Identity Management: *The authentic & authoritative guide for the modern enterprise*

Ellen Newlands, Product Manager
Dmitri Pal, Director, Engineering
06-26-15
Goals of the Presentation

• Introduce Identity Management problem space

• Give you an overview of the identity management components in the Red Hat portfolio

• Provide examples of some real-world use cases that can be solved with the identity management capabilities Red Hat offers

• Show that these solutions are cost effective
Identity Management Problem Space
What is Identity Management?

● What does this mean to you?

● What issues are you running into in this area?
Wikipedia as the “authoritative source” for definitions:

*Identity Management - (noun)*

“Identity management (IdM) describes the management of individual principals, their authentication, authorization, and privileges within or across system and enterprise boundaries with the goal of increasing security and productivity while decreasing cost, downtime and repetitive tasks.”

*Wikipedia*
Identity Management Problem Space

• Identities
  – Where are my users stored? What properties do they have? How is this data made available to systems and applications?

• Authentication
  – What credentials do my users use to authenticate? Passwords? Smart Cards? Special devices? Is there SSO? How can the same user access file stores and web applications without requiring re-authentication?

• Access control
  – Which users have access to which systems, services, applications? What commands can they run on those systems? What SELinux context is a user is mapped to?

• Policies
  – What is the strength of the password? What are the automount rules? What are Kerberos ticket policies?
Overview of the Identity Management Components
Components of the Portfolio

- Identity Management in Red Hat Enterprise Linux (IdM)
- SSSD
- Certmonger
- Ipsilon IdP
- Apache modules
Identity Management

- Domain controller for Linux/UNIX environments
- Combines LDAP, Kerberos, DNS and certificate management capabilities
- Provides centralized authentication, authorization and identity information for Linux/UNIX infrastructure
- Enables centralized policy and privilege escalation management
- Integrates with Active Directory on the server-to-server level
SSSD:
(The System Security Services Daemon)

- Client-side component
- Part of Red Hat Enterprise Linux and many other Linux distributions
- Allows connecting a system to the identity and authentication source of your choice
- Caches identity and policy information for offline use
- Capable of connecting to different sources of identity data at the same time
Certmonger

- Client side component
- Connects to central Certificate Server and requests certificates
- Tracks and auto renews the certificates it is tracking
Ipsilon IdP

- Identity Provider implementation

- Allows federation between different applications using SAML based SSO
Apache Modules

- Modules that can be integrated with Apache server
- Modules that support forms-based, Kerberos or SAML authentication
- Authorization and identity data lookups are also possible using corresponding modules
Example Architecture

Active Directory

Trust

IdM

Linux Server

Business Application

Modules

SSSD

Certmonger

Active Directory

Business Application

IdP

IdM
Solving Real World Identity Management Challenges
Use Cases and Challenges

• How can I provide centralized authentication?
• Can I define access control to hosts without copying configuration files?
• Can I manage SSH keys for users and hosts?
• Can I provide centralized SUDO, automount, SELinux user mappings?
• Is there a cost effective solution that provides strong authentication using OTP?
• Can I provide a smooth SSO experience for my users inside the enterprise?
• How can I integrate my applications into the same identity space?
• How to address Active Directory interoperability challenges?
Centralized Authentication

Steps:
- Consolidate your user accounts
- Load your user data into a IdM
- Connect your Linux/UNIX systems to IdM
  - `ipa-client-install`

Why would I use IdM?
- Different authentication methods:
  - LDAP, Kerberos, OTP, Certificates
- Integrated solution
  - Easy to install and manage
- Integrates with AD
- Has a lot of other valuable capabilities
Use Cases and Challenges

- How can I provide centralized authentication?
- Can I define access control to hosts without copying configuration files?
- Can I manage SSH keys for users and hosts?
- Can I provide centralized SUDO, automount, SELinux user mappings?
- Is there a cost effective solution that provides strong authentication using OTP?
- Can I provide a smooth SSO experience for my users inside the enterprise?
- How can I integrate my applications into the same identity space?
- How to address Active Directory interoperability challenges?
Host Based Access Control

- Which users or group of users can access
- Which hosts or groups of hosts
- Using which login services (console, ssh, sudo, ftp, sftp, etc.)

