

# Prometheus: LDAP Based Account Provisioning

---

Simon Wilkinson <[simon@sxw.org.uk](mailto:simon@sxw.org.uk)>  
School of Informatics, University of Edinburgh

# Introduction

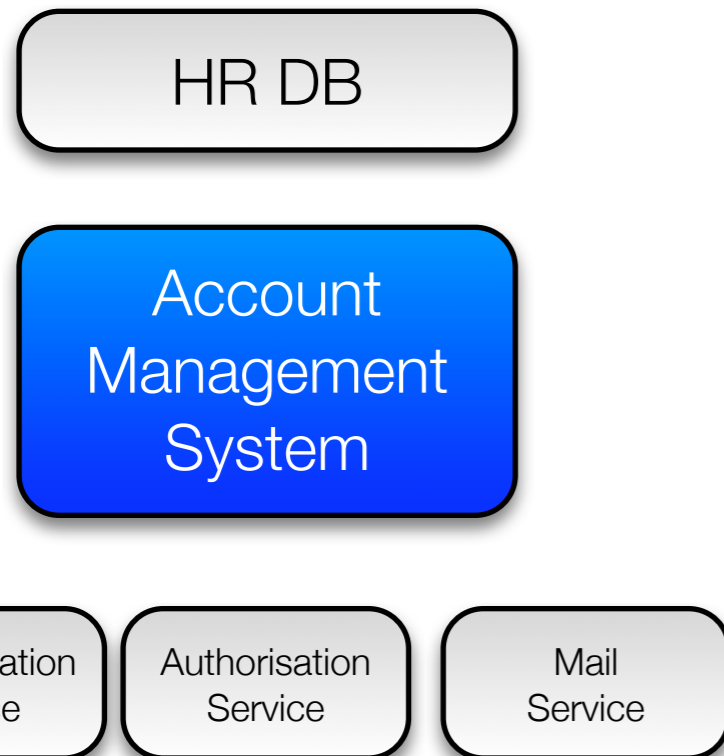
---

- Background & Requirements
  - Decentralisation
  - Account types
  - Access and Identity Models
  - Auditing
- Our Implementation
  - Data Model
  - Architecture
  - Technologies
- Conclusions

# Watchu talkin' 'bout, Willis?

---

- What do we mean by Account/Identity\* Management?



- Layer between corporate database, and your services
- Ensures that changes propogate down (*Identity Lifecycle*)
- Ensures that data in all locations matches the corporate copy

\*Insert buzzword of choice

# Distribution and decentralisation

---

- Historically, an organisation would have either
  - A central account database used by every service
  - Every service using its own database
  - (or, a mixture of the above)
- Both are untenable in a distributed, decentralised world

# Account Database per Service

---

- No idea who your 'users' are
- Deletion/disabling is impossible
- Management is fragmented
- Delegated services are difficult

# Centralised database for all services

---

- Delegated services are impossible
- Every service must interoperate with the central database
- Central database must know about every service.
- Scalability is poor, especially for systems without a strong centre

# Decentralisation

---

- Bringing up a service at the periphery shouldn't require action at the centre
- Users have to be able to deploy their own services, which use the central account management system
- Users must be able to manage their own access control, both for these and for centrally managed services

# Account spectra

---

- Traditionally have 'accounts' for institutional users
- Gulf separates those from 'visitors', and again from web application accounts.

**STAFF**

**VISITOR**

**WEB**

- These boundaries are increasingly archaic
- Accounts must be able to slide in position on the spectrum



# Role Based Access Control

---

- Traditionally user access has been controlled via user lists or groups.
- Role based access control adds additional flexibility
- A user has one or more roles, which describes functions they perform
  - Staff
  - Student
  - Head of Department
  - System administrator

# Entitlements

---

- Each role confers upon the user a set of *entitlements*
- Entitlements determine what they can do on the system
  - Log in to webmail
  - Log in to the compute cluster
  - Access the finance system
  - Edit the DNS
- For flexibility, we also allow roles to include other roles
  - For example, every **student** is a **person**

# Identity Modelling

---

- Traditionally, all of a users entitlements are associated with a single identity
- The user authenticates to that identity, and then gets all of the access granted to that identity
- Doesn't solve
  - “I want a password that can just access my webmail”
  - “I want a key that will let a process just write to this directory”
  - “I shouldn't have admin permissions unless necessary”
  - “I should need a smartcard to login as an administrator”

# Facets of Identity

---

- Split identity into multiple facets (or instances)
- Each instance has a subset of the base identity's entitlements
- Some instances may have additional entitlements
- Let users create instances as required, and distribute entitlements between them

# Assurance

---

- Different levels of access require different levels of identity verification
  - Passwords
  - One-time passwords
  - Hardware tokens
- Important to be able to keep this different levels separate on the system (*can't all share a Unix UID, if they're shell sessions*)

# Accounts

---

- Ultimately, an account is a property of a particular system
- Different systems may have different account attributes (uids, gids, shells, etc)
- Not all identities will have accounts
- Some accounts may have

# Entities

---

- Machines access services too, and their accounts must be managed.
  - Not an Account Management System
  - Not an Identity Management System
  - An Entity Management System?

