# Next Generation Directory-Based User Management for Cloud Infrastructure

October 20, 2017

LDAPCon, Brussels

# Introduction

Shawn McKinney

- symas Software Architect
- PMC Apache Directory Project
- OpenLDAP™ PROJECT Engineering Team

# Session Objective

Think about how to implement user access controls on machines running in the cloud.

# Session Objective

Not an RBAC Talk

Think about how to implement user access controls on machines running in the cloud.

# Session Agenda

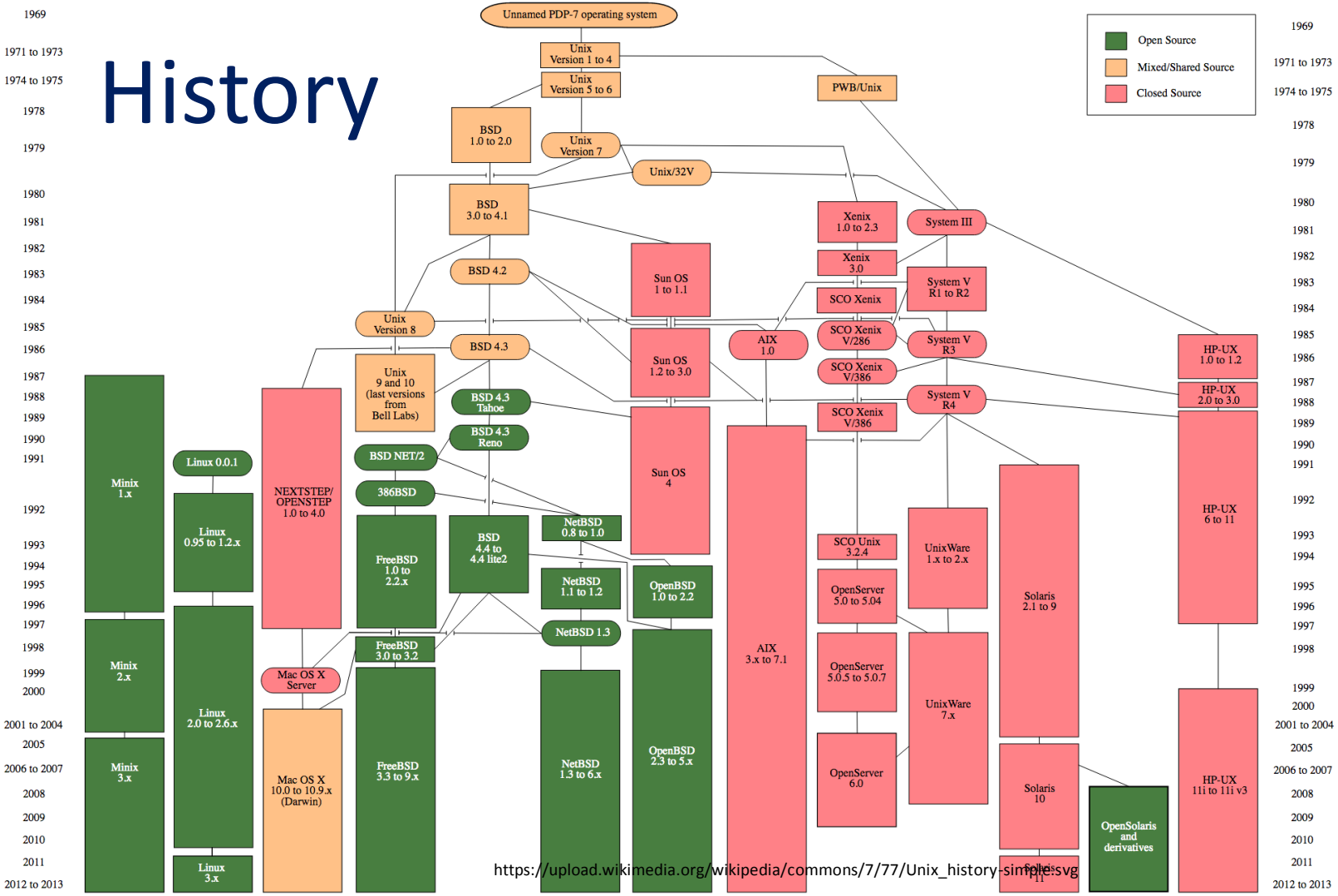
- History of Unix
- Building Blocks
- Security Model
- Data Model
- Solution
- Demo


symas

# History

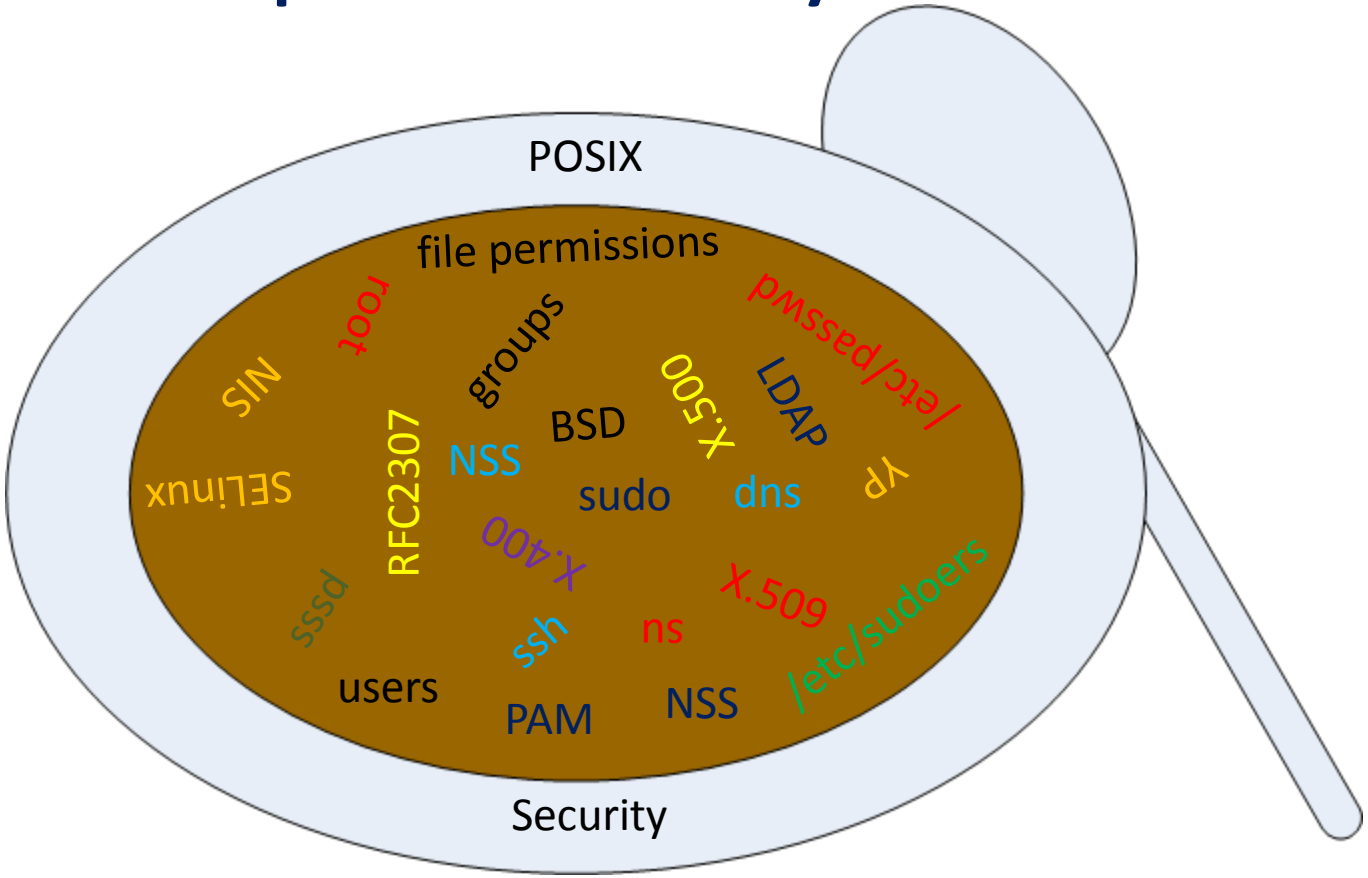
Knowing the path forward means understanding where we've been.

