of the values in LDAP of the attributes chosen during setup.
- The "User name attribute" should be used for the users.txt file, and the "Group name attribute" should be used for the groups.txt file.
  - Group membership is determined using the Group Membership Attribute (groupMembershipAttr)
  - User name is determined by using the Username Attribute (usernameAttribute)
- Command Will Find, Import, and Synchronize the Matching LDAP Entities with Ambari

  ambari-server sync-ldap --user users.txt --group groups.txt

Use this option to synchronize a specific set of users and groups from LDAP into Ambari. Provide the command a text file of comma-separated users and groups. The comma separated entries in each of these files should be based off of the values in LDAP of the attributes chosen during setup. The "User name attribute" should be used for the users.txt file, and the "Group name attribute" should be used for the groups.txt file. This command will find, import, and synchronize the matching LDAP entities with Ambari.

Configure Ambari for LDAP/AD Authentication

Sync'ing Existing Users/Groups

- After Performed Option to Synchronize a Specific Set of Users/Groups
- Use Option to Synchronize Existing Entities Within Ambari

  ambari-server sync-ldap --existing

- Users That No Longer Exist in LDAP Will Be Removed
- Group Membership in Ambari Update to Match LDAP
  - Group membership is determined using the Group Membership Attribute (groupMembershipAttr)

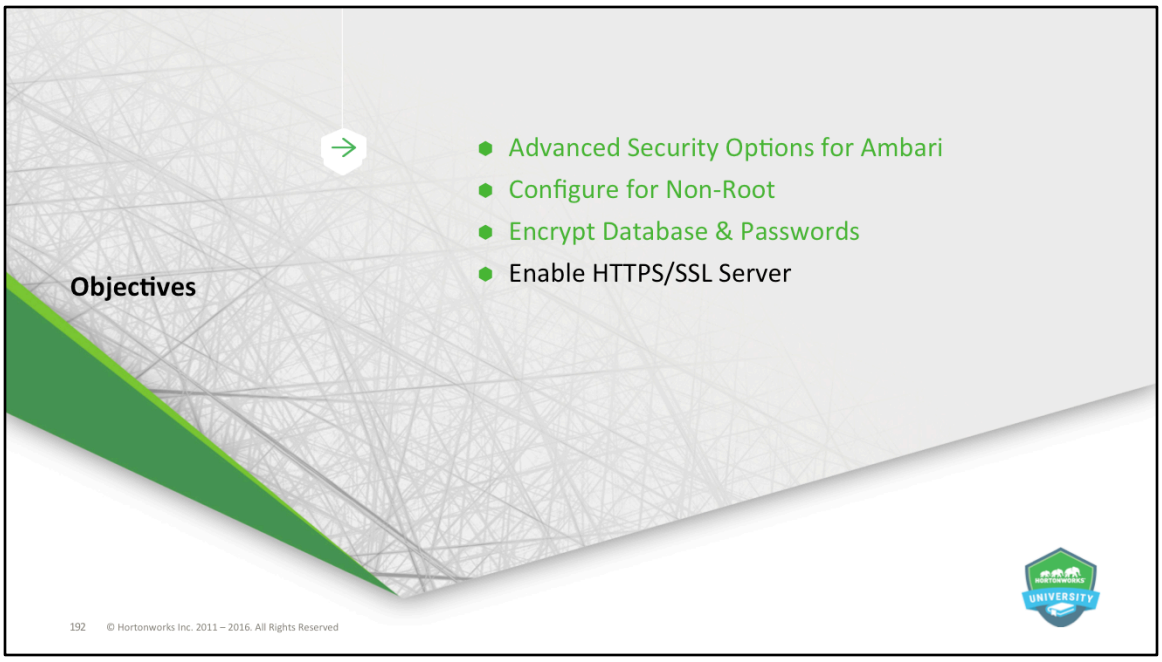
Configure Ambari for LDAP/AD Authentication

Sync'ing ALL Users/Groups

- Use Option to Synchronize ALL Users/Groups Within Ambari

  ambari-server sync-ldap --all

- This will Import All Entities with Matching LDAP User/Group Object Classes


Objectives
Advanced Security Options for Ambari
Configure for Non-Root
Encrypt Database & Passwords
Enable HTTPS/SSL Server
192
© Hortonworks Inc. 2011 – 2016. All Rights Reserved

Set Up SSL for Ambari Server

- Set Up Ambari Server to Utilize HTTPS Connections
- Obtain a Certificate or Create Temporary Self-Signed Certificate
  - Stop Ambari Server
  - Determine the Location of Certificate
  - Run Security Setup Command, Answer Prompts

    # ambari-server setup-security
  - Select 1 - "Enable HTTPS for Ambari server"
  - "Do you want to configure HTTPS ?" - Enter "y"
  - Select Port to Use for SSL - Default Port Number 8443
  - Provide Complete Path to Certificate File and Private Key File
  - Provide Password for Private Key
  - Start Ambari Server

Set Up SSL for Ambari Server

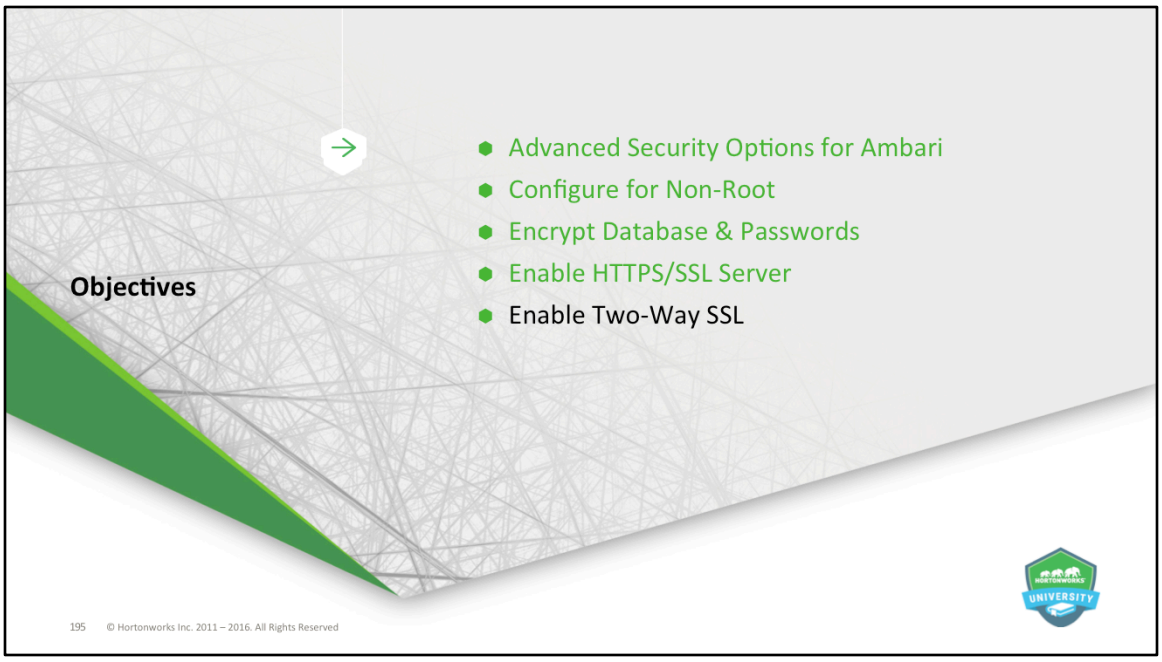
- Set Up Ambari Server to Utilize HTTPS Connections - Self-Signed Certificate
- Create a Temporary Self-Signed Certificate

  # openssl genrsa -out ambari.key 2048

  # openssl req -new -key ambari.key -out ambari.csr

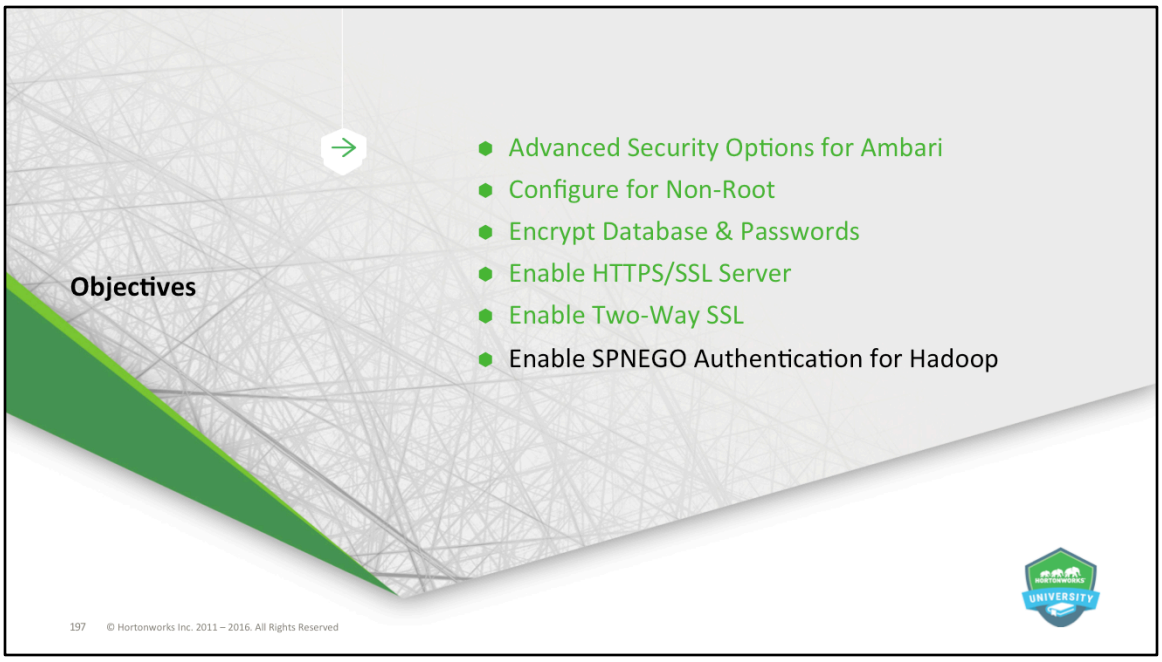
  # openssl x509 -req -days 365 -in ambari.csr -signkey ambari.key -out ambari.crt
- Certificate Must Be PEM-encoded Not DER-encoded
  - To Convert DER-encoded to PEM-encoded Using the Following Command