# Distribution

---

- A central system doesn't scale, either physically, or in terms of administrator effort
- Each service must be responsible for managing its local database based on the contents of the central system
- Active, regular synchronisation is vital
- Must be local to the service, rather than pushed from the centre



# Auditing

---

- Important to regularly report on the integrity of the system
- Audit runs identify anomalies and inconsistencies between databases
- Audit reports can highlight software bugs, and operator errors
- Provided as a tool for service administrators

# Delegation

---

- Users should be able to manage their own identity
- Users should be able to own groups and entitlements
- Users should be able to bring up services which use the central system
- A service administrator is just another user

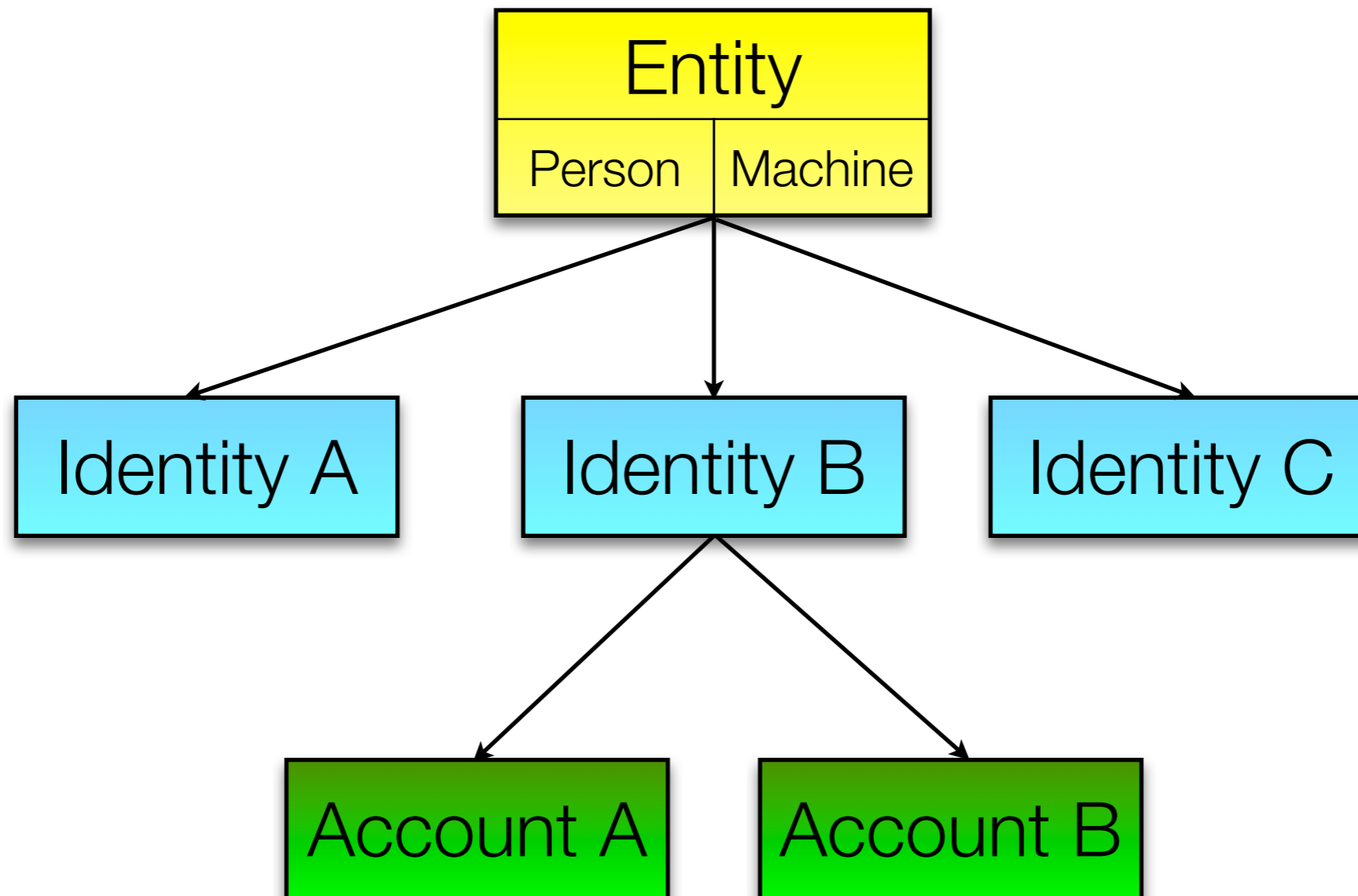
# Introducing Prometheus

---

- The School of Informatics's new account management system
- Currently named *Prometheus*
- Designed to address all of the previously discussed issues
- Very much a work in progress!