# History

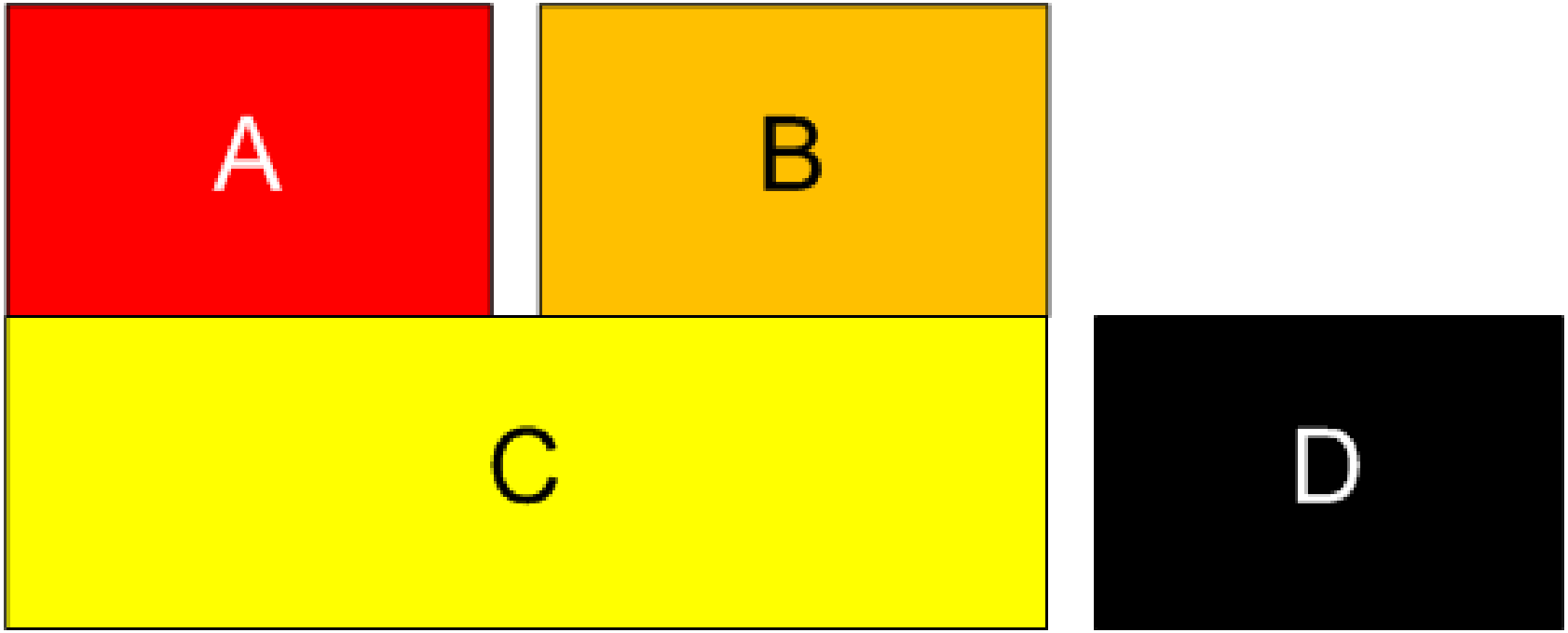



https://upload.wikimedia.org/wikipedia/commons/7/77/Unix_history-simple.svg

# Soup of the Day

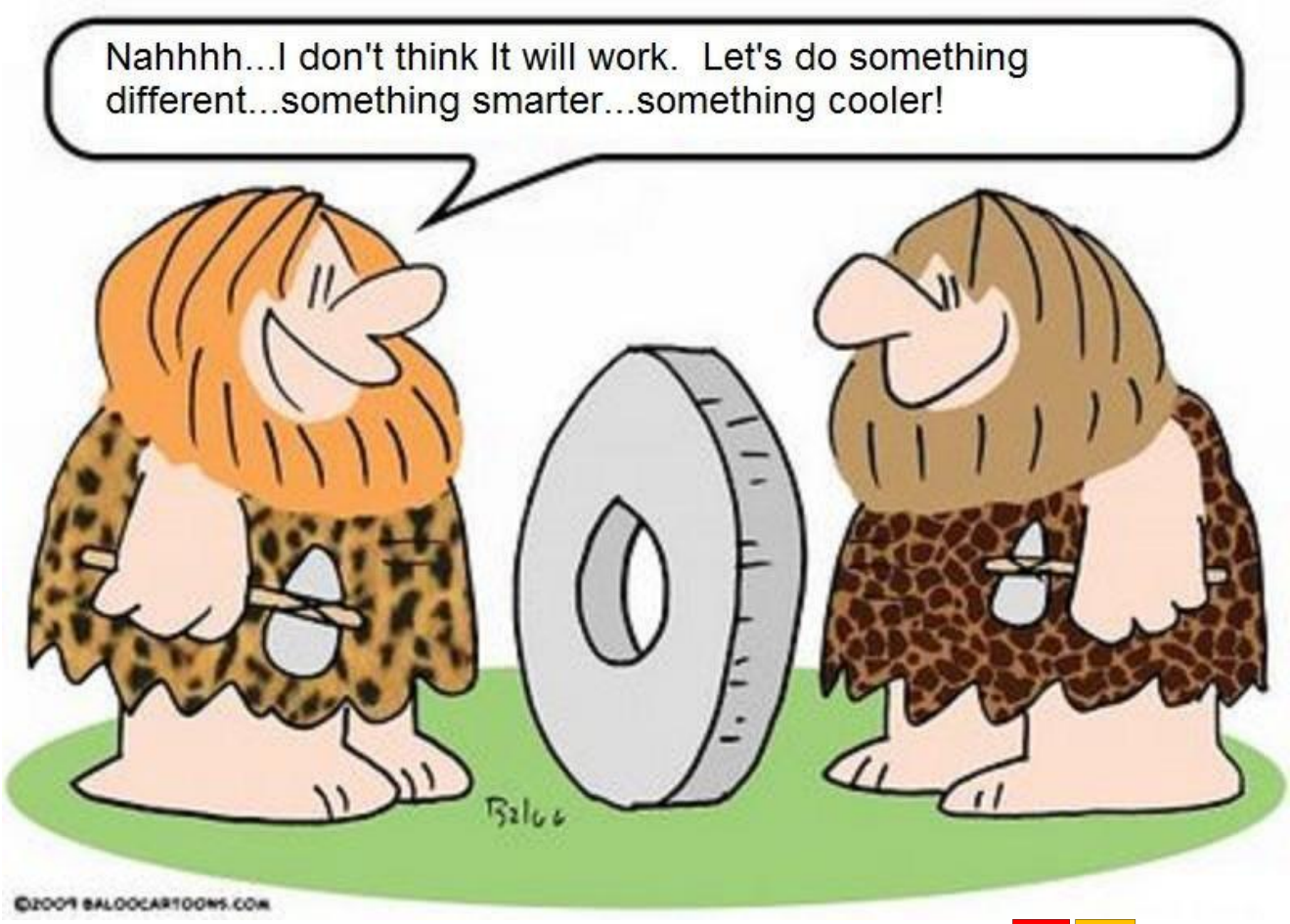


# Building Blocks



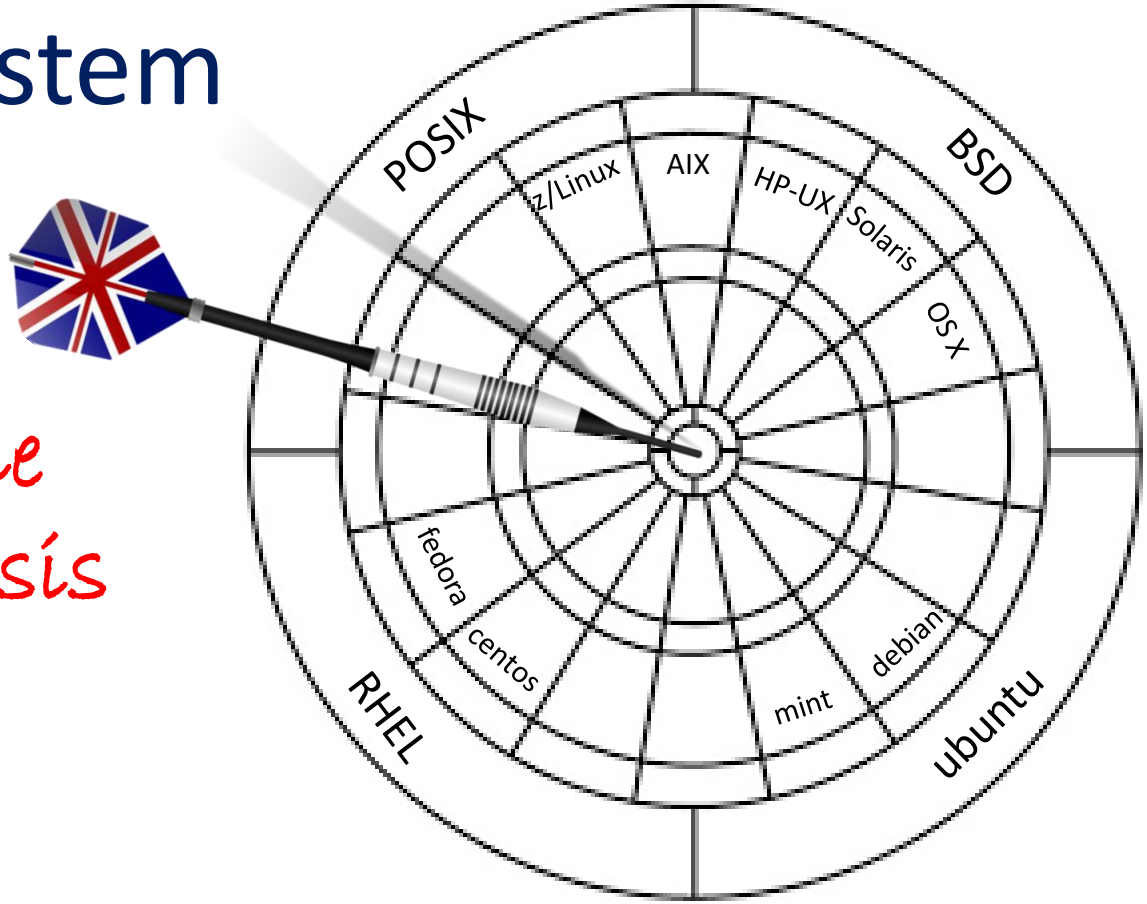
# The Wheel

- Let’s not reinvent



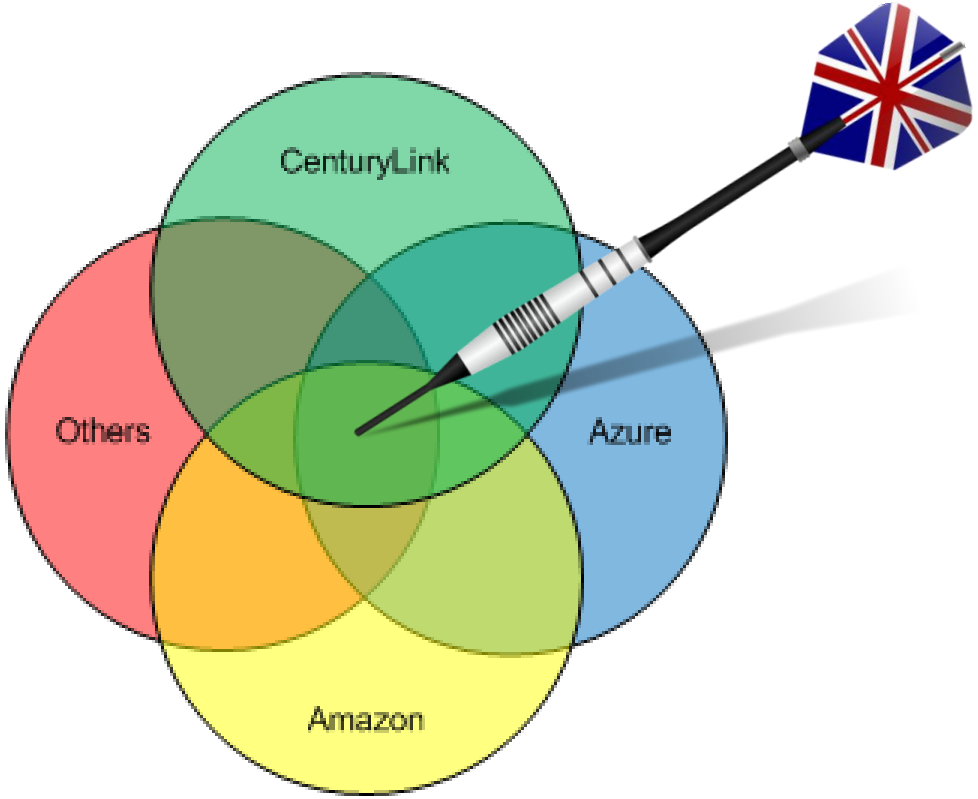
# Operating System

The ldm engine bolts into chassis



# Cloud Infrastructure

runs on the highways



# Basic Building Blocks

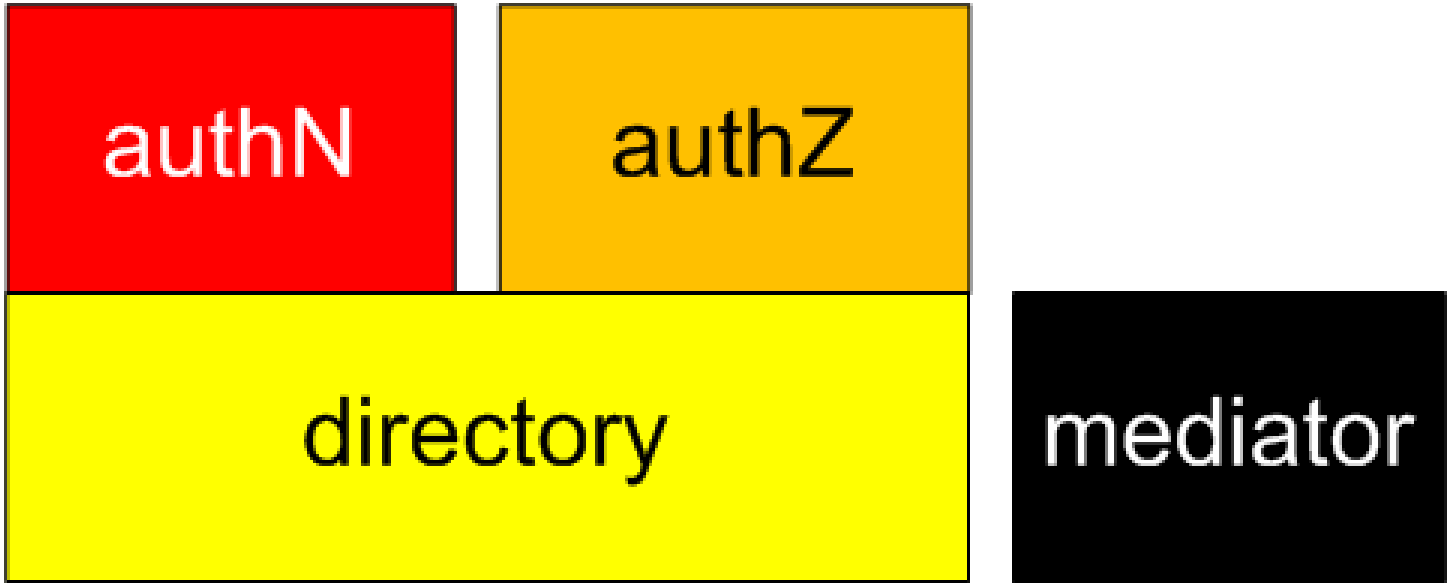
1. POSIX security controls
2. Directory services

Best practices

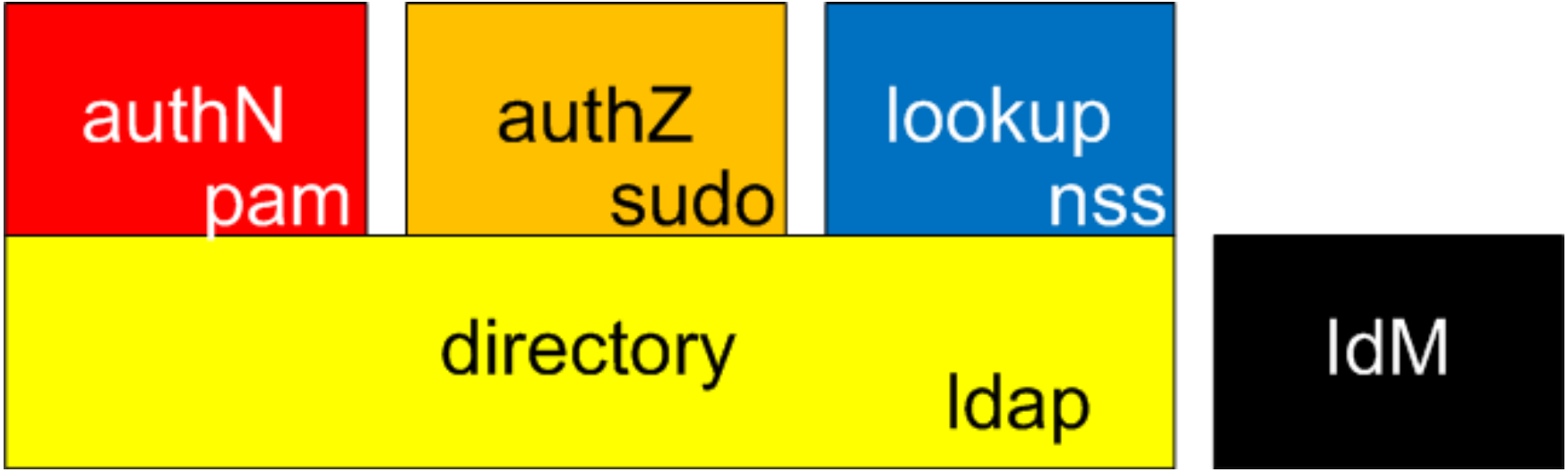
# Advanced Building Blocks

3. Mediation *relatively new practice*

# Building Blocks Conceptual



# Building Block Actual



# Building Blocks - AuthN

pam

## Pluggable authentication module

From Wikipedia, the free encyclopedia

A **pluggable authentication module** (**PAM**) is a mechanism to integrate multiple low-level authentication schemes into a high-level application programming interface (API). It allows programs that rely on authentication to be written independently of the underlying authentication scheme. PAM was first proposed by Sun Microsystems in an Open Software Foundation Request for Comments (RFC) 86.0 dated October 1995. It was adopted as the authentication framework of the Common Desktop Environment. As a stand-alone open-source infrastructure, PAM first appeared in Red Hat Linux 3.0.4 in August 1996. PAM is currently supported in the AIX operating system, DragonFly BSD,[1] FreeBSD, HP-UX, Linux, Mac OS X, NetBSD and Solaris.