    #openssl x509 -in ambari.crt -inform der -outform pem -out ambari.pem
- Copy ambari.key, ambari.csr and ambari.crt - /etc/security/ssl directory


Objectives
Advanced Security Options for Ambari
Configure for Non-Root
Encrypt Database & Passwords
Enable HTTPS/SSL Server
Enable Two-Way SSL
195
© Hortonworks Inc. 2011 – 2016. All Rights Reserved

Set Up Two-Way SSL Between Ambari Server/Agents

- Two-Way SSL Provides Encrypted Communication Between Ambari Server and Agents
- Disabled By Default
  - Stop Ambari Server
  - Enable Two-Way SSL By Editing ambari.properties File
    /etc/ambari-server/conf/ambari.properties
  - Add Property - security.server.two_way_ssl = true
  - Start Ambari Server
- Agent Certificates are Downloaded Automatically During Agent Registration


Objectives
Advanced Security Options for Ambari
Configure for Non-Root
Encrypt Database & Passwords
Enable HTTPS/SSL Server
Enable Two-Way SSL
Enable SPNEGO Authentication for Hadoop
197
© Hortonworks Inc. 2011 – 2016. All Rights Reserved

Enabling SPNEGO Authentication for Hadoop

- By Default Access to HTTP-Based Services/UI's Do Not Require Authentication
- Kerberos Authentication Can Be Configured to Web UI's
- Prerequisite - Ambari Server Configured for Kerberos
- Configuring HTTP Authentication for HDFS, YARN, MR2, Hbase, Oozie, Falcon, Storm
- Create Secret Key Used for Signing Authentication Tokens
- File Should Contain Random Data and Placed on Each Node
- Owned by "hdfs" User; Group Owned by "hadoop" Group
- Permission Set - "440" or "-r--r----

```
# dd if=/dev/urandom of=/etc/security/http_secret bs=1024 count=1
# chown hdfs:hadoop /etc/security/http_secret
# chmod 440 /etc/security/http_secret
```

In order for Ambari to work with a cluster in which authenticated HTTP access to the Web UI's is required, you must configure the Ambari Server for Kerberos. Refer to Set Up Kerberos for Ambari Server for more information.

Enabling SPNEGO Authentication for Hadoop

- Open Services > HDFS > Configs
- Add/Modify The Following Properties – Advanced/Custom core-site:

Property	Value
hadoop.http.authentication.simple.anonymous.allowed	false
hadoop.http.authentication.signature.secret.file	/etc/security/http_secret
hadoop.http.authentication.type	kerberos
hadoop.http.authentication.kerberos.keytab	/etc/security/keytabs/spnego.service.keytab
hadoop.http.authentication.kerberos.principal	HTTP/_HOST@HORTONWORKS.COM
hadoop.http.filter.initializers	org.apache.hadoop.security.AuthenticationFilterInitializer
hadoop.http.authentication.cookie.domain	hortonworks.com

- Save Configuration
- Restart Affected Services

The entries listed in the above table in bold and italicized are site-specific. The hadoop.http.authentication.cookie.domain property is based off of the fully qualified domain names of the servers in the cluster. For example if the FQDN of your NameNode is host1.hortonworks.local, the hadoop.http.authentication.cookie.domain should be set to hortonworks.local.

Knowledge Check

Knowledge Check

1. True/False - Ambari Server and Agents must operate using root user access?
2. What protocol can be used to limit access to the Ambari Server?
3. What setup in Ambari Server will allow Kerberos KDC administration account credentials be retained?
4. Why is SPNEGO authentication needed?
5. Why must Ambari Server be configured for Kerberos?

1. FALSE

2. HTTPS with a self-signed or CA certificate

3. Database and Password Encryption

4. To require Kerberos authentication for HTTP-based services and User Interfaces.

5. In order for Ambari to work with a cluster in which authenticated HTTP access to the Web UI's is required, you must configure the Ambari Server for Kerberos

Summary

Summary

- Secure environments require restricting access to and limiting services that run as root.
- Ambari Server and Agents can operate without direct root user access.
- Ambari retains the Kerberos KDC administration account credentials and therefore the Ambari database and AD/LDAP passwords require encryption.
- Ambari utilizes an internal database as a user store for Ambari authentication and authorization and must be synchronized with AD/LDAP users and groups.
- Ambari Server can limit access to HTTPS connections utilizing a self-signed or CA certificate.
- Two-way SSL provides a way to encrypt communication between Ambari Server and Ambari Agents.
- SPNEGO authentication can be configured to require Kerberos authentication for HTTP-based services and User Interfaces (UI's).

Lab: Security Options for Ambari

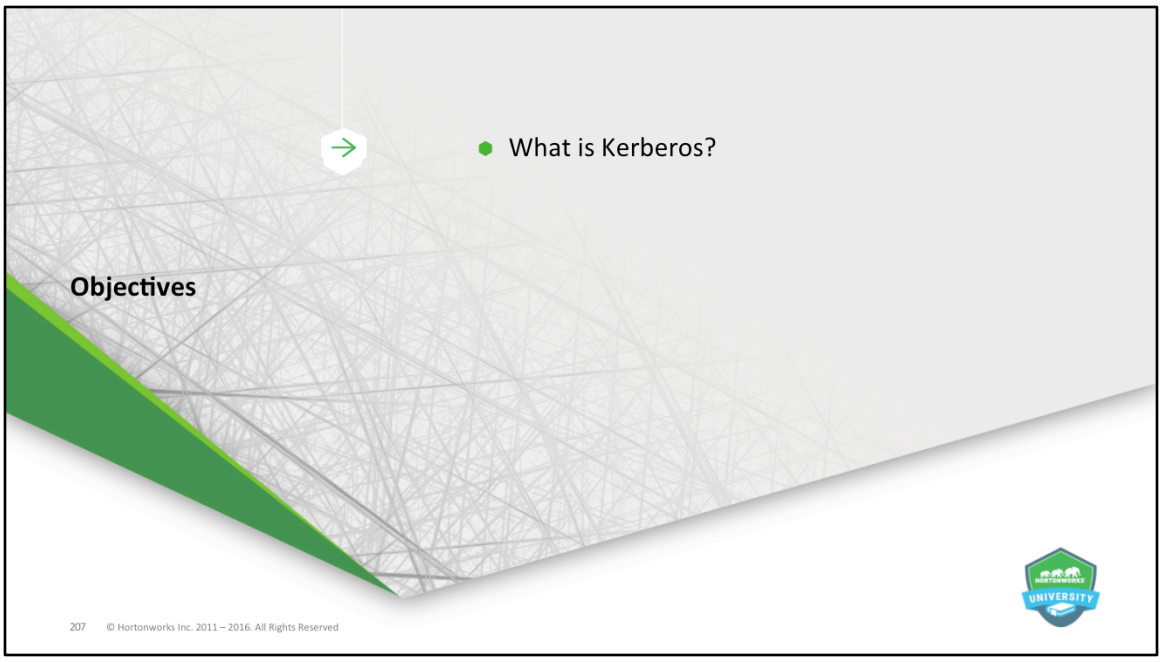

Kerberos Deep Dive
HORTONWORKS
UNIVERSITY

Objectives

After completing this lesson, students should be able to:

- Describe Kerberos
- Explain why Kerberos is needed
- Describe the Kerberos architecture
- Plan and deploy Kerberos Master KDC and Slave KDC

After completing this lesson, students should be able to:
Describe Kerberos
Explain why Kerberos is needed
Describe the Kerberos architecture
Plan and deploy Kerberos Master KDC and Slave KDC


What is Kerberos?
Objectives
207
© Hortonworks Inc. 2011 – 2016. All Rights Reserved