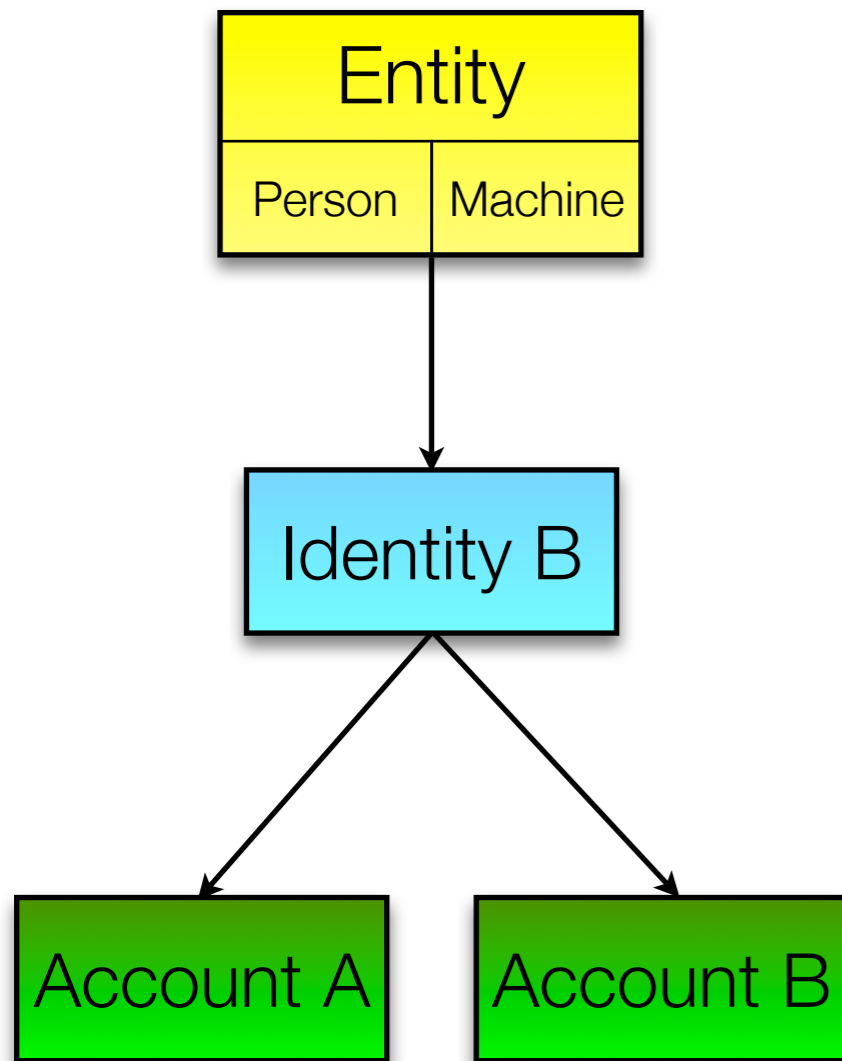
# The Entity Data Model

---



# What goes where ...

---



- Entities contain real-world information, plus overall role data
- Identities contain instance specific data, and authentication details
- Accounts contain OS specific data (uid, gid and the like)

# Role and Entitlement data model

---

- A role contains
  - other roles
  - entitlements
  - negated entitlements
- An entity has both roles and explicit entitlements
- An identity may have any roles and entitlements owned by its parent entity (and must have any negated entries)
- An identity may have additional roles and entitlements