# Pluggable Authentication Module

pam

- Authentication
- Coarse-grained Authorization

Just an authN service

# Building Blocks - AuthZ

sudo

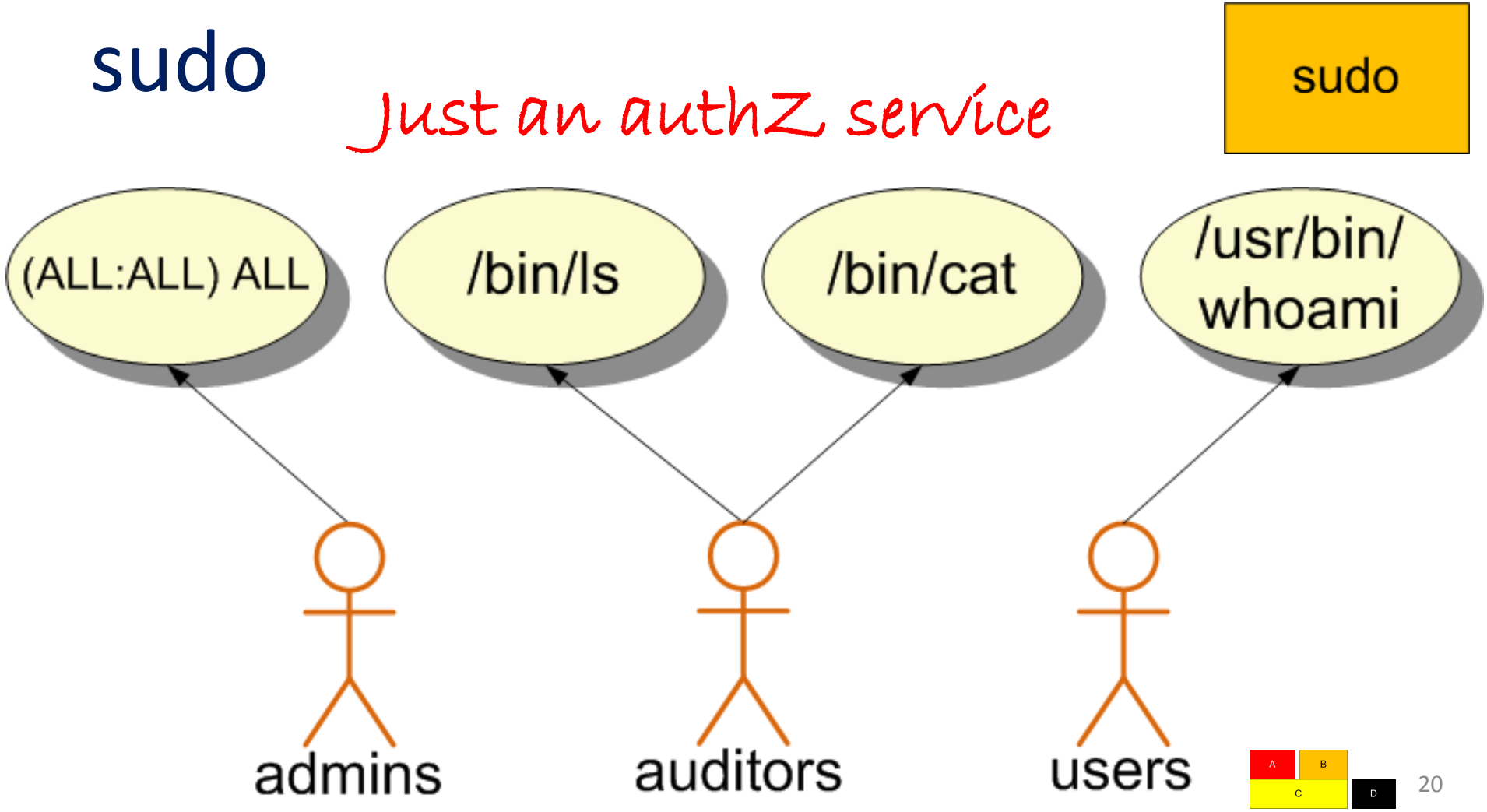
## sudo

From Wikipedia, the free encyclopedia

**sudo** (/ˈsuːduː/[2] or /ˈsuːdoʊ/[2][3]) is a program for Unix-like computer operating systems that allows users to run programs with the security privileges of another user, by default the superuser.[4] It originally stood for "superuser do"[5] as the older versions of sudo were designed to run commands only as the superuser. However, the later versions added support for running commands not only as the superuser but also as other (restricted) users, and thus it is also commonly expanded as "substitute user do".[6][7] Although the latter case reflects its current functionality more accurately, sudo is still often called "superuser do" since it is so often used for administrative tasks.

# sudo

*Just an authZ service*



# Building Blocks – Reporting

nss

## Name Service Switch

From Wikipedia, the free encyclopedia

The **Name Service Switch** (**NSS**) is a facility in Unix-like operating systems that provides a variety of sources for common configuration databases and name resolution mechanisms. These sources include local operating system files (such as `/etc/passwd`, `/etc/group`, and `/etc/hosts`), the Domain Name System (DNS), the Network Information Service (NIS), and LDAP

# Name Service Switch

nss

- Used by unix processes to lookup user and group info

Just a lookup service

# Lightweight Directory Access Protocol

ldap

From Wikipedia, the free encyclopedia

The **Lightweight Directory Access Protocol** (**LDAP**; /ˈɛldæp/) is an open, vendor-neutral, industry standard application protocol for accessing and maintaining distributed directory information services over an Internet Protocol (IP) network.[1] Directory services play an important role in developing intranet and Internet applications by allowing the sharing of information about users, systems, networks, services, and applications throughout the network.[2] As examples, directory services may provide any organized set of records, often with a hierarchical structure, such as a corporate email directory. Similarly, a telephone directory is a list of subscribers with an address and a phone number.

LDAP is specified in a series of Internet Engineering Task Force (IETF) Standard Track publications called Request for Comments (RFCs), using the description language ASN.1. The latest specification is Version 3, published as RFC 4511.

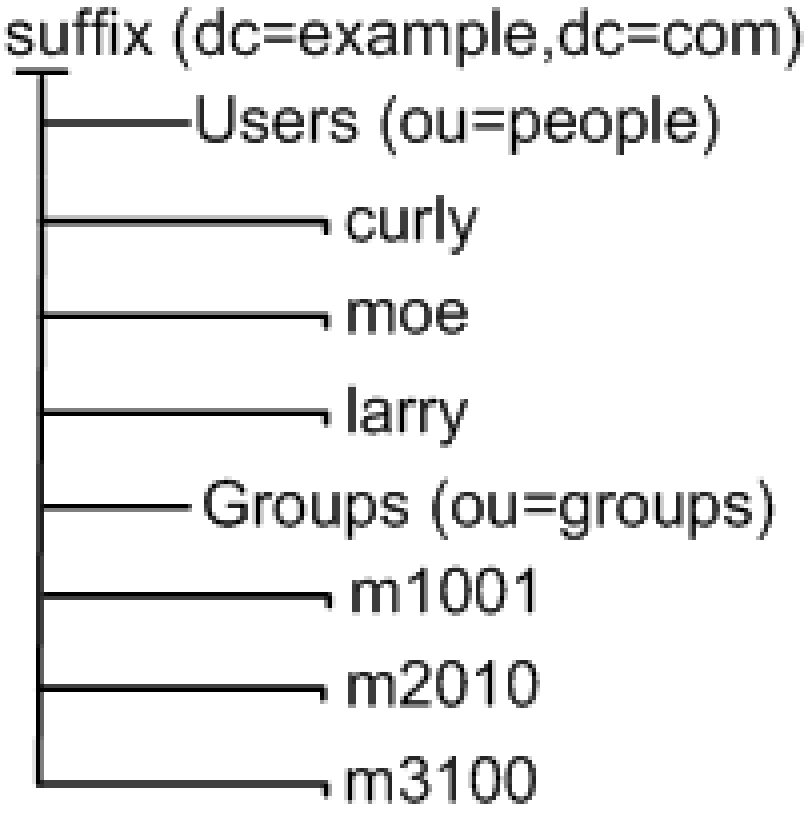
# Building Blocks - LDAP

ldap

Just a

System of record

- Users
- Passwords
- Groups




symas

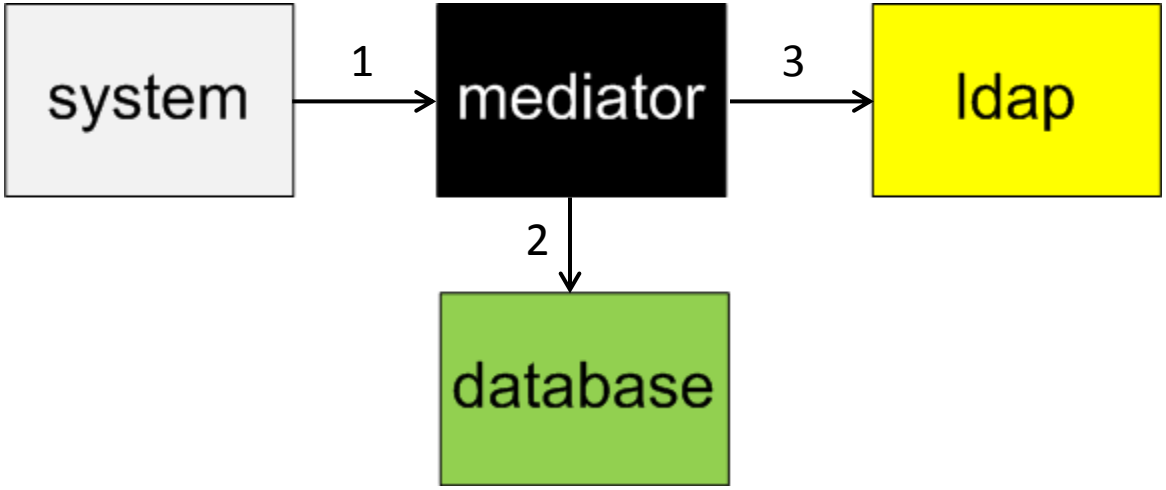
# Building Blocks - Mediator

mediator

- Keeps things in synch between the machines and LDAP as things change.