- You define rules centrally
- Rules are enforced on the client
- Rules are cached
- There is a test tool to help you
Use Cases and Challenges

- How can I provide centralized authentication?
- Can I define access control to hosts without copying configuration files?
- **Can I manage SSH keys for users and hosts?**
- Can I provide centralized SUDO, automount, SELinux user mappings?
- Is there a cost effective solution that provides strong authentication using OTP?
- Can I provide a smooth SSO experience for my users inside the enterprise?
- How can I integrate my applications into the same identity space?
- How to address Active Directory interoperability challenges?
SSH Key Management

- Host public keys uploaded at the client installation time
- User can upload his public key to IdM manually
- When user SSHs from a system A the public key of to the target system B is delivered to system A (no need to validate digest)
- User public key is automatically delivered to system B
Use Cases and Challenges

- How can I provide centralized authentication?
- Can I define access control to hosts without copying configuration files?
- How I can manage SSH keys for users and hosts?
- **Can I provide centralized SUDO, automount, SELinux user mappings?**
- Is there a cost effective solution that provides strong authentication using OTP?
- Can I provide a smooth SSO experience for my users inside the enterprise?
- How can I integrate my applications into the same identity space?
- How to address Active Directory interoperability challenges?
SUDO Integration

- Centrally define commands and groups of commands
- Define which groups of users can run these commands or groups of commands on which hosts or groups of hosts
- Rules are enforced on client
- Rules are cached
- Capability is integrated into the sudo utility
SELinux User Mapping

- Mappings can be defined centrally
- Allow different users on different systems have different SELinux context
- Default SELinux labels are available in IPA configuration
- Mappings are enforced on the client
- Mappings are cached

Diagram:
- IdM
  - Linux system
    - Unprivileged
    - Privileged
    - Super privileged
Automount

- Define direct or indirect maps
- Associate maps with a particular location
- Configure clients to pull data from that location (part of the LDAP tree)

- Maps are defined centrally
- Maps are applied on the client
- Maps are cached
- Maps are integrated with autofs
Use Cases and Challenges

- How can I provide centralized authentication?
- Can I define access control to hosts without copying configuration files?
- How can I manage SSH keys for users and hosts?
- Can I provide centralized SUDO, automount, SELinux user mappings?
- **Is there a cost effective solution that provides strong authentication using OTP?**
- Can I provide a smooth SSO experience for my users inside the enterprise?
- How can I integrate my applications into the same identity space?
- How to address Active Directory interoperability challenges?
Two Factor Authentication

- Native 2FA
  - Yubikey, FreeOTP, Google authenticator
  - HOTP/TOTP compatible
  - Over LDAP or Kerberos
- Proxied over RADIUS
  - Any third party that has RADIUS support
  - Kerberos only
- Easy migration
Use Cases and Challenges

• How can I provide centralized authentication?
• Can I define access control to hosts without copying configuration files?
• How I can manage SSH keys for users and hosts?
• Can I provide centralized SUDO, automount, SELinux user mappings?
• Is there a cost effective solution that provides strong authentication using OTP?
• **Can I provide a smooth SSO experience for my users inside the enterprise?**
• How can I integrate my applications into the same identity space?
• How to address Active Directory interoperability challenges?
Kerberos Based SSO

1. NFS Client requests TGT from IdM/AD/Kerberos.
2. NFS Client uses TGT to obtain ST from IdM/AD/Kerberos.
3. NFS Client uses ST to request access to NFS Server.
4. NFS Server verifies ST and grants access.
5. NFS Server provides access to Kerberized application.
Kerberos SSO Flow

- User logs into the system that is connected to a Kerberos server
  - It can be: Kerberos KDC, Active Directory or IdM
- User authenticates (1) and receives a ticket granting ticket (TGT) from Kerberos server
- User accesses a resource (2), for example NFS client
- Kerberos library will request a service ticket from KDC on behalf of the user (3)
- Ticket is presented to the service, for example NFS server (4)
- Server or service decrypts using using its Kerberos key
- Keys are distributed at the configuration time, IdM provides a command to get the Kerberos keys for the client systems
SAML Based SSO

1. User accesses the application through a web browser.
2. The browser sends a request to Apache.
3. Apache passes the request to the Identity Provider (IdP).
4. The IdP interacts with the Identity Source to fetch user attributes.
5. The IdP sends the user attributes to Apache.
6. Apache decrypts and interprets the SAML module.
7. Apache passes the user attributes to the Application Framework.