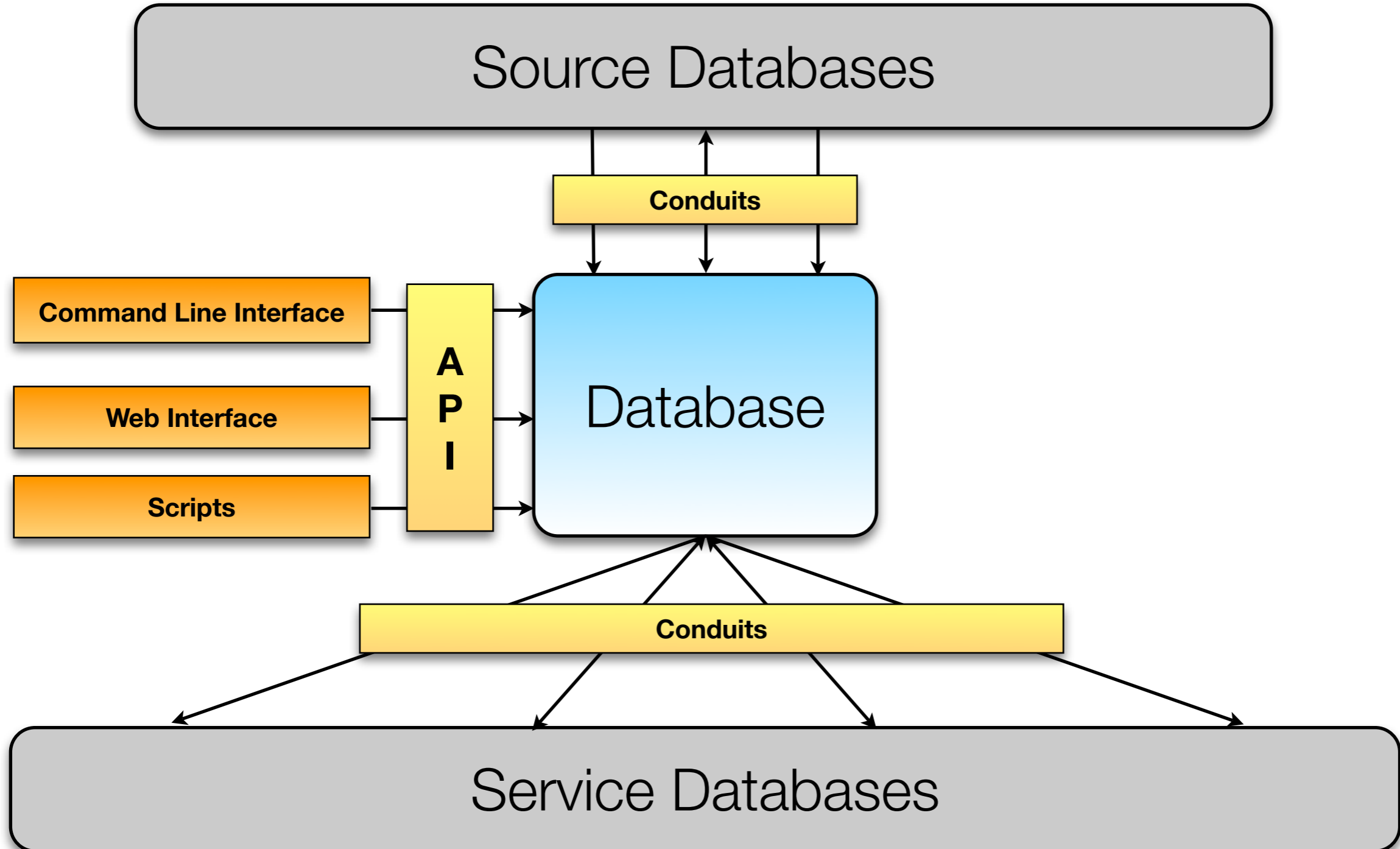
# Role and Entitlement Ownership

---

- In order to achieve *delegation* we have to have an ownership model for roles and entitlements
- An owner may grant that object to another role or user
- An owner may *resign* (remove) any object that they've been granted, but cannot then restore it.
- Owners may restrict who can use an entitlement
- Owners may delegate these powers to other users

# Architecture - abstract

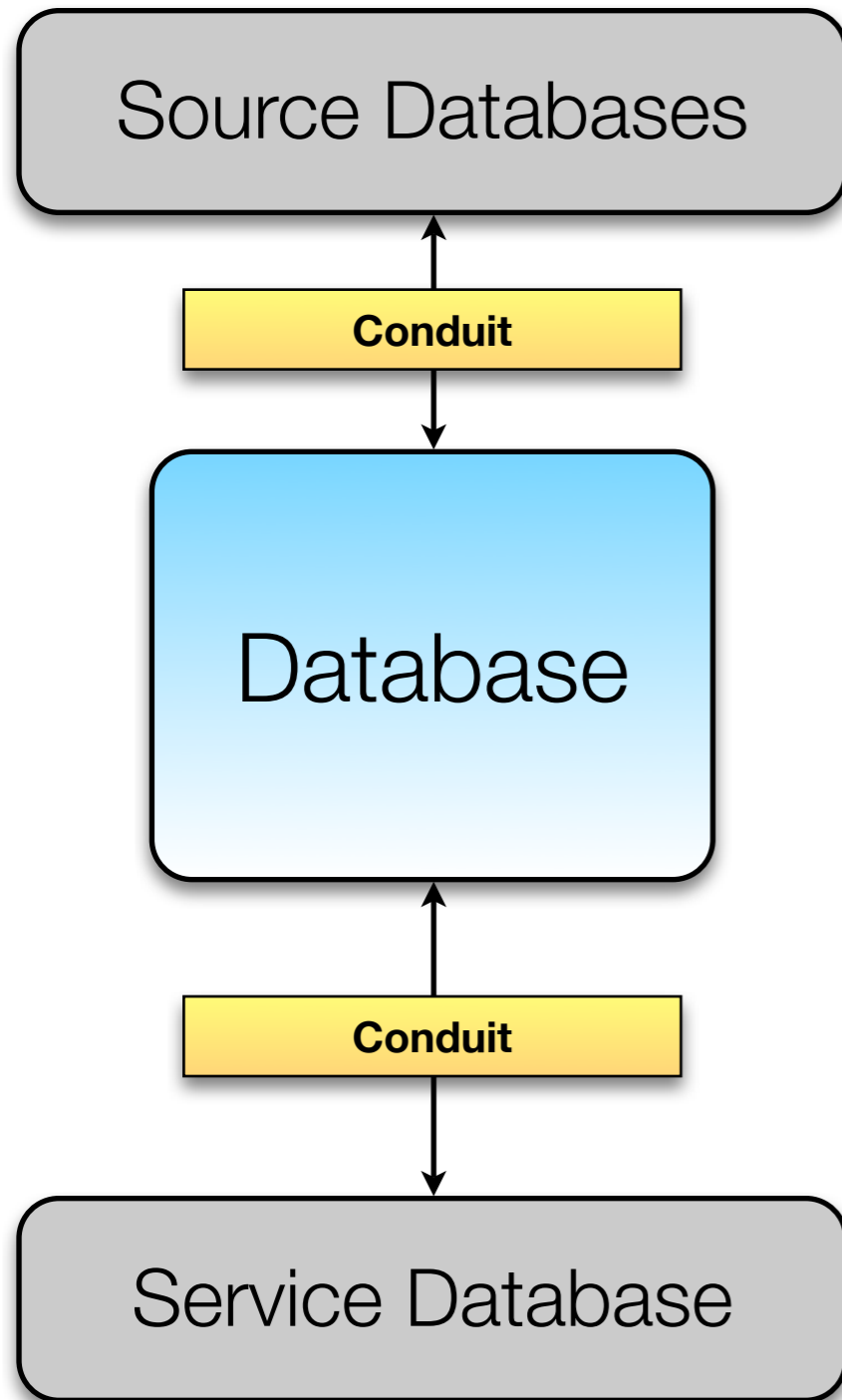
---





# Data flow

---



- Source database may be flat file, RDBMs, LDAP, whatever you like
- Import *conduit* translates from source database to abstract attribute store
- Output *conduit* translates from attribute store to service-specific database
- This can be circular!