# Mediator

1. Machine added to network, notifies mediator
2. Based on policies stored in DB
3. Updates ldap accordingly



# Mediator == IdM

1. Provisioning
2. Parameterized Roles
3. Organizational Controls
4. Self-service
5. Approvals
6. Workflow

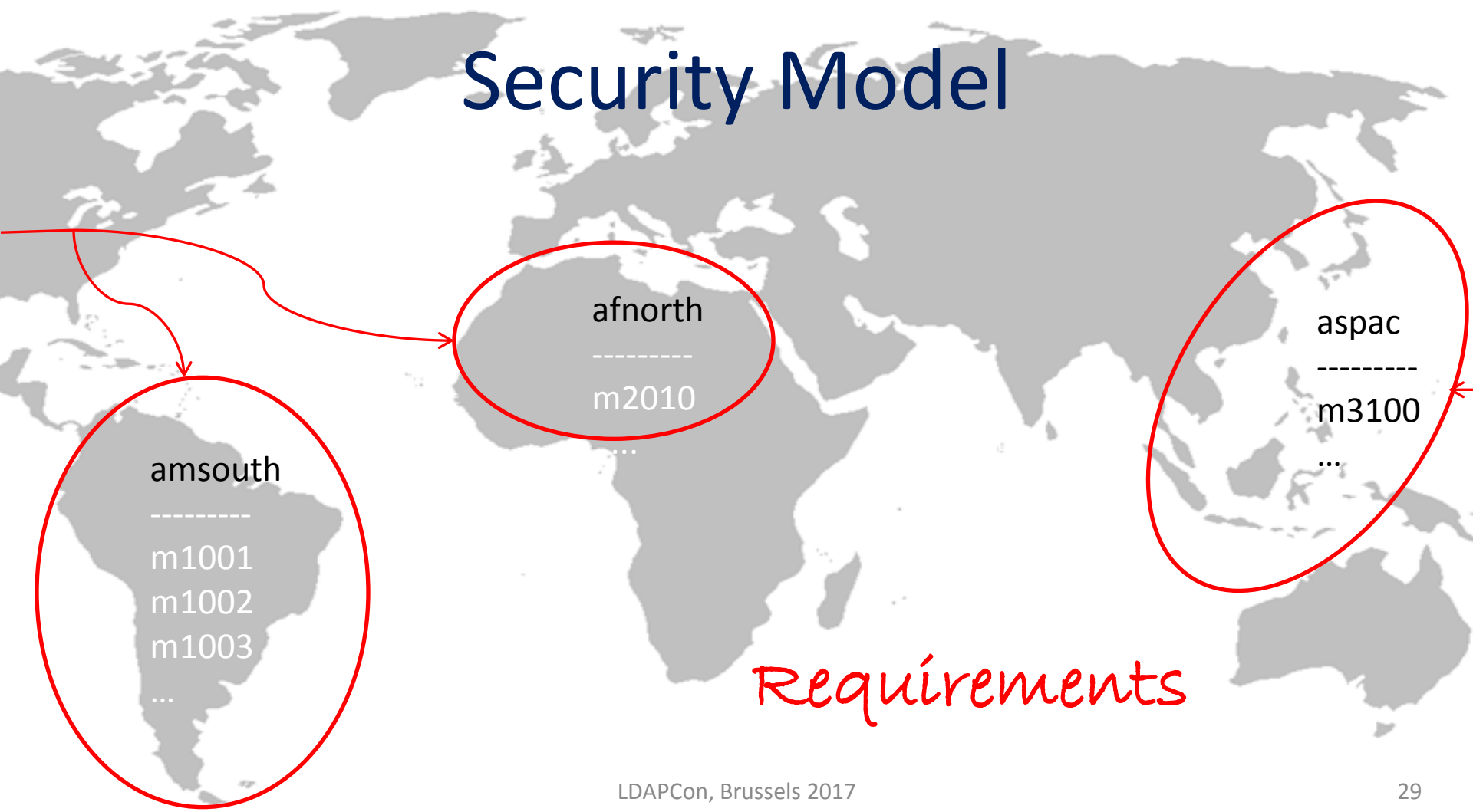
Requirements

# Open Source IdM Products

1. midPoint
2. Apache Syncope
3. OpenIDM
4. WSO2 Identity Server

lacking standards

# Security Model



# Three Kinds of Security Checks

PAM

1. Authentication with LDAP
2. Coarse-grained authZ - memberOf target machine
   - (i.e. LDAP group name == hostname)

sudo

3. Medium-grained authZ. memberOf at least one:
   - Admin - root access
   - User - typical user access
   - Auditor - read-only access to entire machine.

# Three Types of Control Groups

Mediator 1. Machine Sets

PAM 2. Machines

3. Security Roles

sudo 4. sudo Roles

# 1. Machine Sets

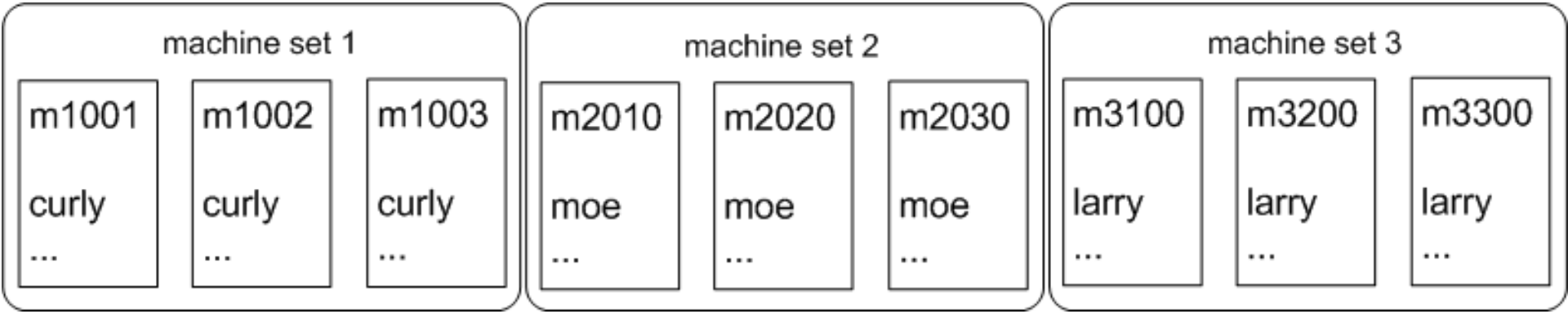
m3set
---
m3100
m3200
m3300
...

m2set
---
m2010
m2020
m2030
...

m1set
---
m1001
m1002
m1003
...

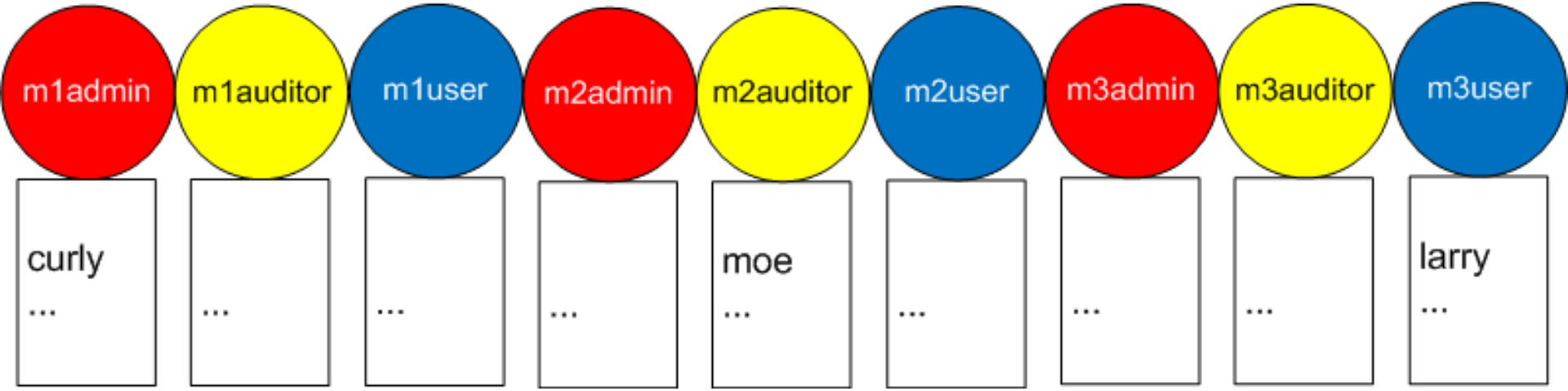
Used by mediator to compute policies

# 2. Machines




Used by PAM

# 3. Security Roles




Sudo needs this

# 4. sudo Roles


Sudo needs this

# Policy Combiner

User, role and machine set

m3set
---
m3100
m3200
m3300
...


user

m2set
---
m2010
m2020
m2030
...

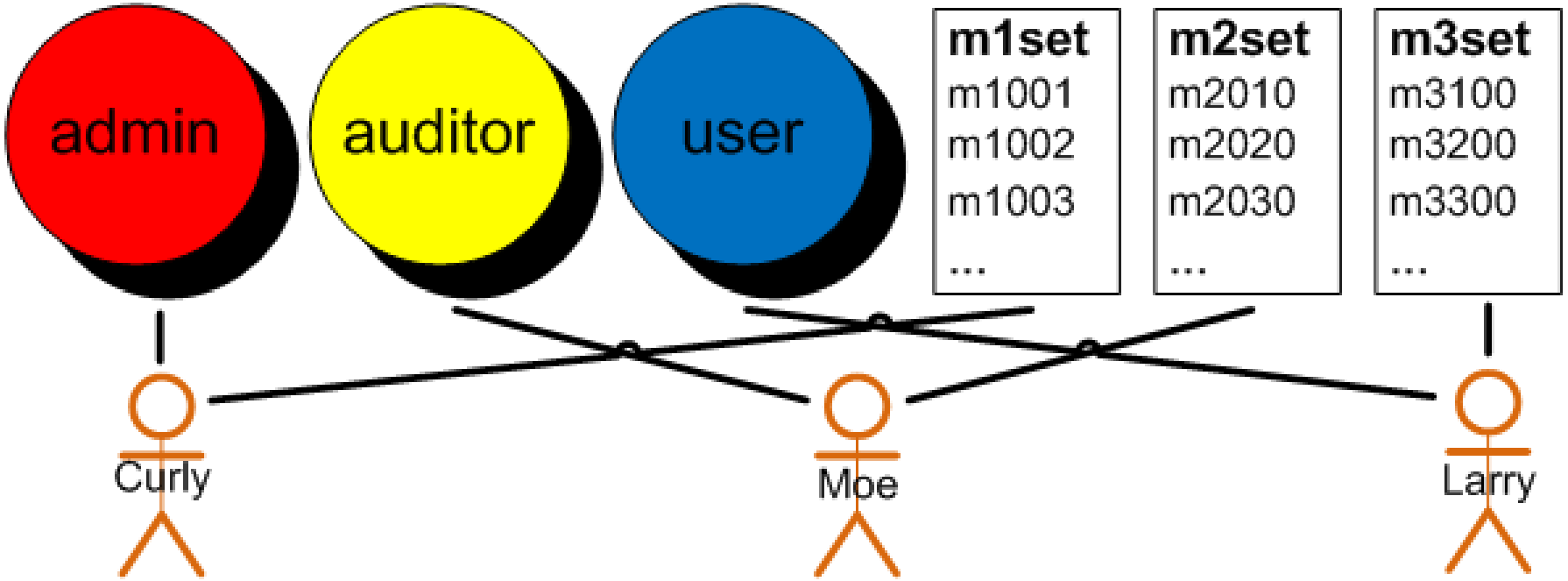

auditor

m1set
---
m1001
m1002
m1003
...