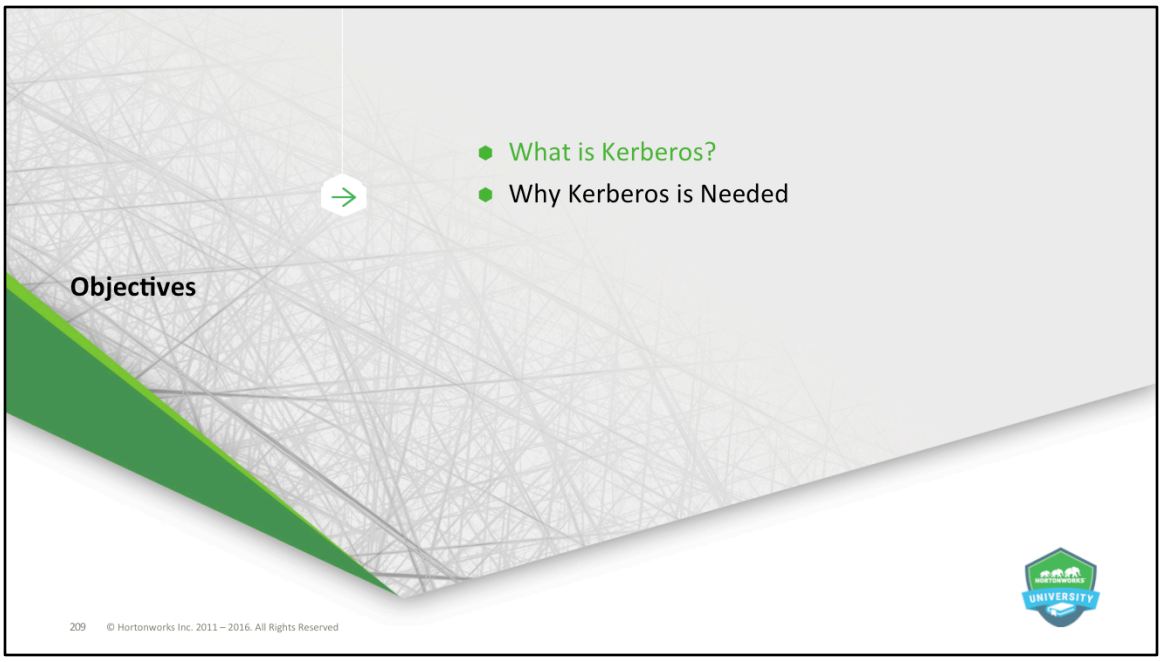
What is Kerberos

- Network authentication protocol
- Created to solve network security problems
- Designed to provide strong authentication/authorization
- Uses strong secret-key cryptography
- Encrypt communication to assure privacy & data integrity
- Free implementation from MIT
- Shipped with all major operating systems
  - Unix/Linux
  - Apple Macintosh
  - Windows
- Available in commercial products

Kerberos was originally developed for MIT's Project Athena in the 1980s and has grown to become the most widely deployed system for authentication and authorization in modern computer networks. Kerberos is currently shipped with all major computer operating systems and is uniquely positioned to become a universal solution to the distributed authentication and authorization problem of permitting universal "single sign-on" within and between federated enterprises and peer-to-peer communities. MIT has developed and maintains implementations of Kerberos software for the Apple Macintosh, Windows and Unix operating systems.

Kerberos is a network authentication protocol. It is designed to provide strong authentication for client/server applications by using secret-key cryptography. A free implementation of this protocol is available from the Massachusetts Institute of Technology. Kerberos is available in many commercial products as well.

The Internet is an insecure place. Many of the protocols used in the Internet do not provide any security. Tools to "sniff" passwords off of the network are in common use


Objectives
What is Kerberos?
Why Kerberos is Needed
209
© Hortonworks Inc. 2011 – 2016. All Rights Reserved

Why Kerberos is Needed

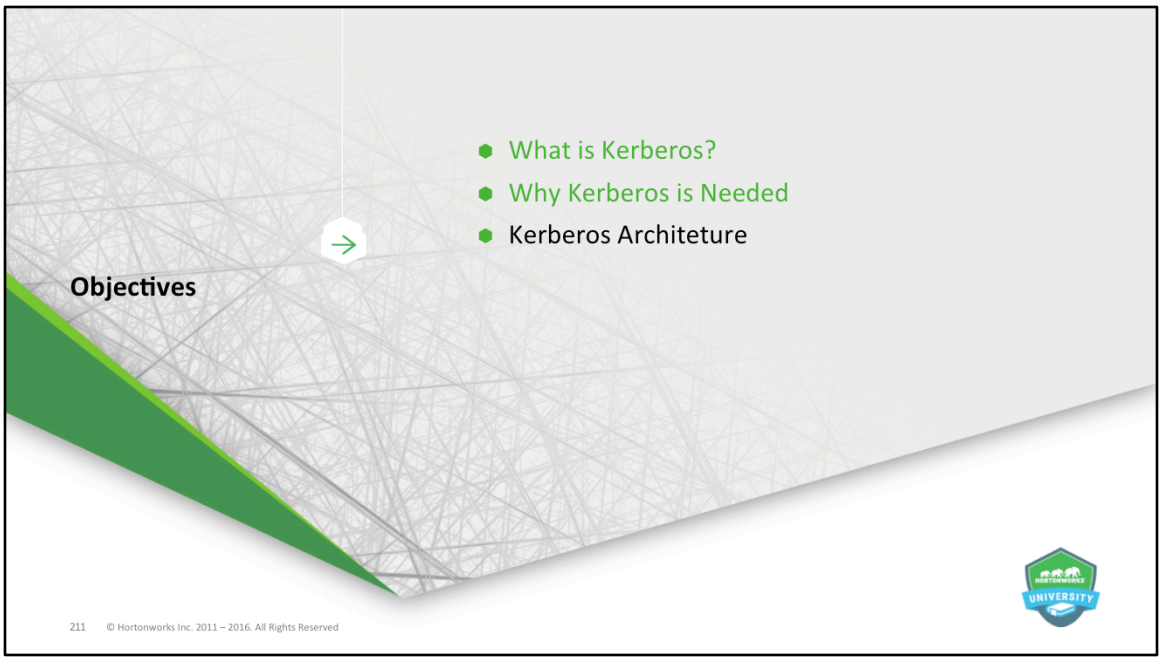
- Internet is insecure place
- Many protocols do not provide any security
- Application send unencrypted passwords over network
- Application rely on client to restrict user activities
- Client/server applications rely on client to be “honest”

- Solution to network security problems
- Provides tools for authentication and strong cryptography
- Secure information systems across enterprise

Kerberos was originally developed for MIT's Project Athena in the 1980s and has grown to become the most widely deployed system for authentication and authorization in modern computer networks. Kerberos is currently shipped with all major computer operating systems and is uniquely positioned to become a universal solution to the distributed authentication and authorization problem of permitting universal "single sign-on" within and between federated enterprises and peer-to-peer communities. MIT has developed and maintains implementations of Kerberos software for the Apple Macintosh, Windows and Unix operating systems.

Kerberos is a network authentication protocol. It is designed to provide strong authentication for client/server applications by using secret-key cryptography. A free implementation of this protocol is available from the Massachusetts Institute of Technology. Kerberos is available in many commercial products as well.

The Internet is an insecure place. Many of the protocols used in the Internet do not provide any security. Tools to "sniff" passwords off of the network are in common use by malicious hackers. Thus, applications which send an unencrypted password over


Objectives
What is Kerberos?
Why Kerberos is Needed
Kerberos Architeture
211
© Hortonworks Inc. 2011 – 2016. All Rights Reserved

Kerberos Architecture

- Designed to provide reliable authentication over open/insecure networks
- No guarantees if computer used are themselves vulnerable

Protocol Goals

- User's Password - Must
  - Never travel over network
  - Never be stored in any form
  - Discarded after use
  - Never stored unencrypted
  - Entered once per session – Single Sign On (SSO)

The Kerberos protocol is designed to provide reliable authentication over open and insecure networks where communications between the hosts belonging to it may be intercepted. However, one should be aware that Kerberos does not provide any guarantees if the computers being used are vulnerable: the authentication servers, application servers (imap, pop, smtp, telnet, ftp, ssh , AFS, lpr, ...) and clients must be kept constantly updated so that the authenticity of the requesting users and service providers can be guaranteed.

The above points justify the sentence: "Kerberos is an authentication protocol for trusted hosts on untrusted networks". By way of example, and to reiterate the concept: Kerberos' strategies are useless if someone who obtains privileged access to a server, can copy the file containing the secret key. Indeed, the intruder will put this key on another machine, and will only have to obtain a simple spoof DNS or IP address for that server to appear to clients as the authentic server.

The user's password must never travel over the network;

The user's password must never be stored in any form on the client machine: it must

Kerberos Architecture

Protocol Goals - Continued

- Authentication Information Management
  - Centralized and resides on authentication server – AS
  - User information not contained on any application servers – essential for:
    - Administrator disable account of any user by acting in a single location
    - User changes password, changed for all services at the same time
    - No redundancy of authentication information
- Mutual Authentication
  - User have to demonstrate who they say
  - Application server must prove authenticity when requested
- Encrypted Connection
  - Provides support for generation & exchange of encryption keys
  - Used to encrypt data

The Kerberos protocol is designed to provide reliable authentication over open and insecure networks where communications between the hosts belonging to it may be intercepted. However, one should be aware that Kerberos does not provide any guarantees if the computers being used are vulnerable: the authentication servers, application servers (imap, pop, smtp, telnet, ftp, ssh , AFS, lpr, ...) and clients must be kept constantly updated so that the authenticity of the requesting users and service providers can be guaranteed.

The above points justify the sentence: "Kerberos is an authentication protocol for trusted hosts on untrusted networks". By way of example, and to reiterate the concept: Kerberos' strategies are useless if someone who obtains privileged access to a server, can copy the file containing the secret key. Indeed, the intruder will put this key on another machine, and will only have to obtain a simple spoof DNS or IP address for that server to appear to clients as the authentic server.