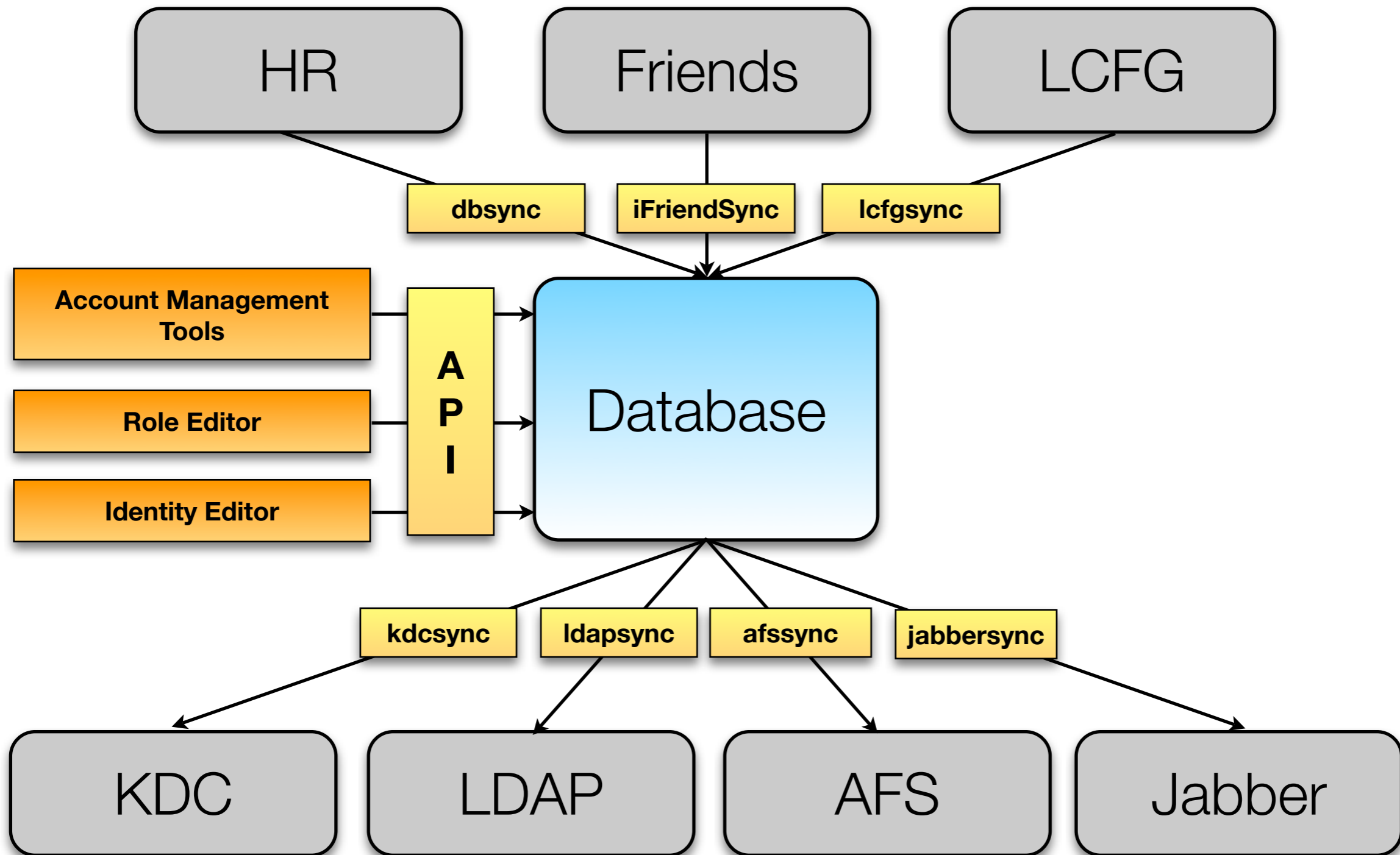
# Conduit operation

---

- Conduits can operate over entire data set, or be triggered by entitlements (ie `kdcentry` gives an instance a principal)
- Conduits can be pushed data or they may pull it. (support for notified pulls is planned)
- Conduits can work with deltas, or the whole database
- Audit runs are supported, and a reporting interface is provided
- Conduits may add arbitrary schema to the attribute store

# Architecture - detail

---



# Implementation

---



- Central database, and communication protocol LDAP based. Server implemented on top of OpenLDAP
- Advantages
  - Common, secure, well specified protocol
  - Hierarchical directory layout suits our data model
  - Straightforward, well documented, common place client API
- Disadvantages
  - Hard to express role and entitlement ownership model
  - Server data model requires external scripts, or plugins

