How to build an Identity Management System on Linux

Simo Sorce
Principal Software Engineer
Red Hat, Inc.

What is an Identity Management System and why should I care ?

- In a nutshell: an IdM system is a set of services and rules to manage the users of an organization.

- It includes information about individuals, computers, groups, roles, authentication and authorization rules that apply to the set of users and devices managed by the system.
- If you need to manage more than a handful of machines you do not want to manually configure all these functions on each one, instead you use an IdM system generally hosted on a centralized server.

Identities

- When you encounter the word *Identity* usually you think about a person, or a user.
- But computers and even single programs often need their own identity in order to be authorized to perform operations over a network.
- Identities are also often managed in groups to apply authorization decisions to multiple similar objects in a simpler/consistent way.

What do we need to manage

- At the core:
  - Users' life-cycle
    - Creation, deletion, and other status changes
    - Relations (groups, roles)
    - Policies (passwords, privileges)
  - Computers' life-cycle
    - Enrollment, retirement
    - Creation/Revocation of Keys (Kerberos, SSH, X509, ...)
    - Policies (Access control, authorization rules)
- Additionally
  - Other “security” related aspects of networking

Centralize or distribute ?

- Striking the right balance is not an easy task
  - Being able to flexibly shift balance between centralization and distribution based on the situation is nice, but also harder to implement in practice.
- This is a problem on multiple levels
  - Networking
    - How to spread services to avoid single points of failure ?
    - Distribute heavily ?
  - Security
    - How do we reduce attack surface ?
    - Centralize heavily ?
  - Administration
    - How can we allow delegation of tasks securely ?

Pros and Cons of Centralization

- Centralization is good because ...
  - Management is easier
  - Reporting is easier
  - Enforcement is easier
  - Development is easier
- ... on the other hand, distributing makes it ...
  - More resilient to failure
  - Scales better

Responsibilities of an IdM server ...

- Authentication for users and services
  - Passwords, SSO ? 2FA ?
  - Certificates, Keys
- Authorization rules for all services
  - Access rules per host
  - Users roles and admin delegation
- Network related administration ?
  - DNS, DHCP, ...
- Auditing and reporting

... and of the clients

- Retrieving Information
  - Users, Groups, netgroups, host groups, roles
  - Certificates, keytabs
  - Automount maps, other configuration
- Authentication
  - Passwords, tickets
- Authorization
  - HBAC, sudo rules, SSH keys, SELinux users
- Misc
  - DNS discovery, DNS Updates, time synchronization

There is a lot to manage

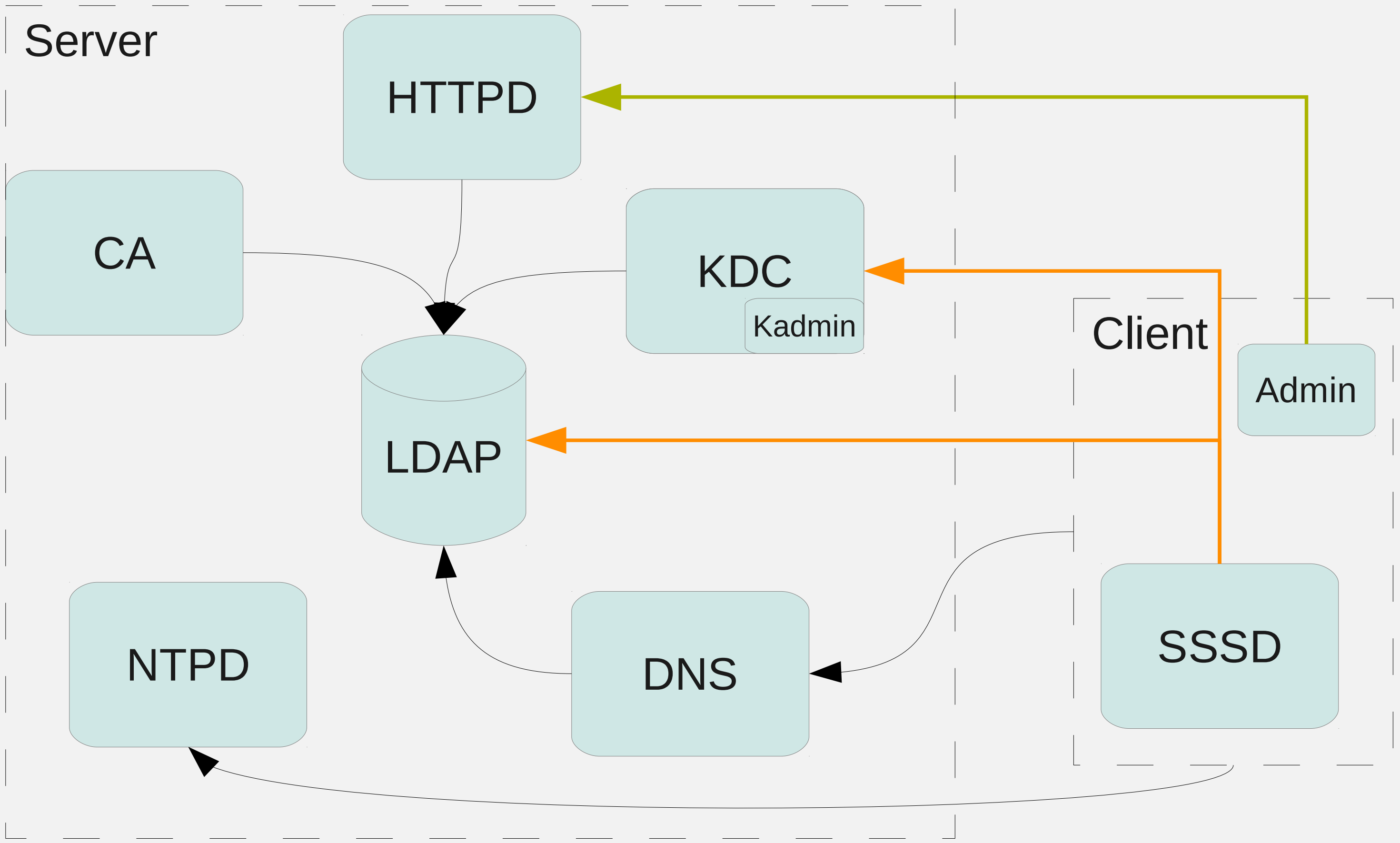
- Management tools are as important as the underlying technologies used
    - If it can't be managed effectively, it can't be used
    - Sadly management is very often overlooked in Free Software
- Security and Complexity are enemies
    - Complex interfaces need to be simplified to make them understandable to users
- Diagnostic tools are also important
    - Complex systems tend to break more easily
- Keep it simple if you can
    - If you can't, make it manageable at least