The user's password must never travel over the network;

The user's password must never be stored in any form on the client machine: it must

Kerberos Architecture

Components

- Realm
  - Authentication Administrative Domain
  - Intention to Establish Boundaries an AS has Authority to Authenticate – User/Host/Service
  - Defined Using UPPER CASE LETTERS – `HORTONWORKS.COM`
- Principal
  - Named Used to Refer to Entries in the AS Database
  - Associated with Each User/Host/Service in Given Realm
  - General Format:
    `HTTP/node1.hortonworks.com@HORTONWORKS.COM`

http://www.kerberos.org/software/tutorial.html

Kerberos Architecture

Components

- Ticket
  - Something client presents to application server
  - Demonstrate authenticity of identity
  - Issued by AS
  - Encrypted using secret key of intended service
  - Secret key only shared between AS and server providing service
  - Has an Expiration
- Encryption
  - Uses only symmetrical key encryption
  - Needed to encrypt/decrypt messages
  - Types Include: DES, RC4-HMAC, Triple DES (3DES) & Newer AES128, AES256

http://www.kerberos.org/software/tutorial.html

Kerberos Architecture

Components

- Key Distribution Center (KDC)
  - Fundamental object in authentication of user/services
  - Single process logically divided into three parts:
    - Database
      - Container for Entries Associated with User/Services - Principal
      - Encrypted Using Master Key
    - Authentication Server (AS)
      - Replies to Initial Authentication Requests From Client/User – Must Enter Password
      - In Response Issues Special Ticket – Ticket Granting Ticket or TGT
    - Ticket Granting Server (TGS)
      - Distributes Service Tickets to Clients with Valid TGT
      - Guaranteeing Authenticity of Identity for Requested Resource on Application Servers

http://www.kerberos.org/software/tutorial.html

Kerberos Architecture

Components

- Session Key
  - User shares secret with service during time work session is open on server
  - Generated by KDC When Ticket Issued
  - Plays Fundamental Role in Demonstrating Authenticity of User
- Authenticator
  - Along with request containing ticket the client adds this packet
  - User principal and time stamp (its at that time) are included and encrypts with Session Key
- Replay Cache
  - The capacity to remember Authenticators which arrived within the last 2 minutes
  - Reject Authenticators if they are replicas

http://www.kerberos.org/software/tutorial.html

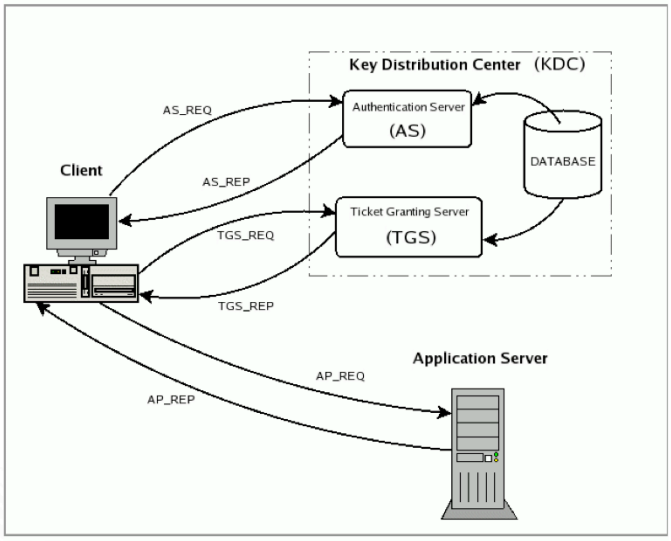
Kerberos Architecture

Components

- Credential Cache
  - To implement the single sign-on (SSO) characteristic
  - Necessary to memorize tickets and related session keys
  - Placed in area of memory accessible only to kernels and not swappable to disk

http://www.kerberos.org/software/tutorial.html

Kerberos Architecture



http://www.kerberos.org/software/tutorial.html

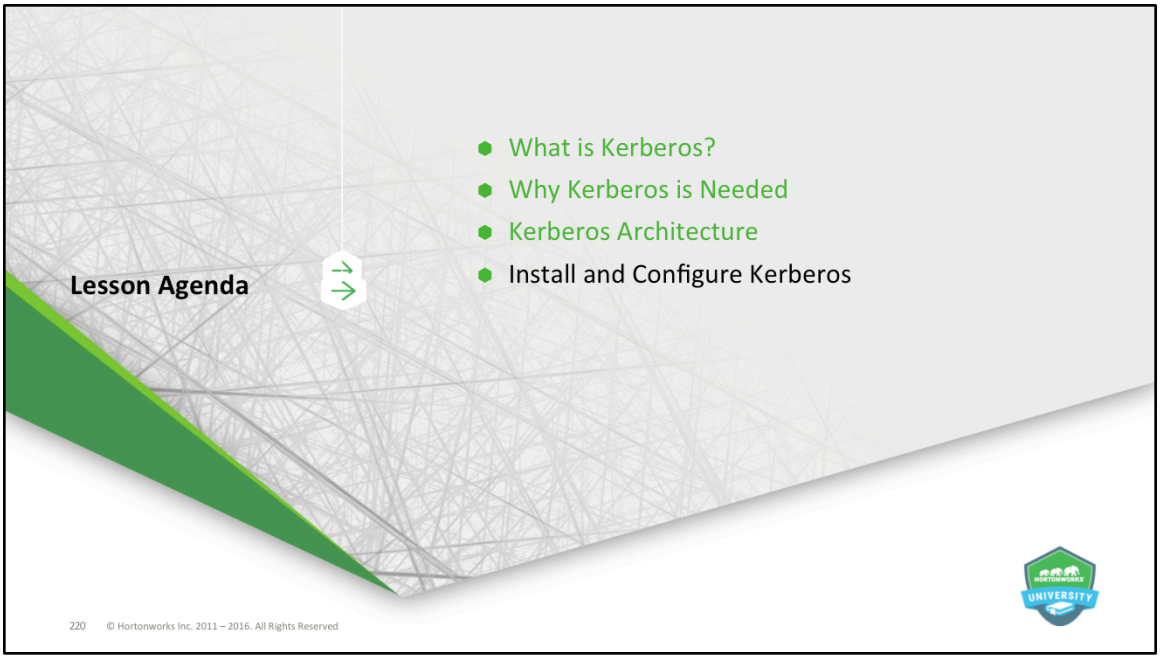
AS_REQ is the initial user authentication request (i.e. made with kinit) This message is directed to the KDC component known as Authentication Server (AS);

AS_REP is the reply of the Authentication Server to the previous request. Basically it contains the TGT (encrypted using the TGS secret key) and the session key (encrypted using the secret key of the requesting user);

TGS_REQ is the request from the client to the Ticket Granting Server (TGS) for a service ticket. This packet includes the TGT obtained from the previous message and an authenticator generated by the client and encrypted with the session key;

TGS_REP is the reply of the Ticket Granting Server to the previous request. Located inside is the requested service ticket (encrypted with the secret key of the service) and a service session key generated by TGS and encrypted using the previous session key generated by the AS;

AP_REQ is the request that the client sends to an application server to access a service. The components are the service ticket obtained from TGS with the previous reply and an authenticator again generated by the client, but this time encrypted


Lesson Agenda
What is Kerberos?
Why Kerberos is Needed
Kerberos Architecture
Install and Configure Kerberos
220
© Hortonworks Inc. 2011 – 2016. All Rights Reserved

Kerberos Master KDC Installation

Kerberos Master KDC

- Make sure Network Time Protocol (NTP) is setup on all nodes
- Kerberos Packages
  - `krb5-libs`
  - `krb5-server`
  - `krb5-workstation`
- Install Kerberos Master

```
# yum install krb5-libs krb5-server krb5-workstation
```

- Edit `/etc/krb5.conf` – Define Your REALM
  - Set `default_domain` to REALM Name
  - Change the "`realms`" Name and "`domain_realm`" mappings
  - Set `kdc` and `admin_server` Variables to Resolvable Hostname of KDC Host(s)

Kerberos Master KDC

/etc/krb5.conf

```
[libdefaults]
    default_realm = HORTONWORKS.COM
    dns_lookup_realm = false
    dns_lookup_kdc = false
    ticket_lifetime = 24h
    renew_lifetime = 7d
    forwardable = yes
[realms]
    HORTONWORKS.COM = {
        kdc = kdcmaster.hortonworks.com:88
        kdc = kdcslave.hortonworks.com:88
        admin_server = kdcmaster.hortonworks.com:749
        default_domain = hortonworks.com }
[domain_realm]
    .hortonworks.com = HORTONWORKS.COM
    hortonworks.com = HORTONWORKS.COM
```

Kerberos Master KDC

- Edit `/var/kerberos/krb5kdc/kdc.conf`
- Add a `key_stash_file` entry: `/var/kerberos/krb5kdc/.k5.<REALM Name>`
- Add the `kadmind_port` entry: `kadmind_port = 749`
- The stash file allows KDC server to start up without needing password

Kerberos Master KDC

/var/kerberos/krb5kdc/kdc.conf

```
[kdcdefaults]
 kdc_ports = 88
 kdc_tcp_ports = 88

[realms]
 HORTONWORKS.COM = {
        master_key_type = aes256-cts
        acl_file = /var/kerberos/krb5kdc/kadm5.acl
        dict_file = /usr/share/dict/words
        admin_keytab = /var/kerberos/krb5kdc/kadm5.keytab
        key_stash_file = /var/kerberos/krb5kdc/.k5.HORTONWORKS.COM
        kadmind_port = 749
        supported_enctypes = aes256-cts:normal aes128-cts:normal des3-hmac-sha1:normal
        }
```

Kerberos Master KDC

- Create the KDC database

```
# kdb5_util create -s
```

- Prompts for Master KDC Password
- Remember Password Entered
- The `-s` Option Creates the Stash File
- Add administrative user to the KDC database

```
# /usr/kerberos/sbin/kadmin.local -q "addprinc admin/
admin@HORTONWORKS.COM"
```

- Edit `/var/kerberos/krb5kdc/kadm5.acl` - Change the "`admin`" Permission

```
admin/admin@HORTONWORKS.COM  *
```