# Implementation - Conduits

---

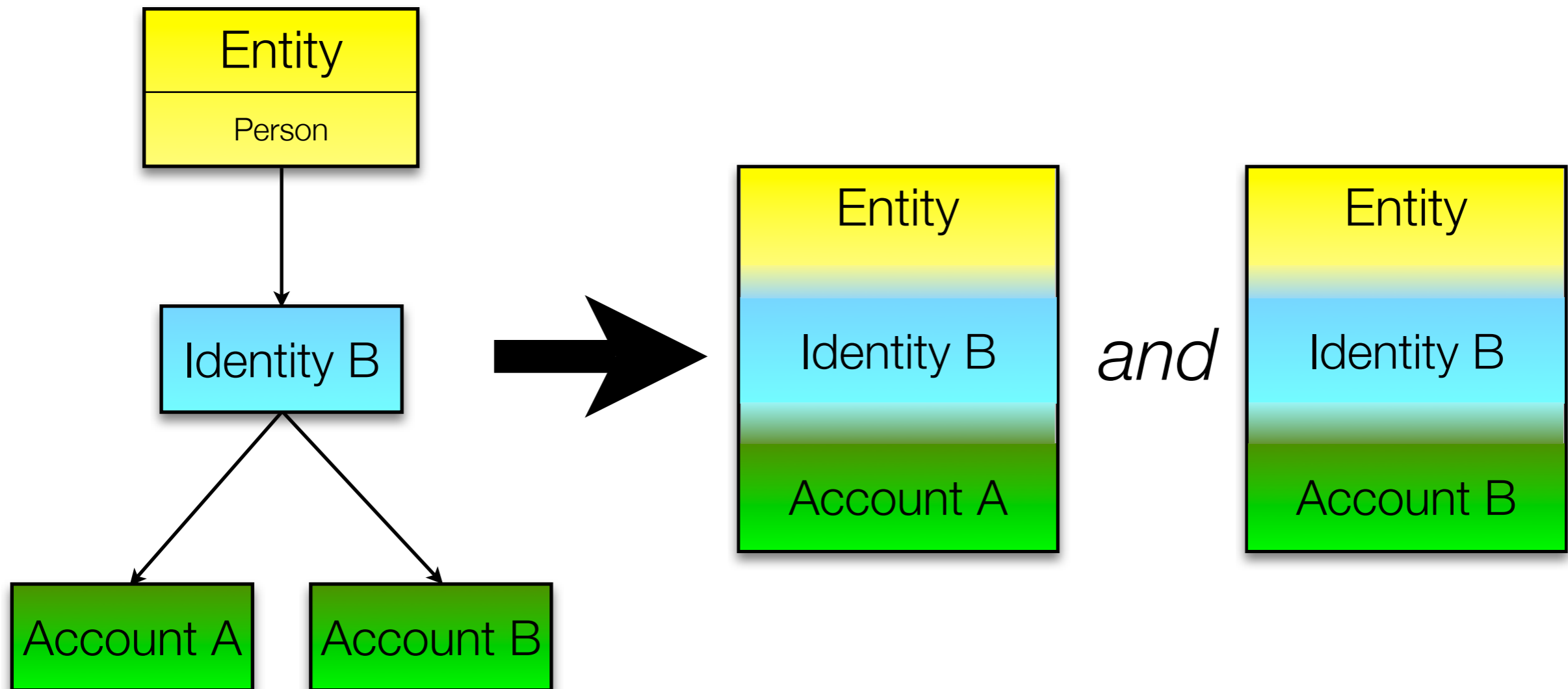
- Conduit simplicity is the primary goal
- Conduits which just want to pull information just need to
  - Register their identity, and triggering entitlement with the server
  - Perform Idapsearches with that identity

```
ldapsearch -h prometheus.inf.ed.ac.uk \  
  -b "o=Prometheus,dc=inf,dc=ed,dc=ac,dc=uk" \  
  (objectClass=prometheusIdentity)
```

# Implementation - Conduit Simplicity

---

- Simple conduits require the server have the ability to collapse the entity data model ...



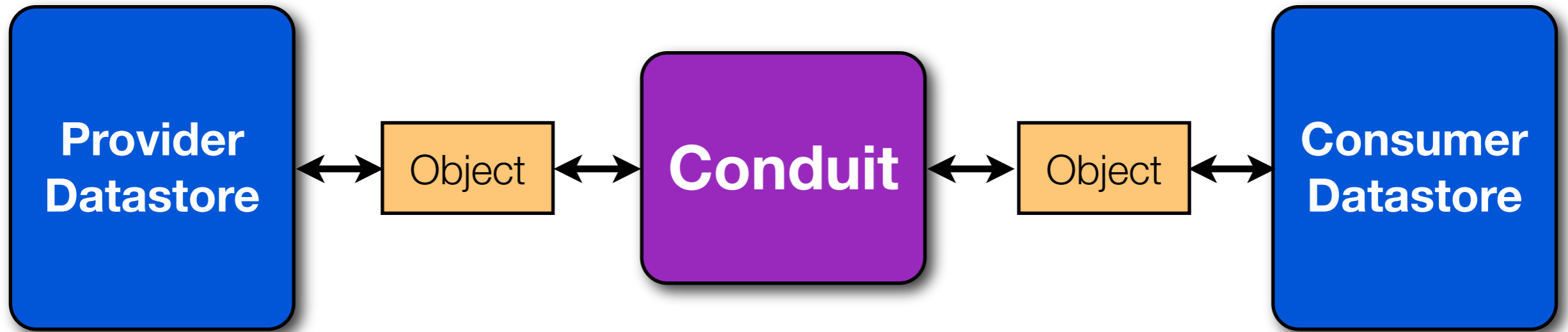
# Implementation - Conduit complexity

---

- More complicated conduits can use a framework
- Uses perl & Moose
- A general purpose mechanism for synchronising the contents of two databases