So, how hard can it be ?

- We just need to install an LDAP server and a Kerberos KDC right ?
  - Have you ever tried ? :-)

- Some numbers from the FreeIPA project
  - Installer: 4(NTP) + 35(DS) + 20(PKI) + 12(KDC) + 16(HTTPD) + 9(DNS) = 96 unique steps
    - This includes no replica, no clients, and only default rules
    - Time taken: approx. 5 minutes
  - Code: ~150k lines on top of existing projects

Basic Idm exploded (FreeIPA)



Why LDAP and Kerberos ?

- Why not a Custom (SQL?) Database ?
  - Integration, custom database = custom clients
  - Multi-master and read-only Replication
  - Fine grained Access Control
  - Interoperability, Standard
- Why LDAP is not enough ? Why Kerberos ?
  - Security: Passwords vs tickets vs certificates
  - Convenience: Single Sign On
  - Performance: Scalability, Availability
  - Security, Standard

Why PKI, DNS integration ?

- Some protocols can be secured only via SSL
  - HTTP, IMAP, SMTP, ..., VPN, ...
  - Central Authority for X509 certificates is a good idea
- DNS is crucial to identify machines
  - Service principals use DNS names
  - X509 Certificates use DNS names
  - SSH identify targets via DNS names
  - IPv6 is coming, very long addresses
  - But DNS is Insecure!
    - DNSSEC
    - (GSS-)TSIG DNS updates

