Introduction to LDAP and Directory Services

Evolveum

Radovan Semančík

Open Source Weekend, April 2016

Radovan Semančík

Current:

Software Architect at Evolveum

Architect of Evolveum midPoint

Contributor to ConnId and Apache Directory API

Past:

Sun LDAP and IDM deployments (early 2000s)

OpenIDM v1, OpenICF

Many software architecture and security projects

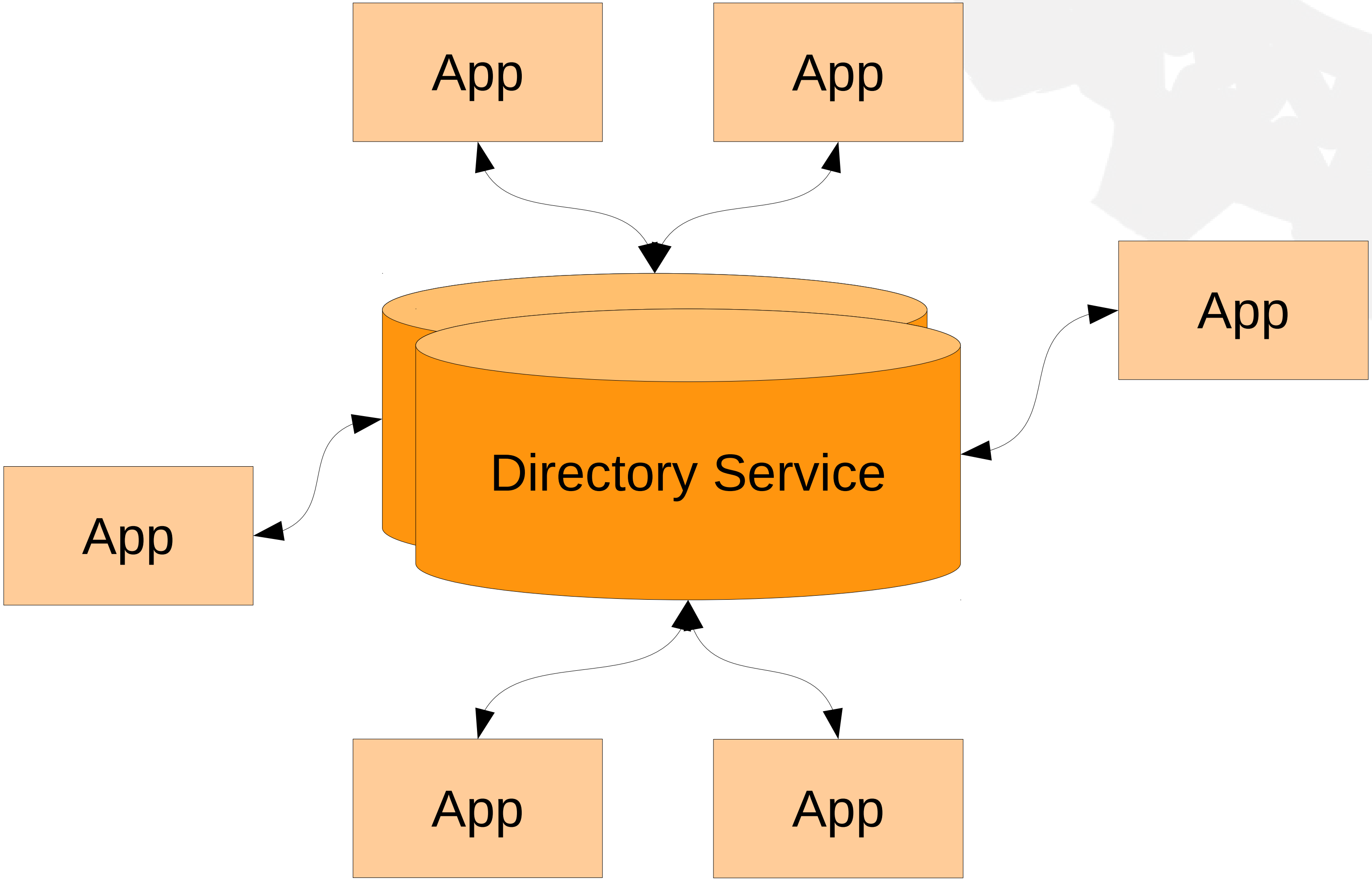
Directory Service

A structured repository of information on people and resources within an organisation, facilitating management and communication.

The Free On-Line Dictionary of Computing

- **“Database” usually containing data about:**
  - People (users, employees, customers, ...)
  - Groups of people (access control groups, roles, ...)
  - Devices (servers, network elements, ...)
  - Configuration data

Directory Service Architecture



Directory Service Features

LDAP = NoSQL
before
it was cool

- **Shared database**
  - Standard protocol (RFC 4511)
  - Standardized schema
    - inetOrgPerson (RFC 2798), posixAccount (RFC 3207), ...
- **Lightweight**
  - No locking, easy to replicate
  - Low overhead, high performance
- **Fast reads, slow writes**
  - Ideal for “configuration” data

Directory Service Evolution

- **X.500**
  - Origin: 1988 CCITT (now ITU-*T*) as support for X.400
  - Global directory service (similar to DNS)
  - Very complex (DAP over OSI protocol stack)
- **LDAP**(*
  - Simplified version of X.500 (DAP)
  - Origin: 1995 IETF (RFC 1777)
  - Currently LDAPv3 (RFC 2251, 3377, 3771)
- **MS Active Directory, NDS**
  - Originated independently of X.500

(* Strictly speaking “LDAP” denotes network protocol. However it is commonly used to refer to the directory system as a whole.