# Conduit Architecture

---



- Provider can give all objects, a subset, or objects which have changed since last run
- Conduit merges object from Provider with that in Consumer
- Provider and Consumer updated with resulting object



# Prometheus Datastore

---

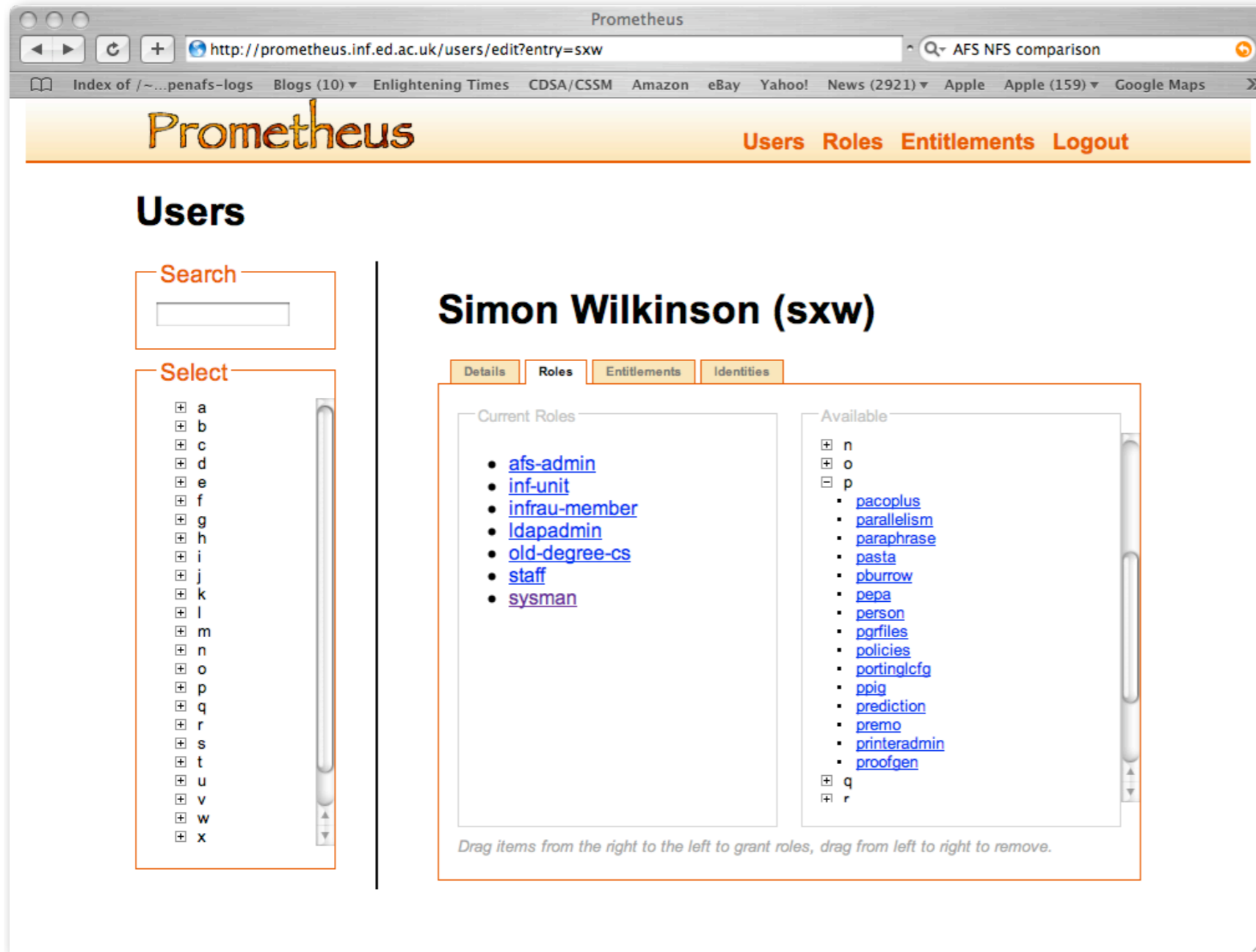
- The prometheus database is a datastore within this model
- Syncrepl implementation permits conduits to perform delta updates
- LDAP directory also stores
  - Conduit configuration information
  - Conduit error reporting
- For centrally owned conduits only!

# Error reporting

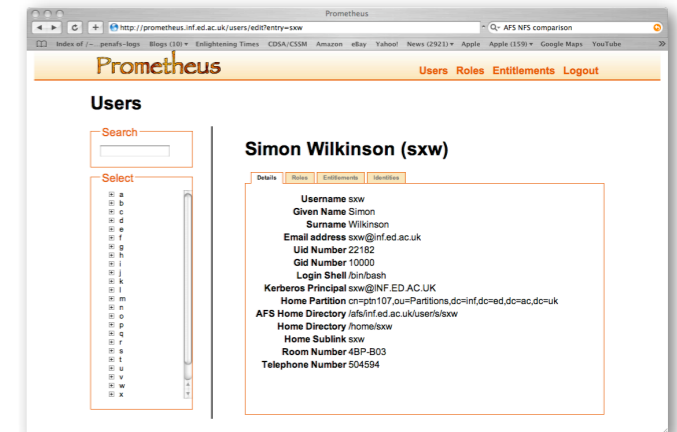
---

- Two different types of error
  - Audit errors
  - Synchronisation errors
- Audit run errors are contained within a single log
- Synchronisation errors are grouped by the conduit identifier, object identifier and message. We only display each error once!