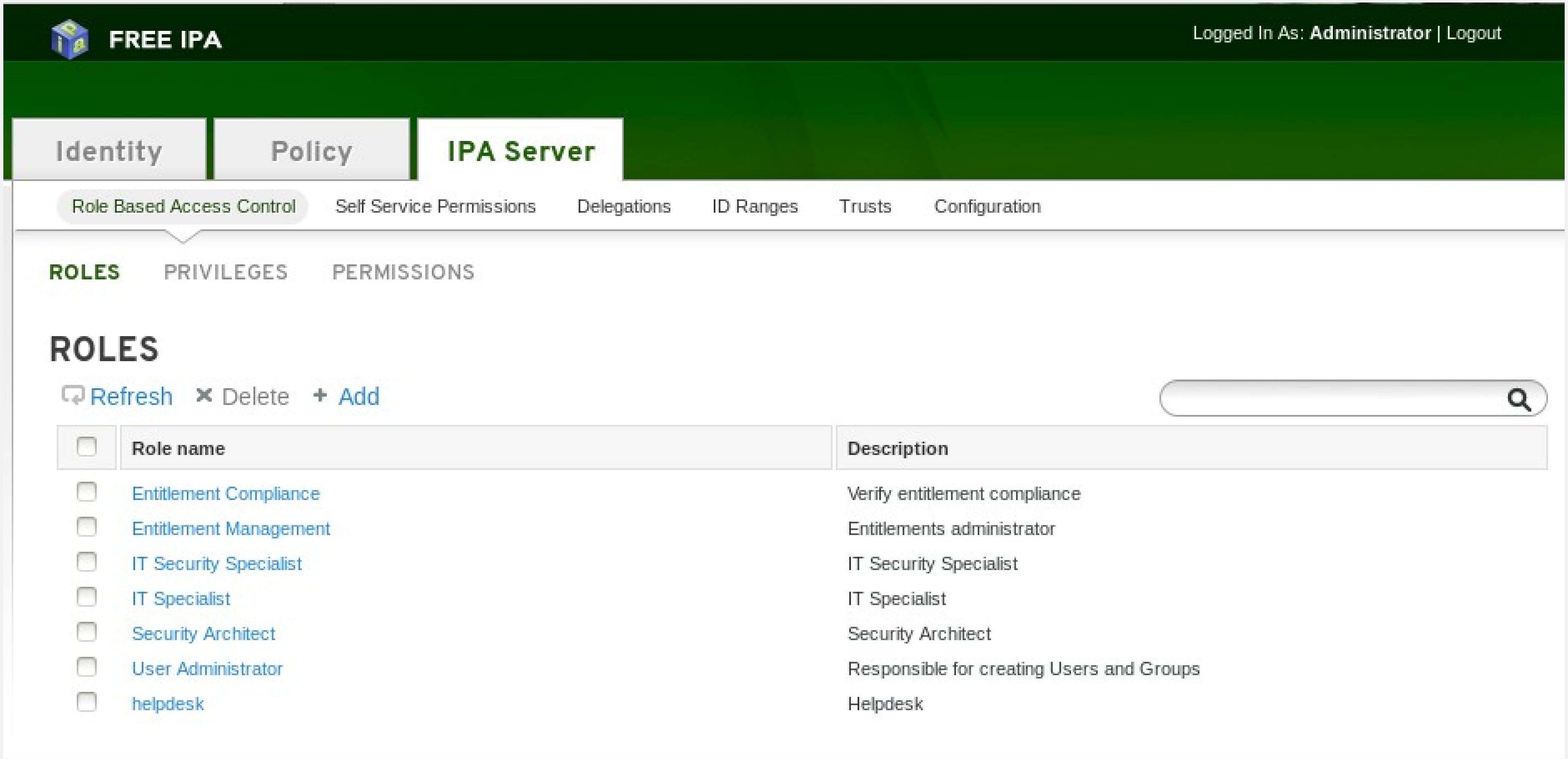
Other services ...

- NTP
  - Time is critical for almost everything
    - Infamous krb5 clock-skew
    - Certificate validity
    - Log correlation
- More ...
  - DHCP
  - Radius
  - Telephony
  - ...

Management Interface

- A complete Management Interface is a fundamental component of an Idm system
- Adding Network APIs makes life easier for 3rd parties. Although CLI tools are often sufficient for small integration tasks.
- Although not mandatory, a graphical interface, such as a Web UI, will make the system usable by a much larger number of people.
  - Helpdesk, Managers, ...

FreeIPA management UI



ROLES

□	Role name	Description
□	Entitlement Compliance	Verify entitlement compliance
□	Entitlement Management	Entitlements administrator
□	IT Security Specialist	IT Security Specialist
□	IT Specialist	IT Specialist
□	Security Architect	Security Architect
□	User Administrator	Responsible for creating Users and Groups
□	helpdesk	Helpdesk

On the client

- A system is as secure as the weakest link
- The client capabilities define what can be done

So ...

  - Classic Linux client
    - nss_ldap & co generally use no authentication
    - Key management is manual , prone to errors
    - Laptops are hard to integrate, poor offline support
    - Access control and sudo rules difficult to manage