Application Framework

User Attributes

Application

Identity Provider (IdP)

Identity Source

SAML module
SAML Flow

1. User starts browser and navigates to a resource or application
2. SAML component checks the presence of the assertion and redirects to IdP
3. IdP prompts user for authentication methods it supports
4. IdP uses some identity source to perform the authentication
5. IdP creates a SAML assertion and redirects browser back to the resource
6. SAML component checks the assertion and extracts user data from it
7. Data is passed to the application – user is authenticated
Use Cases and Challenges

• How can I provide centralized authentication?
• Can I define access control to hosts without copying configuration files?
• How I can manage SSH keys for users and hosts?
• Can I provide centralized SUDO, automount, SELinux user mappings?
• Is there a cost effective solution that provides strong authentication using OTP?
• Can I provide a smooth SSO experience for my users inside the enterprise?
• How can I integrate my applications into the same identity space?
• How to address Active Directory interoperability challenges?
Application Integration

Apache with modules
- Authentication
- Kerberos SSO
- SAML, OpenID...
- Identity

Linux Platform

SSSD

Application Framework
- Application

Identity Source

User Attributes

Identity

User Attributes

Application Framework
- Application

Identity Source
Use Cases and Challenges

- How can I provide centralized authentication?
- Can I define access control to hosts without copying configuration files?
- How can I manage SSH keys for users and hosts?
- Can I provide centralized SUDO, automount, SELinux user mappings?
- Is there a cost effective solution that provides strong authentication using OTP?
- Can I provide a smooth SSO experience for my users inside the enterprise?
- How can I integrate my applications into the same identity space?
- How to address Active Directory interoperability challenges?
AD Integration Options

Direct Integration

Active Directory

Linux system

Linux system

Linux system

Indirect Integration

Active Directory

Central Identity Server

Linux system

Linux system

Linux system
Direct Integration

**Active Directory**
- DNS
- LDAP
- KDC

Authentication can use LDAP or Kerberos

Can map AD SID to POSIX attributes or use SFU/IMU
Can join system into AD domain (realmd)
Leverages native AD protocols and LDAP/Kerberos

**Linux system**

**SSSD**
- Authentication
- Identities
- Name Resolution

**Policies**
- sudo
- hbac
- automount
- selinux

Policies are delivered via configuration files and managed locally or via a config server like Satellite or Puppet.
GPO support for HBAC is implemented since 7.1.
Indirect Integration

Domain trust is established on the Kerberos level. DNS zone can be delegated to IdM, can be a subdomain.

Client software connects to the right server depending on the information it needs.

Policies are managed centrally.
Cost Effectiveness
What is the cost?

- All mentioned components and solutions are provided using Red Hat Enterprise Linux without extra charge
- No third party vendors involved
- Deployment is easy and integrated – saves time
- The main cost is server side subscriptions, but one server can serve about 2-3K clients
Use Cases in Works
Use Cases in the Pipeline

- Integration of different products in Red Hat portfolio
- Smart Card authentication
- Central key store
- User lifecycle management
- One-way trusts
- DNSSEC
Future considerations

- Global catalog support
- Authentication indicator in tickets
- Integration with Samba 4 DC
- Full smart card management capabilities
- IdM to IdM trusts
Pointers and Resources
Resources

- Demo community instance of IdM (FreeIPA): http://www.freeipa.org/page/Demo
- Demo community instance of Ipsilon: https://saml.redhat.com/idp/
Questions!

- What use cases do you want us to address?

- What challenges do you have in your environment that we did not discuss in this presentation?