# The Web Interface



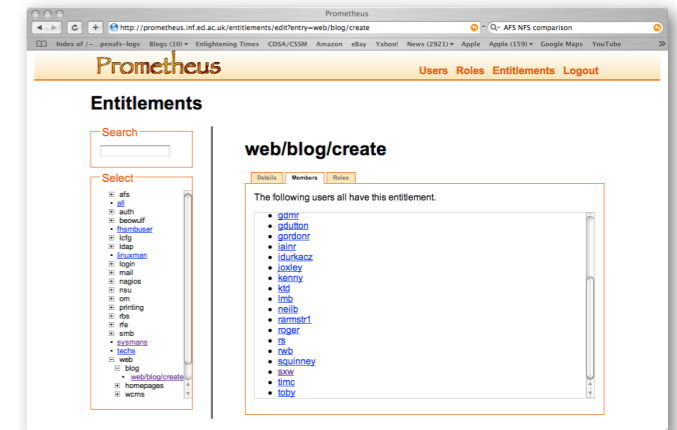
The screenshot shows the Prometheus web interface for editing user 'sxw'. The browser address bar is `http://prometheus.inf.ed.ac.uk/users/edit?entry=sxw`. The page header includes the Prometheus logo and navigation links: [Users](#), [Roles](#), [Entitlements](#), and [Logout](#). On the left, there is a 'Users' section with a search box and a 'Select' menu listing characters a through x. The main content area is titled 'Simon Wilkinson (sxw)' and contains four tabs: 'Details', 'Roles', 'Entitlements', and 'Identities'. The 'Roles' tab is active, showing 'Current Roles' (a list of roles like `afs-admin`, `inf-unit`, `infrau-member`, `ldapadmin`, `old-degree-cs`, `staff`, `sysman`) and 'Available' roles (a list of roles like `n`, `o`, `p`, `pacoplus`, `parallelism`, `paraphrase`, `pasta`, `pburrow`, `pepa`, `person`, `pgrfiles`, `policies`, `portinglcfg`, `ppig`, `prediction`, `premo`, `printeradmin`, `proofgen`, `q`, `r`). A note at the bottom reads: 'Drag items from the right to the left to grant roles, drag from left to right to remove.'



This screenshot shows the 'Details' tab for user 'sxw'. The page includes a search box and a 'Select' menu. The main content area is titled 'Simon Wilkinson (sxw)' and contains a list of user attributes: Username `sxw`, Given Name `Simon`, Surname `Wilkinson`, Email address `sxw@inf.ed.ac.uk`, User Number `22182`, Gid Number `10000`, Login Shell `/bin/bash`, Kerberos Principal `sxw@INF.ED.AC.UK`, Home Partition `cn=pn107,ou=Partitions,dc=inf,dc=ed,dc=ac,dc=uk`, AFS Home Directory `/afs/inf.ed.ac.uk/users/sxw`, Home Directory `/home/sxw`, Home Sublink `sxw`, Room Number `4BP-B03`, and Telephone Number `504594`.



This screenshot shows the 'Roles' page for the role 'sysman'. The page includes a search box and a 'Select' menu listing various roles. The main content area is titled 'sysman' and contains a list of entitlements: `afs/compens/write`, `afs/dnsm/operate`, `afs/quotas/write`, `afs/quota/thead`, `afs/user/write`, `ldap/print/write`, `ldap/write`, `ldap/group/write`, `ldap/mail/write`, `ldap/printer/write`, `on/all`, `on/kill`, `lca/inventory/write`, `printing/restricted/print`, `printing/color/print`, `afs/sysman`, and `web/blog/create`.



This screenshot shows the 'Entitlements' page for the entitlement 'web/blog/create'. The page includes a search box and a 'Select' menu listing various entitlements. The main content area is titled 'web/blog/create' and contains a list of users who have this entitlement: `gdmr`, `gdutton`, `gordonr`, `laur`, `idurkacz`, `joxday`, `keatney`, `ksj`, `lime`, `neilib`, `rammstr1`, `roger`, `rs`, `rtb`, `squintney`, `sxw`, `tmc`, and `loby`.

# Web Interface Implementation

---

- Written using Catalyst - a perl MVC framework
- Uses prometheus API to communicate with LDAP server
- Uses user's authentication tokens to secure server connection
- All authorization and access control checks performed in the server



# Conclusions

---

- System design that meets all of our requirements
- Implementation continuing as an evolution of our existing system
- Code drops will be available shortly
- **Very** interested in talking to other sites that might be interested in any of this!

# Questions?

---

**This talk:** <http://www.dice.inf.ed.ac.uk/publications/>

**Prometheus:** <http://www.dice.inf.ed.ac.uk/prometheus/>

**Me:** [simon@sxw.org.uk](mailto:simon@sxw.org.uk)