Kerberos Master KDC

- Start the Kerberos daemons

```
# service krb5kdc start
# service kadmin start
```

- The Master KDC Ready For Use
- Add Kerberos daemons to automatically restart after reboot

```
# chkconfig krb5kdc on
# chkconfig kadmin on
```

- Verify Kerberos Master KDC is up and functioning

```
$ kinit admin/admin@HORTONWORKS.COM
```

Password for `admin@HORTONWORKS.COM`:

```
$ klist – List the Ticket Granting Ticket (TGT)
```

Kerberos Slave KDC Installation

Kerberos Slave KDC

- Install Kerberos Master

```
# yum install krb5-lib krb5-server krb5-workstation
```

- Copy configuration file from Kerberos Master KDC to appropriate directories
  - `krb5.conf`
  - `kdc.conf`
  - `kadm5.acl`
  - Master Key Stash File
- Create host principals for each KDC Host

```
# kadmin addprinc -randkey host/kdcmaster.hortonworks.com@HORTONWORKS.COM
# kadmin addprinc -randkey host/kdcslave.hortonworks.com@HORTONWORKS.COM
```

- Extract random keys for all KDCs – store on each host

```
# kadmin ktadd host/kdcmaster.hortonworks.com@HORTONWORKS.COM
# kadmin ktadd host/kdcslave.hortonworks.com@HORTONWORKS.COM
```

Kerberos Slave KDC

- Create `kpropd.acl` File Containing "`host`" Principals for each KDC
  - `host/kdcmaster.hortonworks.com@HORTONWORKS.COM`
  - `host/kdcslave.hortonworks.com@HORTONWORKS.COM`
- Ensure /`etc/services` List "`krb5_prop`"
- Propagate database to Slave KDC

```
# kdb5_util dump /var/kerberos/krb5kdc/slave_datatrans
# kprop -f /var/kerberos/krb5kdc/slave_datatrans
kdcslave.hortonworks.com
```

Database propagation to `kdcslave.hortonworks.com`: SUCCEEDED

- To automate propagation of database
  - Create script to dump and propagate
  - Create cron job to execute script

Kerberos Slave KDC

- The Slave KDC ready for use

```
# service krb5kdc start
```

- Add Kerberos daemons to automatically restart after reboot

```
# chkconfig krb5kdc on
```

Kerberos Clients Install via Ambari

Kerberos Client

- Utilize Ambari automated Kerberos setup
- Ambari installs Kerberos Client on each cluster host
- Copies `/etc/krb5.conf` file to each host

Knowledge Check

Knowledge Check

1. True/False - Kerberos was created by MIT.
2. What is Kerberos designed to provide?
3. True/False - User passwords are stored and retained after use.
4. Name the three parts of the Key Distribution Center.

1. TRUE

2. Strong and reliable authentication for client/server applications using secret-key cryptography

3. FALSE

4. Database, Authentication Server, Ticket Granting Server

Summary

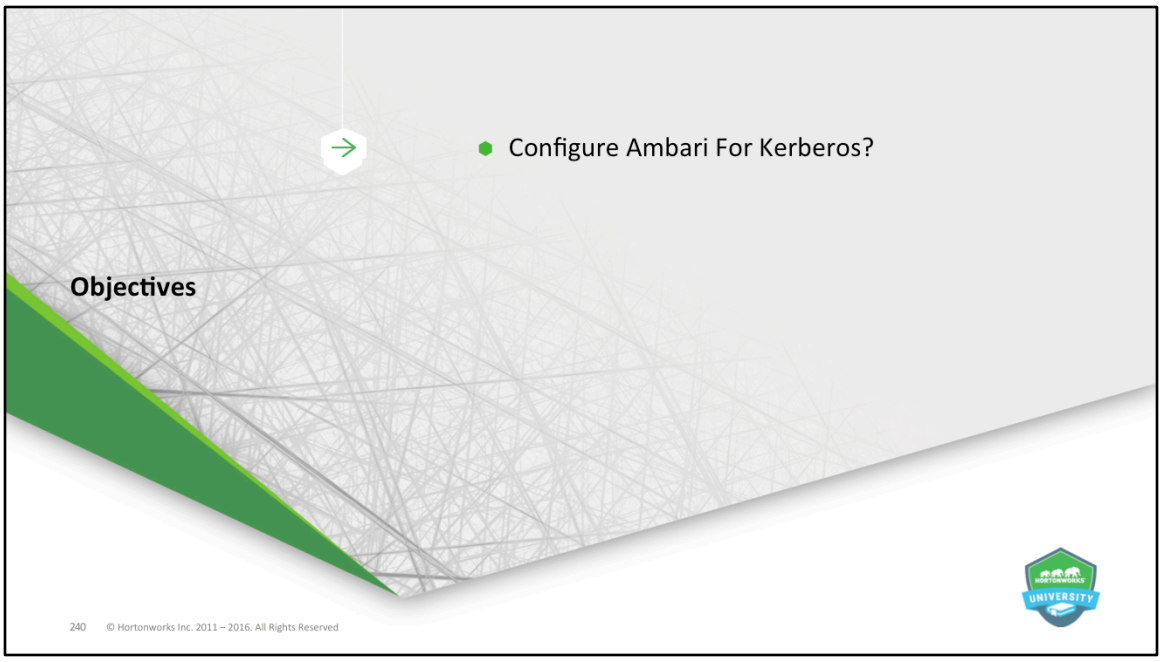
Summary

- Kerberos is a network authentication protocol created by Massachusetts Institute of Technology as a solution to network security issues.
- Designed to provide strong and reliable authentication for client/server applications by using secret-key cryptography.
- Kerberos software is available from the MIT and supports all major Operating Systems.


Enable Kerberos
HORTONWORKS
UNIVERSITY

Objectives

- Configure Ambari for Kerberos
- Configure Hadoop for Kerberos
- Establish Trust Relationship between AD and Kerberos Master KDC


Configure Ambari For Kerberos?
Objectives
240
© Hortonworks Inc. 2011 – 2016. All Rights Reserved

Configure Ambari for Kerberos

- SPNEGO is Simple and Protected GSSAPI Negotiation Mechanism
- Kerberos Enabled Cluster Requires SPNEGO Authentication for Component REST Endpoints
- Ambari Web Need Access to these API's
- Ambari Server Requires Kerberos Principal to Authenticate Via SPNEGO

- Install Kerberos Packages
- Copy /etc/krb5.conf to Ambari Server
- Test Access to KDC Master

  # kinit admin/admin@HORTONWORKS.COM

  Password for admin/admin@HORTONWORKS.COM:

  # klist – Will Display Ticket for admin

Simple and Protected GSSAPI (Generic Security Service Application Program Interface) Negotiation Mechanism (SPNEGO), often pronounced "spenay-go", is a GSSAPI "pseudo mechanism" used by client-server software to negotiate the choice of security technology. SPNEGO is used when a client application wants to authenticate to a remote server, but neither end is sure what authentication protocols the other supports. The pseudo-mechanism uses a protocol to determine what common GSSAPI mechanisms are available, selects one and then dispatches all further security operations to it. This can help organizations deploy new security mechanisms in a phased manner. https://en.wikipedia.org/wiki/SPNEGO

When a cluster is enabled for Kerberos, the component REST endpoints (such as the YARN ATS component) require SPNEGO authentication.

Depending on the Services in your cluster, Ambari Web needs access to these APIs. As well, views such as the Tez View need access to ATS. Therefore, the Ambari Server requires a Kerberos principal in order to authenticate via SPNEGO against these APIs. This section describes how to configure Ambari Server with a Kerberos principal and

Configure Ambari for Kerberos

- Ambari Server Requires Kerberos Principal to Authenticate Via SPNEGO
- Create Ambari Server Principal on Master KDC

  # kadmin

  kadmin: addprinc –randkey ambari.hortonworks.com@HORTONWORKS.COM
- Generate Keytab for Ambari Server on Master KDC

  kadmin: xst –k ambari.server.keytab ambari.hortonworks.com@HORTONWORKS.COM
- Place Keytab File on Ambari Server – Set Permission

  # cp /tmp/ambari.server.keytab /etc/security/keytabs/ambari.server.keytab

  # chmod 400 /etc/security/keytabs/ambari.server.keytab

When a cluster is enabled for Kerberos, the component REST endpoints (such as the YARN ATS component) require SPNEGO authentication.

Depending on the Services in your cluster, Ambari Web needs access to these APIs. As well, views such as the Tez View need access to ATS. Therefore, the Ambari Server requires a Kerberos principal in order to authenticate via SPNEGO against these APIs. This section describes how to configure Ambari Server with a Kerberos principal and keytab to allow views to authenticate via SPNEGO against cluster components.

Create a principal in your KDC for the Ambari Server. For example, using kadmin:

addprinc -randkey ambari-server@EXAMPLE.COM

Generate a keytab for that principal.

xst -k ambari.server.keytab ambari-server@EXAMPLE.COM

Place that keytab on the Ambari Server host. Be sure to set the file permissions so the

Configure Ambari for Kerberos

- Stop Ambari Server
- Configure Ambari Server for Security - Run setup-security command

  # ambari-server setup-security

  – Select 3 for Setup Ambari Kerberos JAAS Configuration

  – Enter Ambari Server's Principal Name – "ambari.hortonworks.com@HORTONWORKS.COM"

  – Enter Path to Keytab File – "/etc/security/keytabs/ambari.server.keytab"
- Start Ambari Server

When a cluster is enabled for Kerberos, the component REST endpoints (such as the YARN ATS component) require SPNEGO authentication.