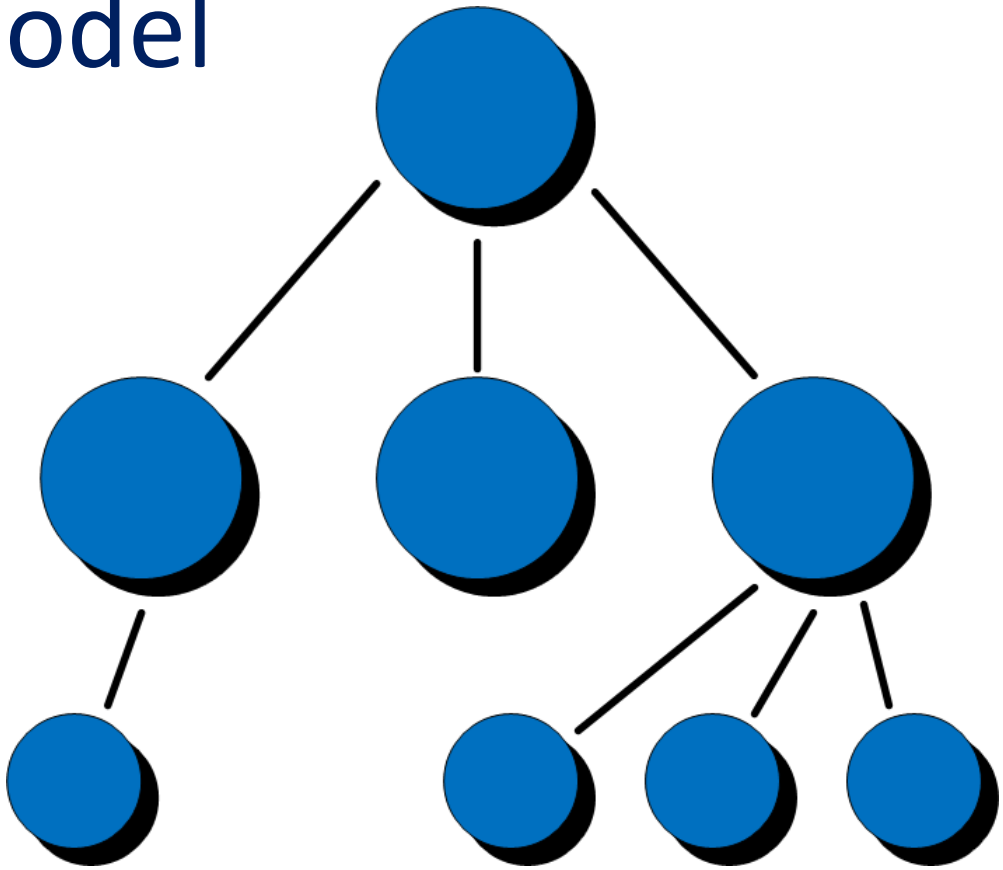

admin

The mediator can do this

# Pick Two


admin
auditor
user
m1set
m1001
m1002
m1003
...
m2set
m2010
m2020
m2030
...
m3set
m3100
m3200
m3300
...
Curly
Moe
Larry

# Data Model



# LDAP Data Model

Employ standard object schemas

1. RFC2307bis
   - posixAccount
   - posixGroup
2. sudoRole
3. groupOfNames

# RFC2307bis

Network Working Group — L. Howard
Internet-Draft — PADL Software
Obsoletes: 2307 (if approved) — H. Chu, Ed.
Intended status: Informational — Symas Corp.
Internet-Draft — LDAP NameService Schema — August 2009

**An Approach for Using LDAP as a Network Information Service**
**draft-howard-rfc2307bis-02.txt**

This document describes a mechanism for mapping entities related to TCP/IP and the UNIX system [UNIX] into [X.500] entries so that they may be resolved with the Lightweight Directory Access Protocol [RFC4511]. A set of attribute types and object classes are proposed, along with specific guidelines for interpreting them. The intention is to assist the deployment of LDAP as an organizational nameservice. No proposed solutions are intended as standards for the Internet. Rather, it is hoped that a general consensus will emerge as to the appropriate solution to such problems, leading eventually to the adoption of standards. The proposed mechanism has already been implemented with some success.

# Covered here before

- LDAPCon 2015, Edinburg
- DBIS: Directory-Based Information Services
- Mark R. Bannister
- link to slides
- link to paper

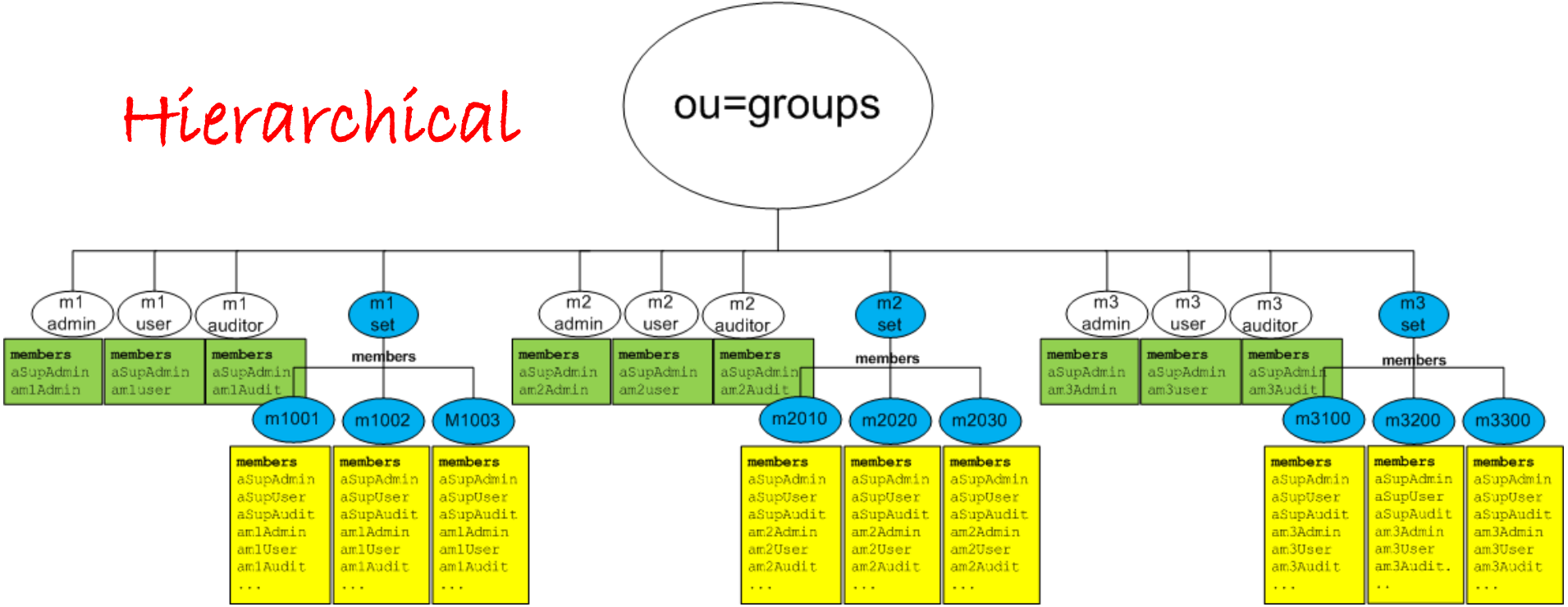
# Use RFC2307bis LDAP Schema

```
( 1.3.6.1.1.1.2.0 NAME 'posixAccount' SUP top AUXILIARY
    DESC 'Abstraction of an account with POSIX attributes'
    MUST ( cn $ uid $ uidNumber $ gidNumber $ homeDirectory )
    MAY ( authPassword $ userPassword $ loginShell $ gecos $
          description ) )
```

```
( 1.3.6.1.1.1.2.2 NAME 'posixGroup' SUP top AUXILIARY
    DESC 'Abstraction of a group of accounts'
    MUST gidNumber
    MAY ( authPassword $ userPassword $ memberUid $
          description ) )
```

# LDAP Data Model

Hierarchical



# Machine Set M1

```
dn: cn=m1set, ou=Groups, ...
description: Machine Set 1
member: cn=m1001,...
member: cn=m1002,...
member: cn=m1003,...
…
```

# Machine M1001

```
dn: cn=m1001, ou=Groups,…
objectClass: posixGroup
description: Machine Group M1001
member: uid=curly,ou=People,…
member: uid=frank,ou=People,…
member: uid=marla,ou=People,…
…
```

# Security Role M1Admin

```
dn: cn=m1admin, ou=Groups, ...
objectClass: posixGroup
description: Admin Machine Set 1
cn: m1admin
member: uid=curly,ou=People,...
member: uid=frank,ou=People,...
member: uid=marla,ou=People,...
…
```

# sudo LDAP Schema

```
objectclass ( 1.3.6.1.4.1.15953.9.2.1
   NAME 'sudoRole' SUP top STRUCTURAL
   DESC 'Sudoer Entries'
   MUST ( cn )
   MAY ( sudoUser $ sudoHost $ sudoCommand
  $ sudoRunAs $ sudoRunAsUser
  $ sudoRunAsGroup $ sudoOption
  $ sudoNotBefore $ sudoNotAfter
  $ sudoOrder $ description )
)
```

# sudo M1Admin

```
dn: cn=admin access to
  m1,ou=sudo,dc=example,dc=com
objectClass: sudoRole
cn: admin access to m1
sudoUser: %m1admin
sudoHost: m1001
sudoHost: m1002
sudoHost: m1003
sudoHost: m1004
```

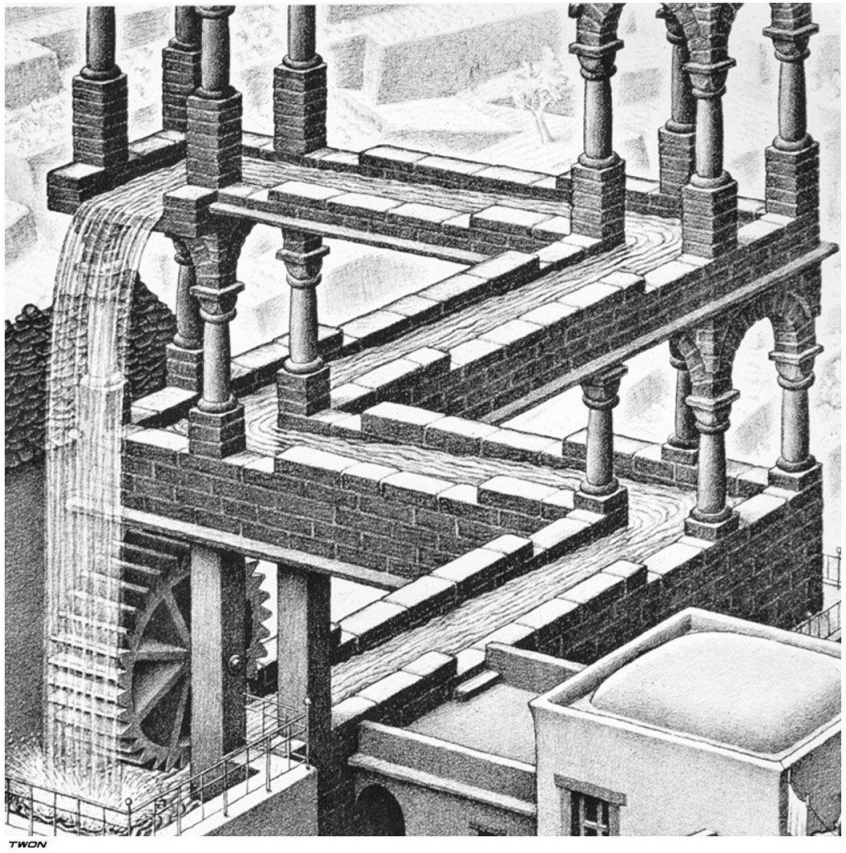
# Data Mapping

How do we do this?