An improved client

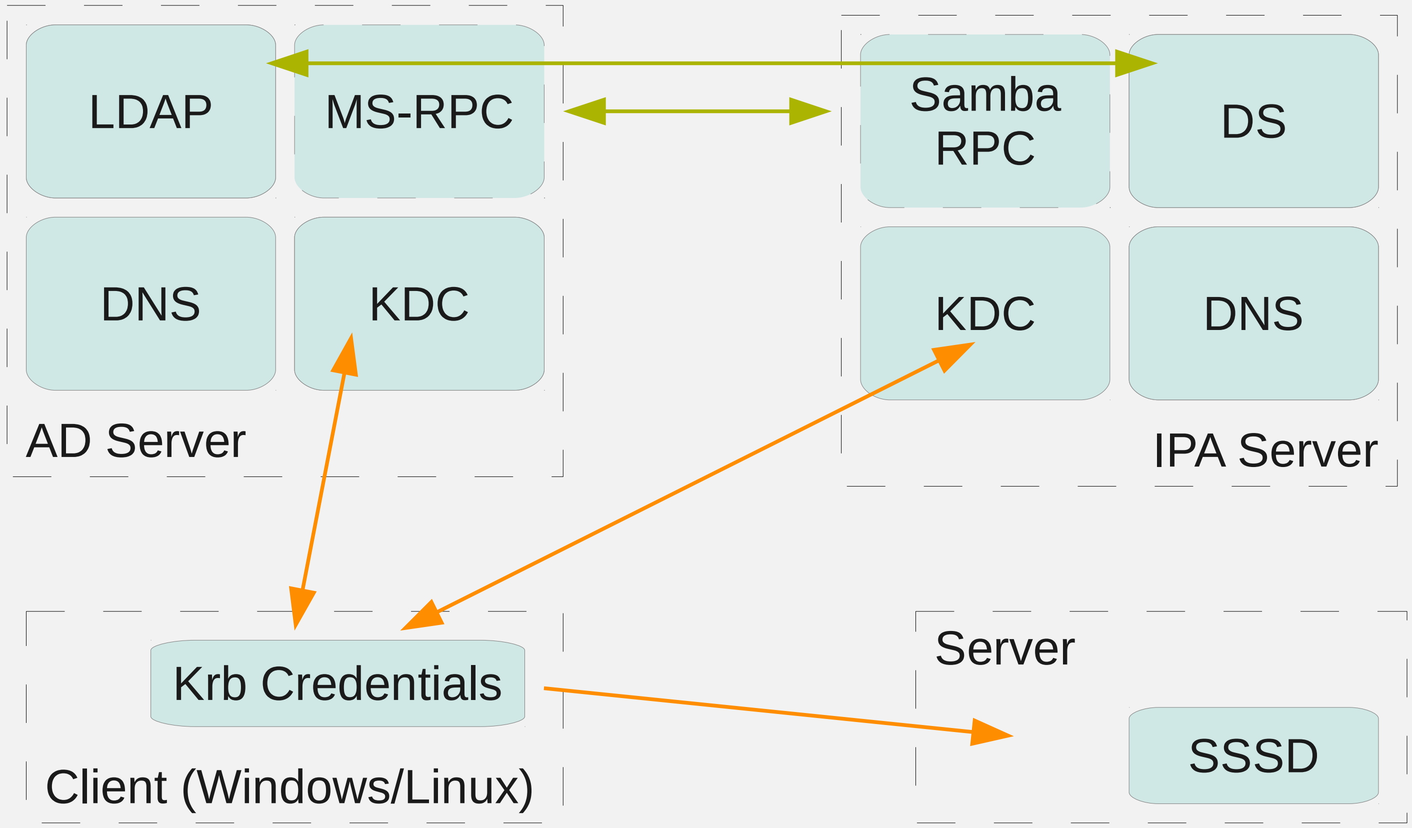
- SSSD was spun off the FreeIPA project
  - Single authenticated server connection
  - Caching of identity and other information
  - Offline authentication
  - HBAC, sudo rules, selinux users, SSH keys
  - Server affinity and DNS updates
- Additional features
  - Certificate renewal (certmonger)
  - Privilege separation (gss-proxy)

Building an Idm system is hard

- It is more of a process than a product
- Installing the bits is just the first step
- An IdM system **must** make things easier to manage
- A management interface is fundamental, even just CLI
- Homegrown may be sufficient, but it is a very big effort
  - Reuse as many components as you can
  - Choose wisely, changing components later is harder
  - Look around you, others have already done this.
    See what they've done and ask yourself why and if the same reasoning applies to your case

Beyond Linux

- FreeIPA has recently added support for creating trust relationships with Active Directory



Questions ?

Thanks to: redhat.

Simo Sorce simo@redhat.com

freeIPA identity | policy | audit http://freeipa.org

Bonus slide

- Acronyms & terminology

  SSO: single Sign On

  2FA: Two-Factor Authentication

  HBAC: Host Based Access Control

  KDC: Key Distribution Center

  Principal: Name of Identities in the Kerberos world

  X509: Encoding standard for SSL certificates

  CA: Certificate Authority, Signs certificates in a PKI system

  PKI: Public Key Infrastructure

- Additional links

  SSSD: http://fedorahosted.org/sssd

  Gss-Proxy: http://fedorahosted/gss-proxy

  Certmonger: https://fedorahosted.org/certmonger/

  Bind-dyndb-ldap: https://fedorahosted.org/bind-dyndb-ldap/

  389 DS: http://port389.org

  Dogtag: http://pki.fedoraproject.org

  MIT Kerberos: http://web.mit.edu/kerberos/