Depending on the Services in your cluster, Ambari Web needs access to these APIs. As well, views such as the Tez View need access to ATS. Therefore, the Ambari Server requires a Kerberos principal in order to authenticate via SPNEGO against these APIs. This section describes how to configure Ambari Server with a Kerberos principal and keytab to allow views to authenticate via SPNEGO against cluster components.

Create a principal in your KDC for the Ambari Server. For example, using kadmin:

addprinc -randkey ambari-server@EXAMPLE.COM

Generate a keytab for that principal.

xst -k ambari.server.keytab ambari-server@EXAMPLE.COM

Place that keytab on the Ambari Server host. Be sure to set the file permissions so the

Configure Ambari for Kerberos - Active Directory

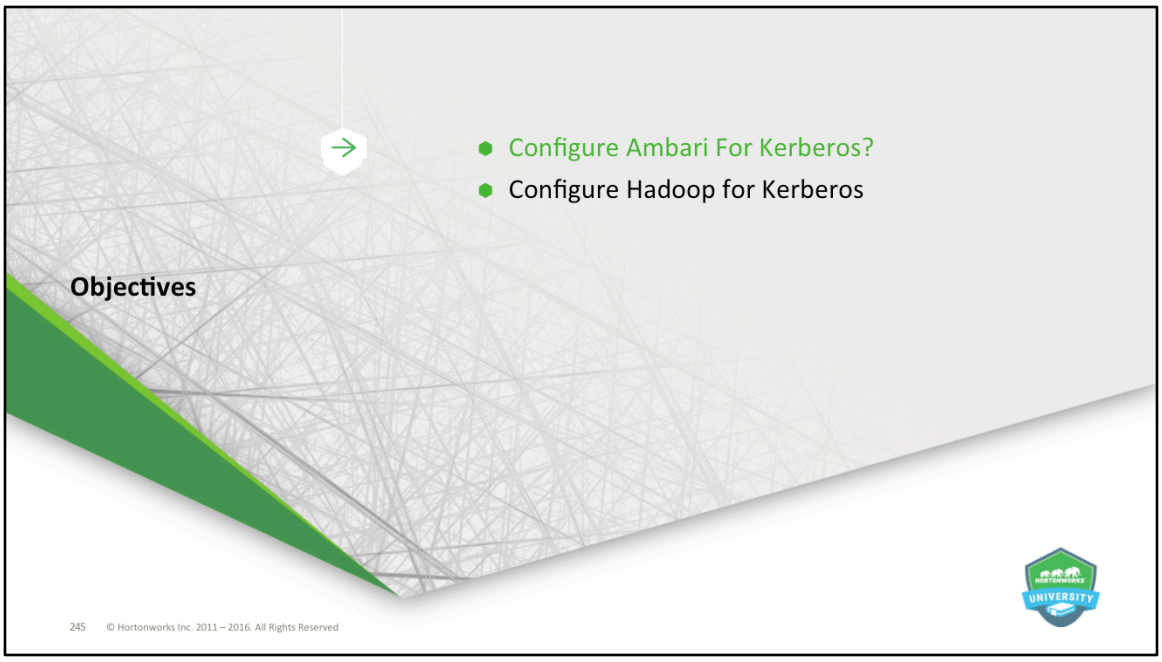
- Active Directory entry for Ambari Server
- Creating the Principal Using PowerShell - AD

  New-ADUser -Name ambari -Path OU=ServiceUsers,DC=lab,DC=hortonworks,DC=net -samAccountName ambari -UserPrincipalName ambari@LAB.HORTONWORKS.NET -AccountPassword BadPass#1 -Enabled $true
- Creating the Keytab File Using PowerShell - AD

  ktpass -out ambari.keytab -princ ambari@LAB.HORTONWORKS.NET -pass BadPass#1 -mapuser ambari@LAB.HORTONWORKS.NET -mapop set -crypto All -ptype KRB5_NT_PRINCIPAL
- SCP/FTP Keytab File from AD Server to Ambari Server

  # cp /tmp/ambari.keytab /etc/security/keytabs/ambari.keytab

  # chmod 400 /etc/security/keytab/ambari.keytab


Configure Ambari For Kerberos?
Configure Hadoop for Kerberos
Objectives
245
© Hortonworks Inc. 2011 – 2016. All Rights Reserved

Configure Hadoop for Kerberos

Installing Java Cryptography Extension (JCE) Security Policy Files – Ambari Server

- Obtain Unlimited Strength JCE Policy file for Oracle JDK Version Installed on Cluster
- Save JCE Zip File in Temporary Location
- Copy JCE Zip File to All Cluster Nodes
- Add Unlimited Strength JCE Security Policy jars into Installed JDK Directory
  /usr/jdk64/jdk<VERSION NUMBER>/jre/lib/security
- Restart Ambari Server

- OpenJDK has Unlimited Strength JCE Policy files in place already
  - Above steps not needed for OpenJDK

Configure Hadoop for Kerberos

Running Kerberos Security Wizard – Ambari Server

- Log in to Ambari Web – Click on Admin > Select Kerberos
- Click on "Enable Kerberos"
- Select "Existing Active Directory" and Confirm the Prerequisites
- Wizard Prompts for:
  - Master KDC Server
  - KDC Admin Account
  - Service and Ambari Principals

Configure Hadoop for Kerberos

Running Kerberos Security Wizard – Ambari Server – Continued

- Next, Kerberos Client will be Installed/Configured on All Cluster Hosts
- Next, Kerberos Identities Used By Hadoop can be Customized
  - Review Principals on General Tab
  - By Default Cluster Name Appended to Each Principal
  - Example - hdfs-training@HORTONWORKS.COM
- Next, Confirm Configuration – Optionally Download CSV File of Principals
- Next, Services will be stopped

Configure Hadoop for Kerberos

Launching Kerberos Wizard - Continued

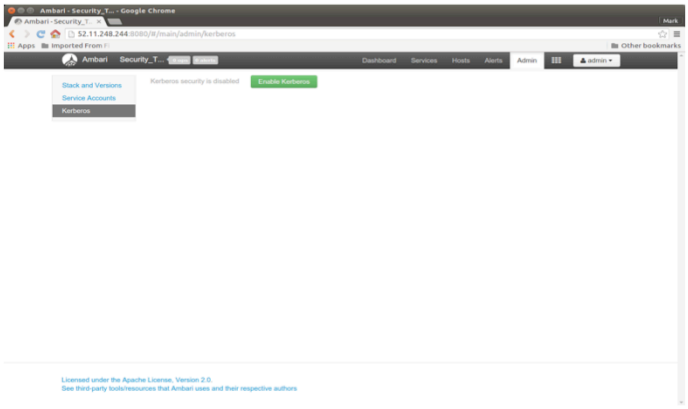
- Next, Cluster will be Kerberized by Configuring Services for Kerberos
  - Principals will be Created
  - Keytabs Generated/Distributed
  - Cluster Configurations Updated
- Next, Services Started to Authenticate against the Master KDC and Tested
- Exit Wizard When Completed

Customize the Kerberos identities used by Hadoop and proceed to kerberize the cluster.

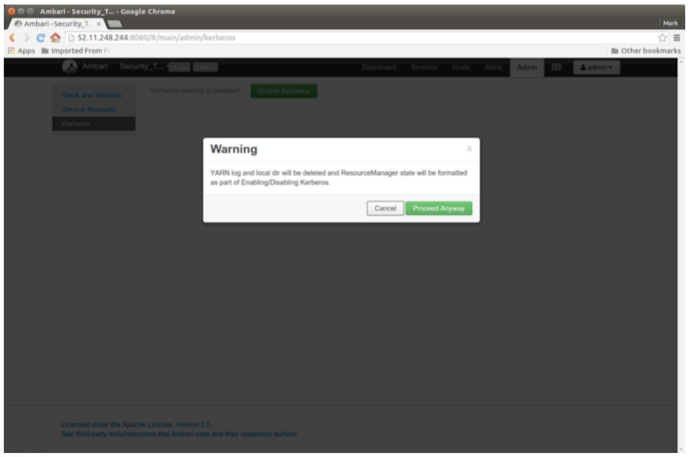
On the Configure Identities step, be sure to review the principal names, particularly the Ambari Principals on the General tab. These principal names, by default, append the name of the cluster to each of the Ambari principals. You can leave this as default or adjust these by removing the "-${cluster-name}" from principal name string. For example, if your cluster is named HDP and your realm is EXAMPLE.COM, the hdfs principal will be created as hdfs-HDP@EXAMPLE.COM.