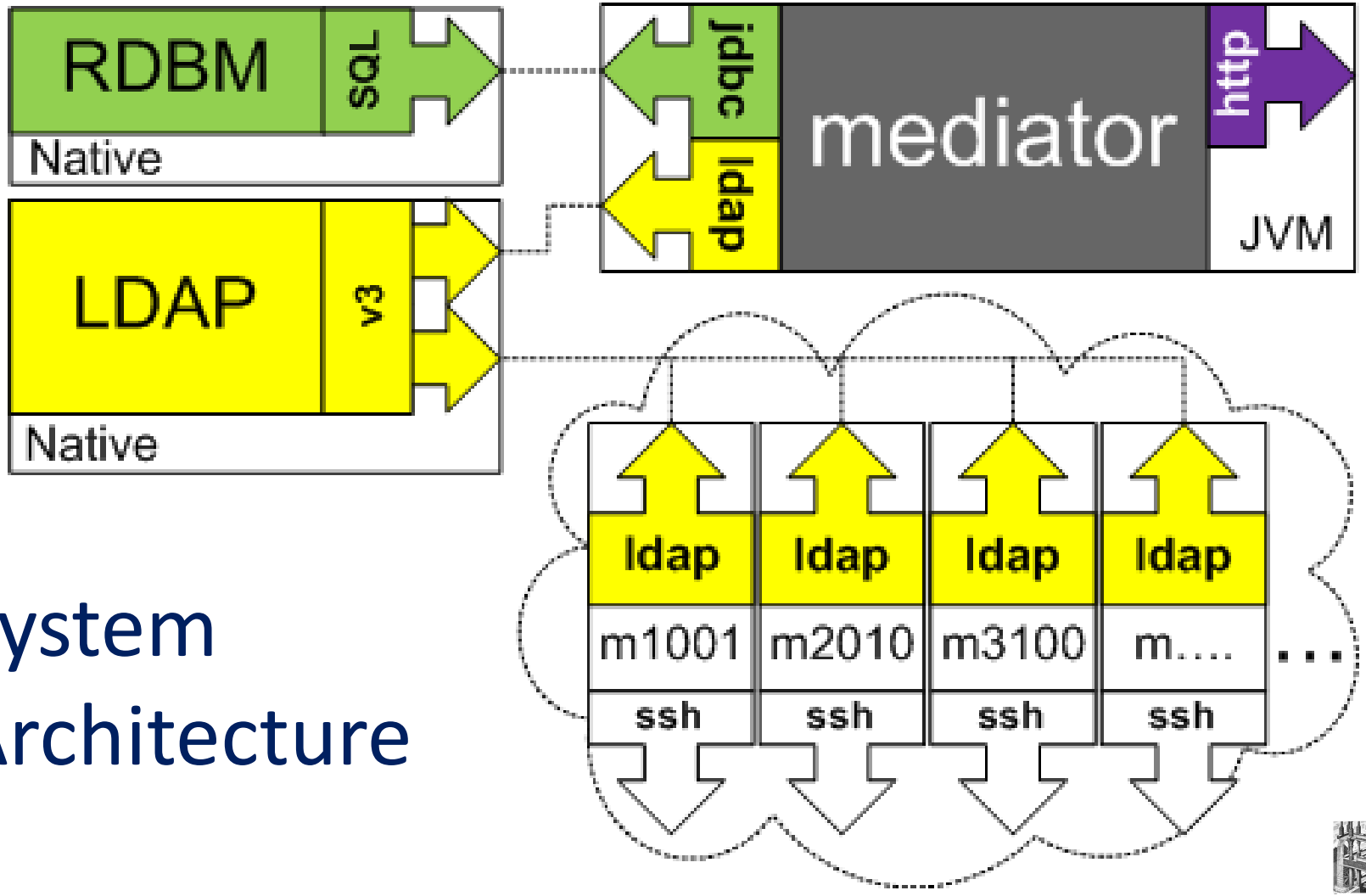
| midPoint | LDAP |
|---|---|
| Services: | Machine Groups: |
| 1. m1001 | 1. m1001: curly, ... |
| 2. m2010 | 2. m2010: moe, ... |
| 3. m3100 | 3. m3100: larry, ... |
| Organizations | Machine Set Groups: |
| 1. m1 | 1. m1set: m1001, ... |
| 2. m2 | 2. m2set: m2010, ... |
| 3. m3 | 3. m3set: m3100, ... |
| Roles: | Security 'Roles': |
| 1. admin | 1. m1admin: curly, ... |
| 2. auditor | 2. m2auditor: moe, ... |
| 3. user | 3. m3user: larry, ... |
| Users: | Users: |
| 1. curly: m1, admin | 1. curly |
| 2. moe: m2, auditor | 2. moe |
| 3. larry: m3, user | 3. larry |

symas

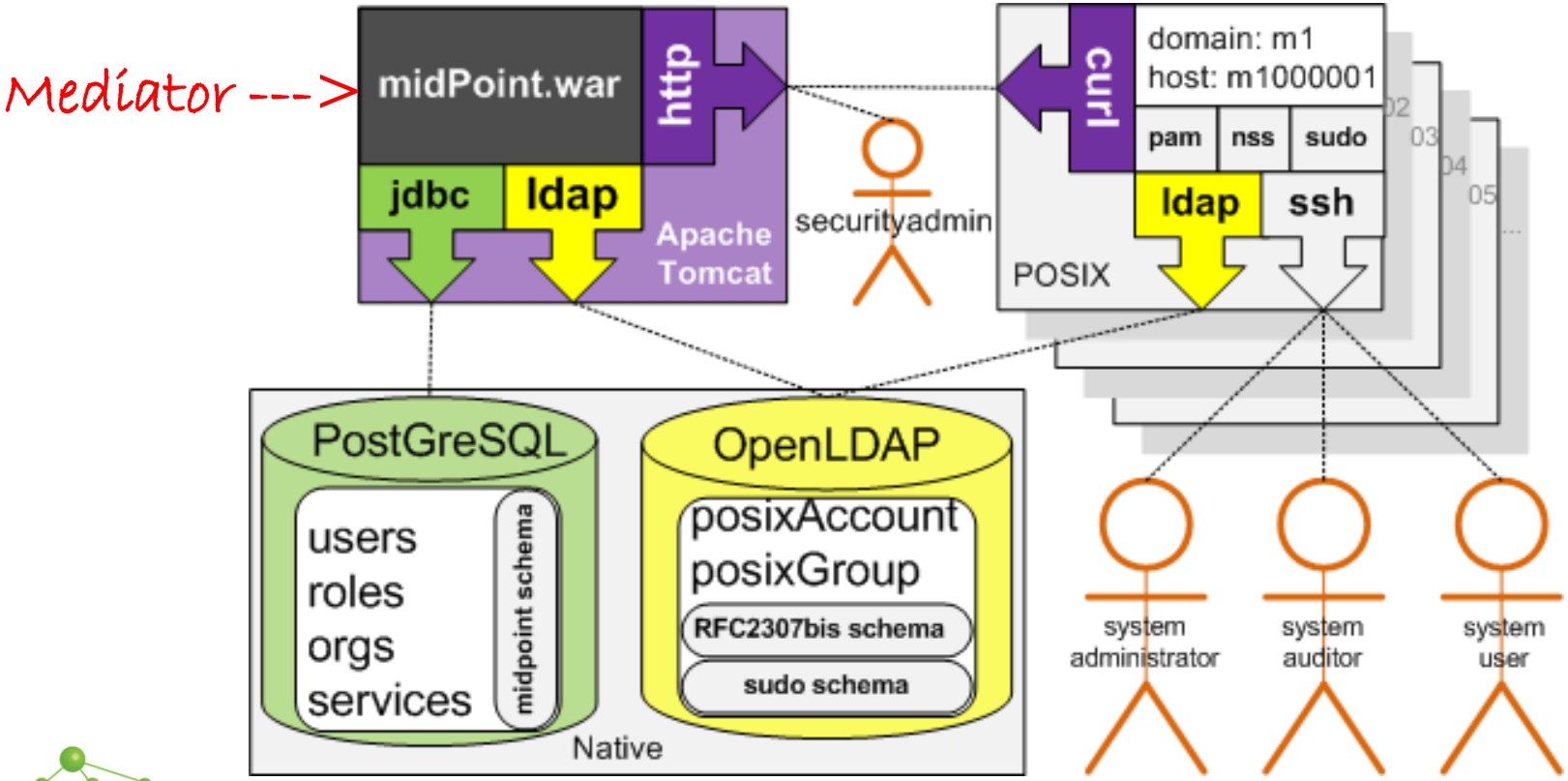
# Solution



# System Architecture



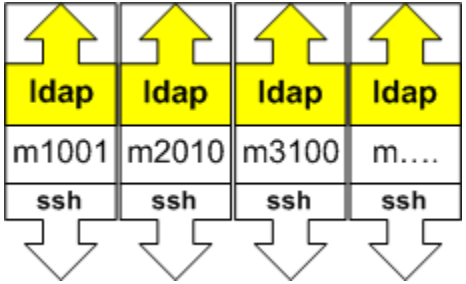
# High-level Design

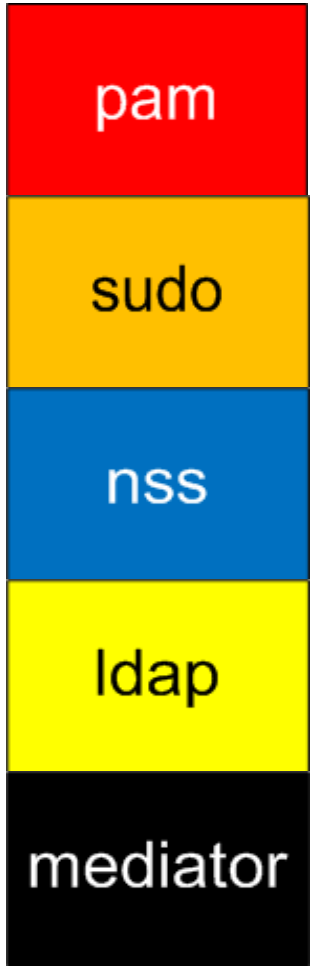


# Client Machines

Script runs during machine instantiation:

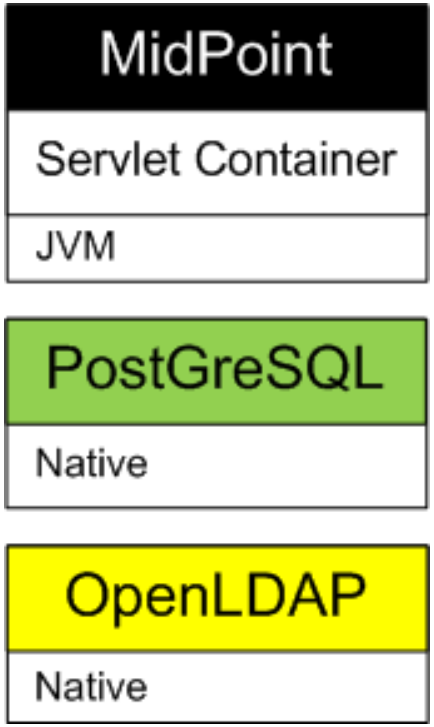
1. Binds PAM, sudo and NSS into the LDAP server.
2. Calls mediator to add or remove from machine set.

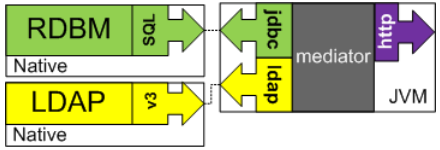


# IdM ‘Server’

1. MidPoint - mediator
   - html & http admin services
2. PostGreSQL – master database
   - users, roles, orgs, svcs
3. OpenLDAP – security database
   - users, groups
   - posixAccount, posixGroup



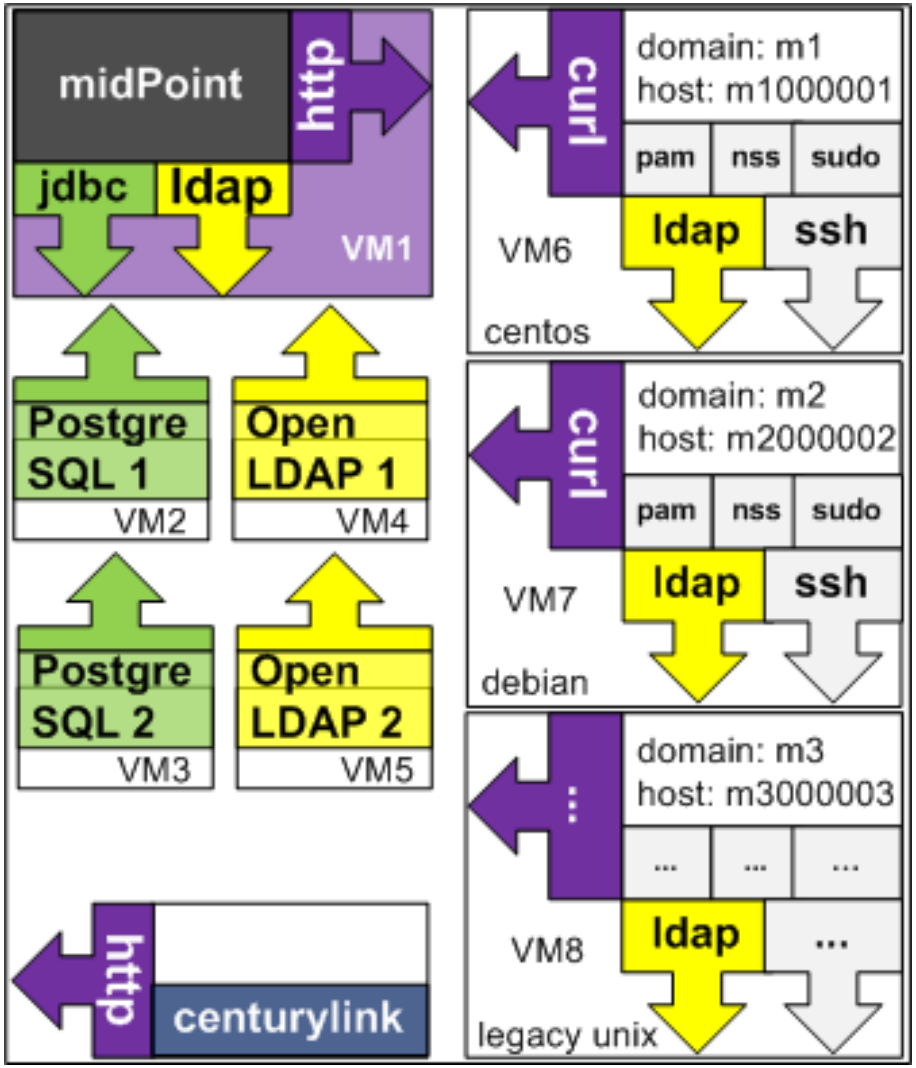
# Deployment

IdM machine#1 --->

IdM machine#2 --->

IdM machine#3 --->

hypervisor --->




<-dev machine#1

<-test machine#2

<-prod machine#3

...

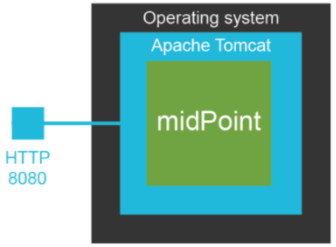
# Demo Scenario

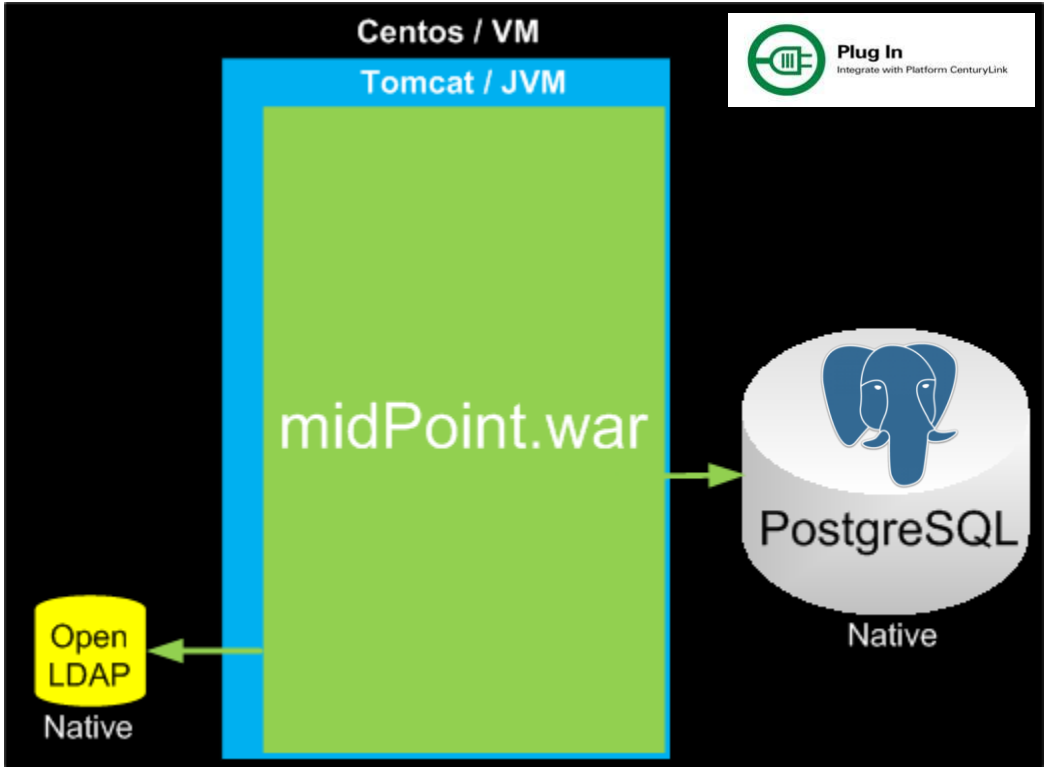
Manage users and unix machines running in the cloud.

# Demo User to Role to Machine

| | <----- Set 1------> | | | <-------Set 2 ------> | | | <----- Set 3 -----> | | |
|---|---|---|---|---|---|---|---|---|---|
| **User-Role-Machine** | **m1001** | **m1002** | **m1003** | **m2010** | **m2020** | **m2030** | **m3100** | **m3200** | **m3300** |
| **Curly** | Admin | Admin | Admin | | | | | | |
| **Moe** | | | | Auditor | Auditor | Auditor | | | |
| **Larry** | | | | | | | User | User | User |

# Demo Environment



# Wrap-up

- Questions
- Next Steps

# Let’s Go

Twitter: @shawnmckinney

Website: https://symas.com

Email: smckinney@symas.com

Blog: https://iamfortress.net

Project: https://directory.apache.org/fortress