Configure Hadoop for Kerberos

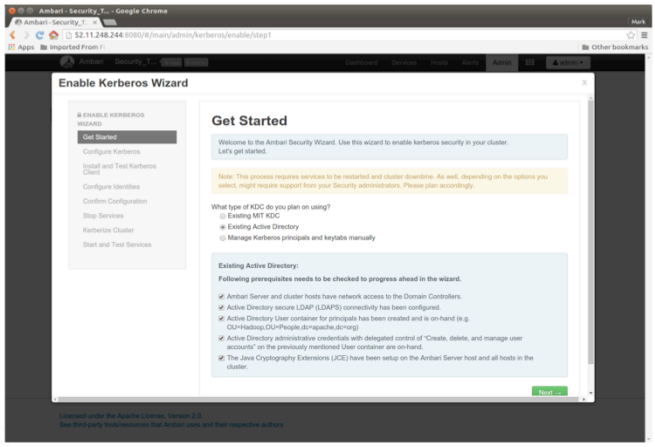
Configure Hadoop for Kerberos



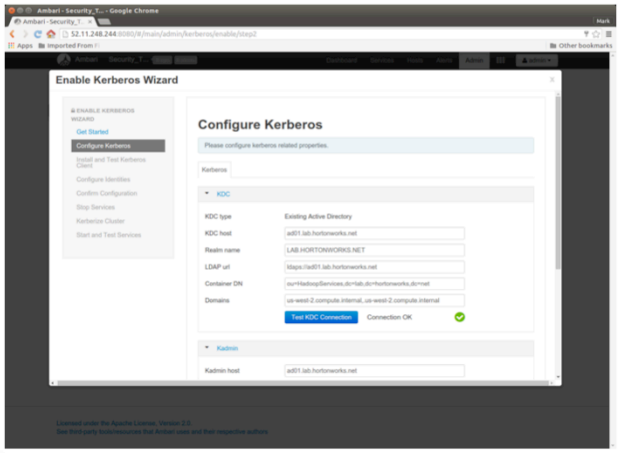
Configure Hadoop for Kerberos



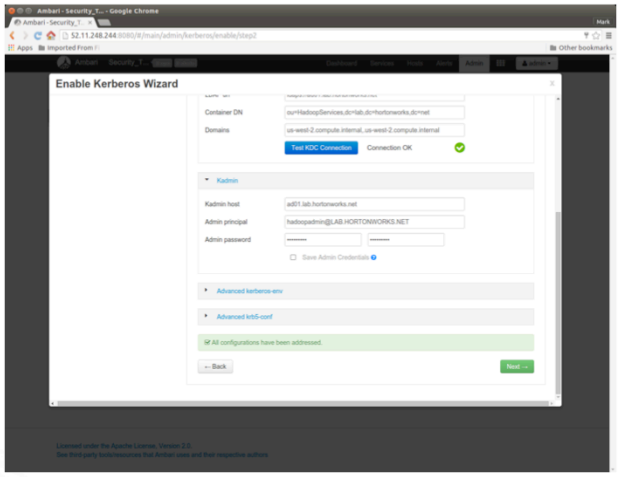
Configure Hadoop for Kerberos



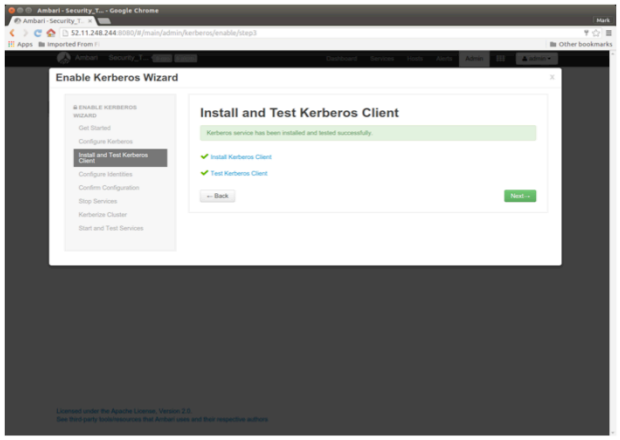
Configure Hadoop for Kerberos



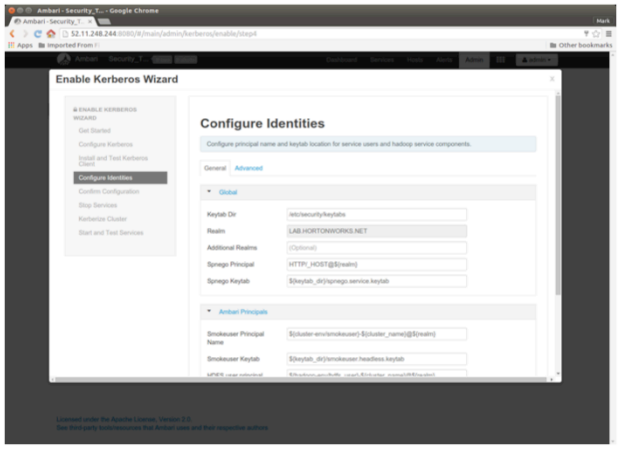
Configure Hadoop for Kerberos



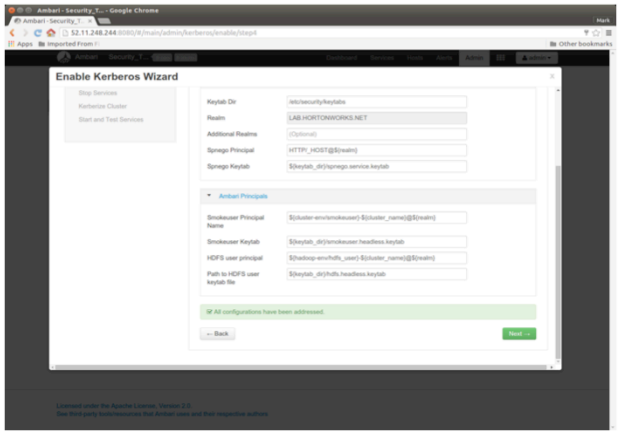
Configure Hadoop for Kerberos



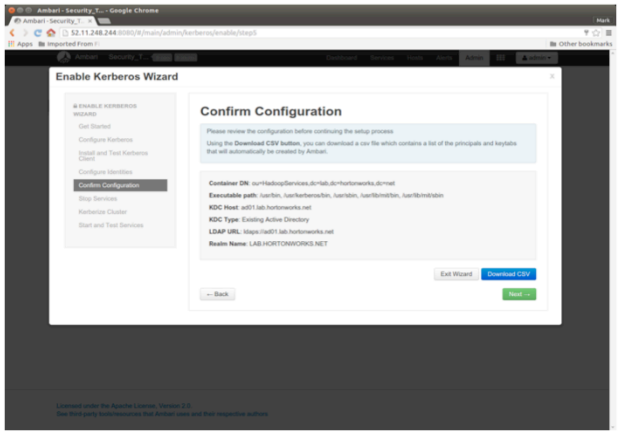
Configure Hadoop for Kerberos



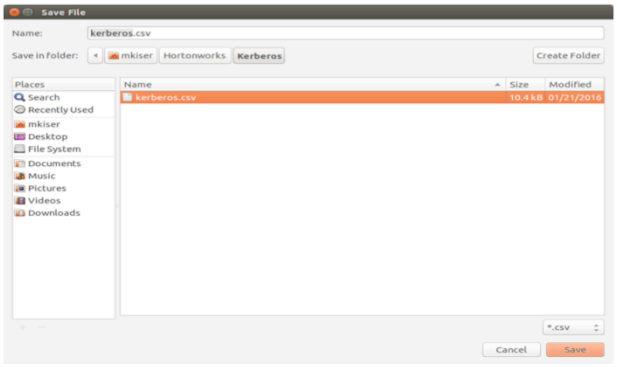
Configure Hadoop for Kerberos



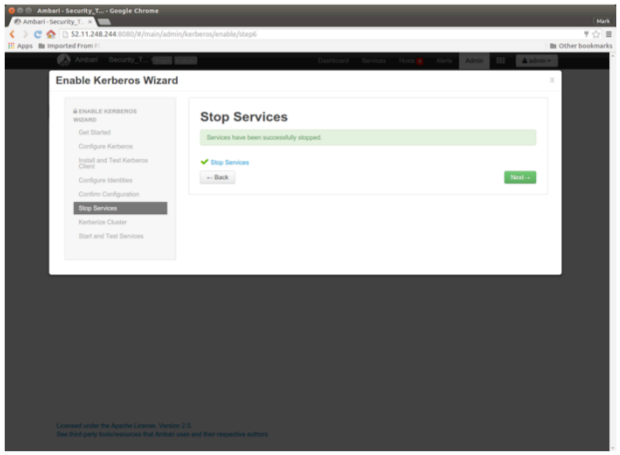
Configure Hadoop for Kerberos



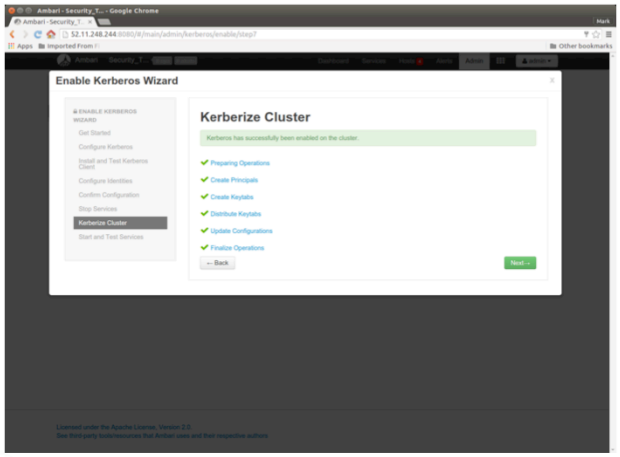
Configure Hadoop for Kerberos



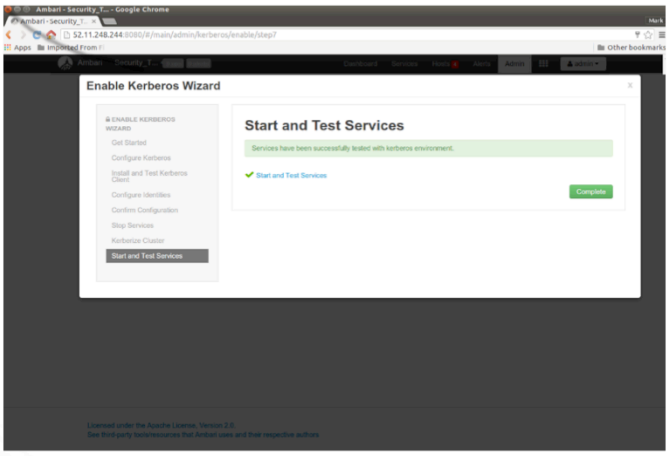
Configure Hadoop for Kerberos



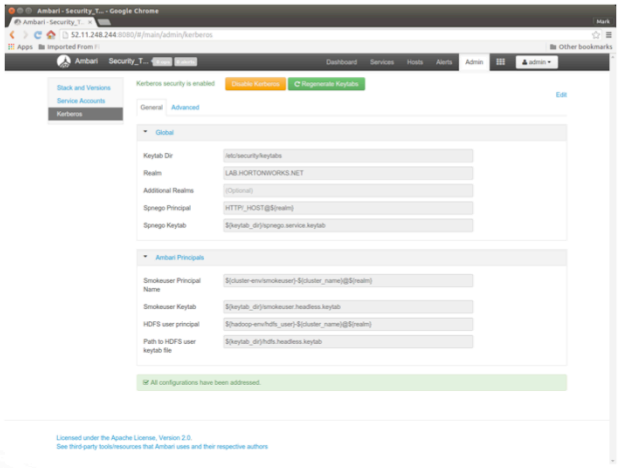
Configure Hadoop for Kerberos

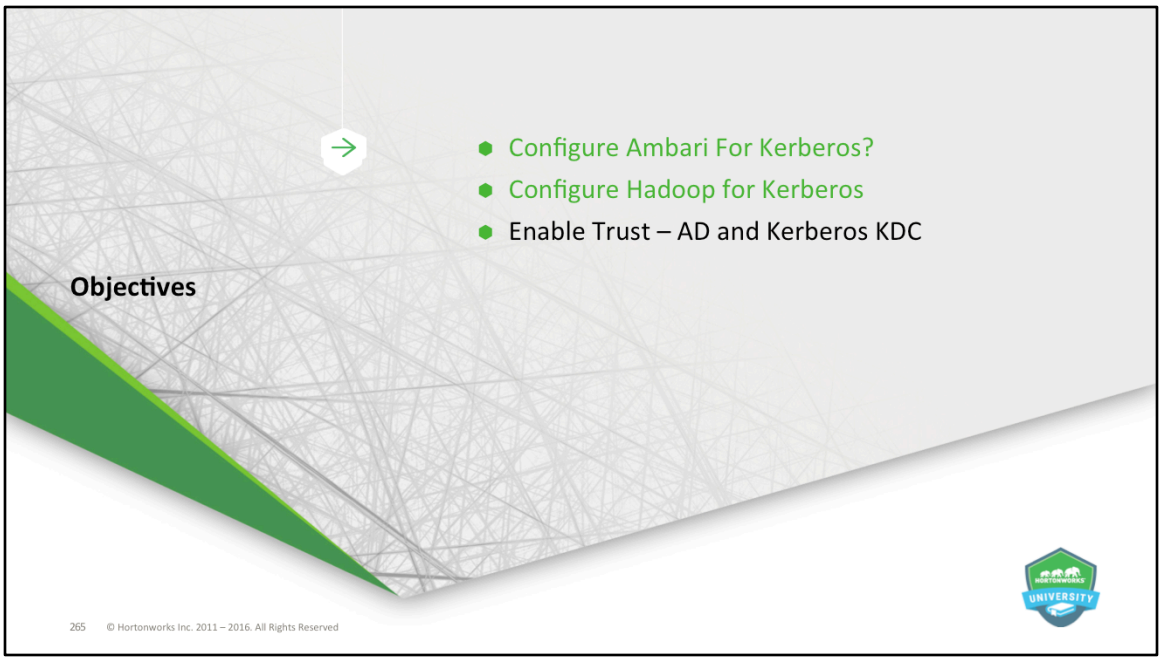


Configure Hadoop for Kerberos



Configure Hadoop for Kerberos




Objectives
Configure Ambari For Kerberos?
Configure Hadoop for Kerberos
Enable Trust – AD and Kerberos KDC
265
© Hortonworks Inc. 2011 – 2016. All Rights Reserved

Enable Trust – AD and Kerberos KDC

Setting Up One-way Trust with Active Directory

Configure Hadoop REALM on the Active Directory Domain Controller Server

- Add Hadoop Kerberos REALM and Master/Slave KDC to Domain Controller DC

  ksetup /addkdc HORTONWORKS.COM kdcmaster.hortonworks.com

- Establish One-Way Trust Between AD Domain and Hadoop REALM

  netdom trust HORTONWORKS.COM /Domain:CORP.AD.DOMAIN /add /realm /passwordt:<trust_password>

- Optional – Create Hostmap for Hadoop Service Hosts
  - Needed when:
    - Windows Clients Need Access Hadoop Services and
    - Domain Does Not Have Search Route to Find Services in Hadoop REALM

  ksetup /addhosttorealmmap <hadoop-service-host> HORTONWORKS.COM

Configure the Hadoop realm on the AD DC server and set up the one-way trust.

1. Add the Hadoop Kerberos realm and KDC host to the DC:

ksetup /addkdc $hadoop.realm $KDC-host

2. Establish one-way trust between the AD domain and the Hadoop realm:

netdom trust $hadoop.realm /Domain:$AD.domain /add /realm /passwordt:
$trust_password

3. (Optional) If Windows clients within the AD domain need to access Hadoop Services, and the domain does not have a search route to find the services in Hadoop realm, run the following command to create a hostmap for Hadoop service host:

Enable Trust – AD and Kerberos KDC

Configure Hadoop REALM on the Active Directory Domain Controller Server

- Optional – Define Encryption Type
  - To List Available Encryption Types:

    ksetup /GetEncTypeAttr HORTONWORKS.COM
  - Set Encryption Types Based on Security Requirements

    ksetup /SetEncTypeAttr HORTONWORKS.COM <encryption_type>
  - Mismatched Encryption Types Causes Problems
  - Verify Encryption Type is Configured for Hadoop REALM in /etc/krb5.conf File

Configure the Hadoop realm on the AD DC server and set up the one-way trust.

1. Add the Hadoop Kerberos realm and KDC host to the DC:

ksetup /addkdc $hadoop.realm $KDC-host

2. Establish one-way trust between the AD domain and the Hadoop realm:

netdom trust $hadoop.realm /Domain:$AD.domain /add /realm /passwordt: $trust_password

3. (Optional) If Windows clients within the AD domain need to access Hadoop Services, and the domain does not have a search route to find the services in Hadoop realm, run the following command to create a hostmap for Hadoop service host:

Enable Trust – AD and Kerberos KDC

- Add the AD Domain as REALM to Kerberos Configuration on All Cluster Nodes
- Edit /etc/krb5.conf Add the Following Properties:

```
[libdefaults]
default_domain = HORTONWORKS.COM
default_tkt_enctypes = aes256-cts aes128-cts rc4-hmac arcfour-hmac-md5 des-cbc-md5 des-cbc-crc
default_tgs_enctypes = aes256-cts aes128-cts rc4-hmac arcfour-hmac-md5 des-cbc-md5 des-cbc-crc
permitted_enctypes = aes256-cts aes128-cts rc4-hmac arcfour-hmac-md5 des- cbc-md5 des-cbc-crc
udp_preference_limit = 1
[realms]
CORP.AD.DOMAIN= {
kdc = ad-server.hortonworks.com
admin_server = ad-server.hortonworks.com
default_domain = ad-server.hortonworks.com
}
```

Add the AD domain as a realm to the krb5.conf on the Hadoop cluster hosts. Optionally configure encryption types and UDP preferences.

Open the krb5.conf file with a text editor and make the following changes:

To libdefaults, add the following properties.

Set the Hadoop realm as default:

[libdefaults]

default_domain = $hadoop.realm

Set the encryption type:

[libdefaults]

Enable Trust – AD and Kerberos KDC

- Add the AD Domain as REALM to Kerberos Configuration on All Cluster Nodes
- Add Trust Principal for AD Domain to Master Kerberos KDC

  # kadmin addprinc krbtgt/HORTONWORKS.COM@AD.CORP.DOMAIN

  – Command Will Prompt for Password – Use Same Trust Password

Add the AD domain as a realm to the krb5.conf on the Hadoop cluster hosts. Optionally configure encryption types and UDP preferences.

Open the krb5.conf file with a text editor and make the following changes:

To libdefaults, add the following properties.

Set the Hadoop realm as default:

[libdefaults]

default_domain = $hadoop.realm

Set the encryption type:

[libdefaults]

Knowledge Check

Questions

1. Where is the kerberos conf file located?
2. What are the 3 necessary steps to enable kerberos?
3. How does the ambari server authenticate via SPNEGO
4. The Enable Kerberos wizard will install the Kerberos Clients on all cluster hosts T/F

1. /etc/krb5.conf
2. Configure Ambari for Kerberos

   Configure Hadoop for Kerberos

   Establish Trust Relationship between AD and Kerberos Master KDC
3. Using a Kerbros principal
4. True

Summary

Summary

- When a cluster is enabled for Kerberos, the component REST endpoints require SPNEGO authentication.
- Depending on the Services in your cluster, Ambari Web needs access to these APIs.
- The Ambari Server requires a Kerberos principal in order to authenticate via SPNEGO against these APIs.
- The Ambari Web UI includes a wizard to "Enable Kerberos" on the Hadoop Cluster.
- The Enable Kerberos wizard will install the Kerberos Clients on all cluster hosts.
- Hortonworks Recommends using Active Directory Replica as the Kerberos KDC

Lab: Kerberize the Cluster