OSS Directory Servers

- **389 Directory Server (Fedora)** RedHat
- **Apache Directory Server** Apache Foundation
- **OpenDJ (OpenDS)** ForgeRock
- **OpenLDAP** Symas

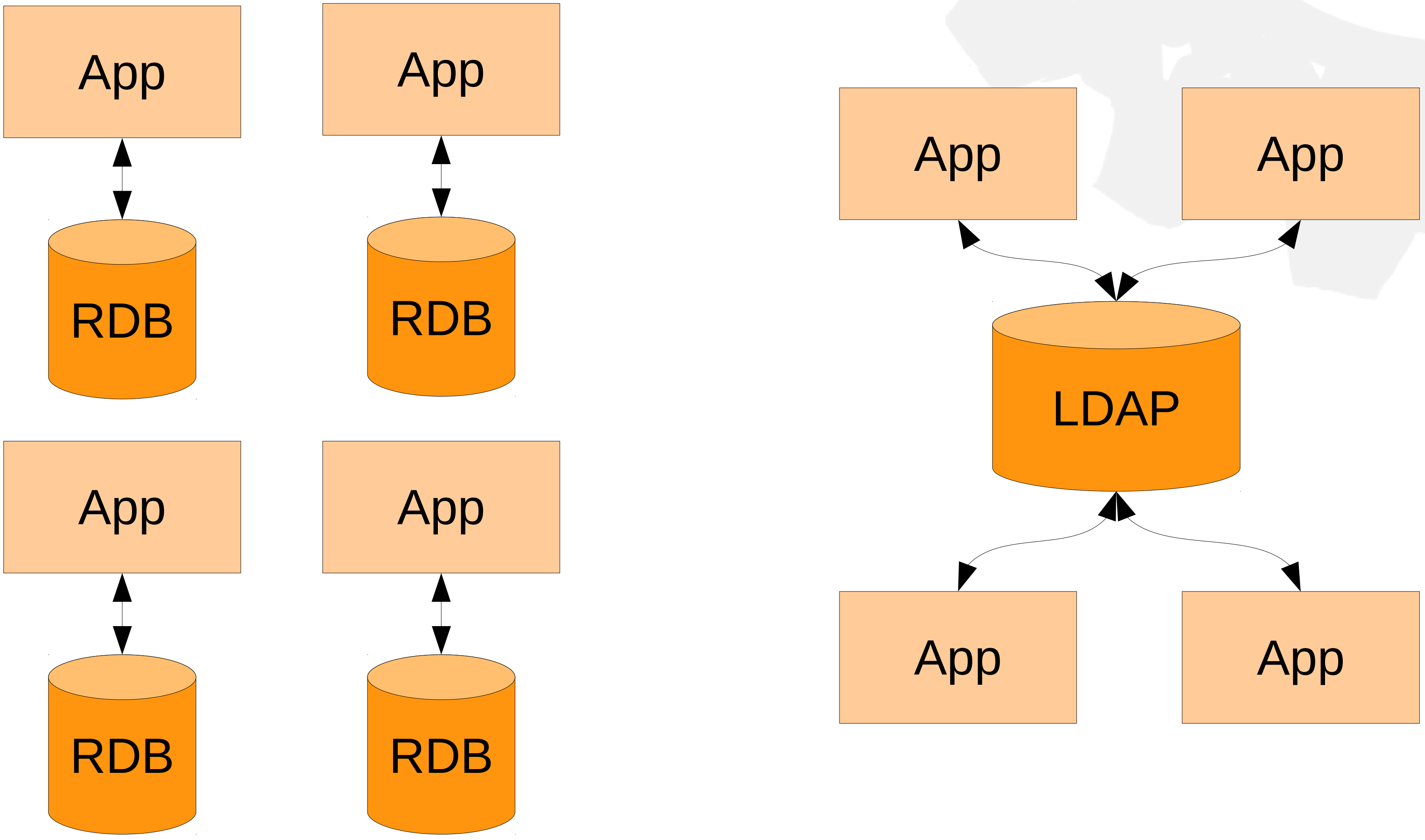
OpenLDAP

- **Native LDAP Server**
  - Storing data in LMDB databases (or other backends)
  - Tailor-made database and indexing
  - Excellent performance
  - Access protocols: LDAP
  - Written in C, long source code history
- **UNIX install root directory**
  - `/opt/symas` or usual OS directories (packages)
- **Configuration directory**
  - `/etc/openldap, /etc/ldap/slapd.conf`

Why Do We Want Directory Services?

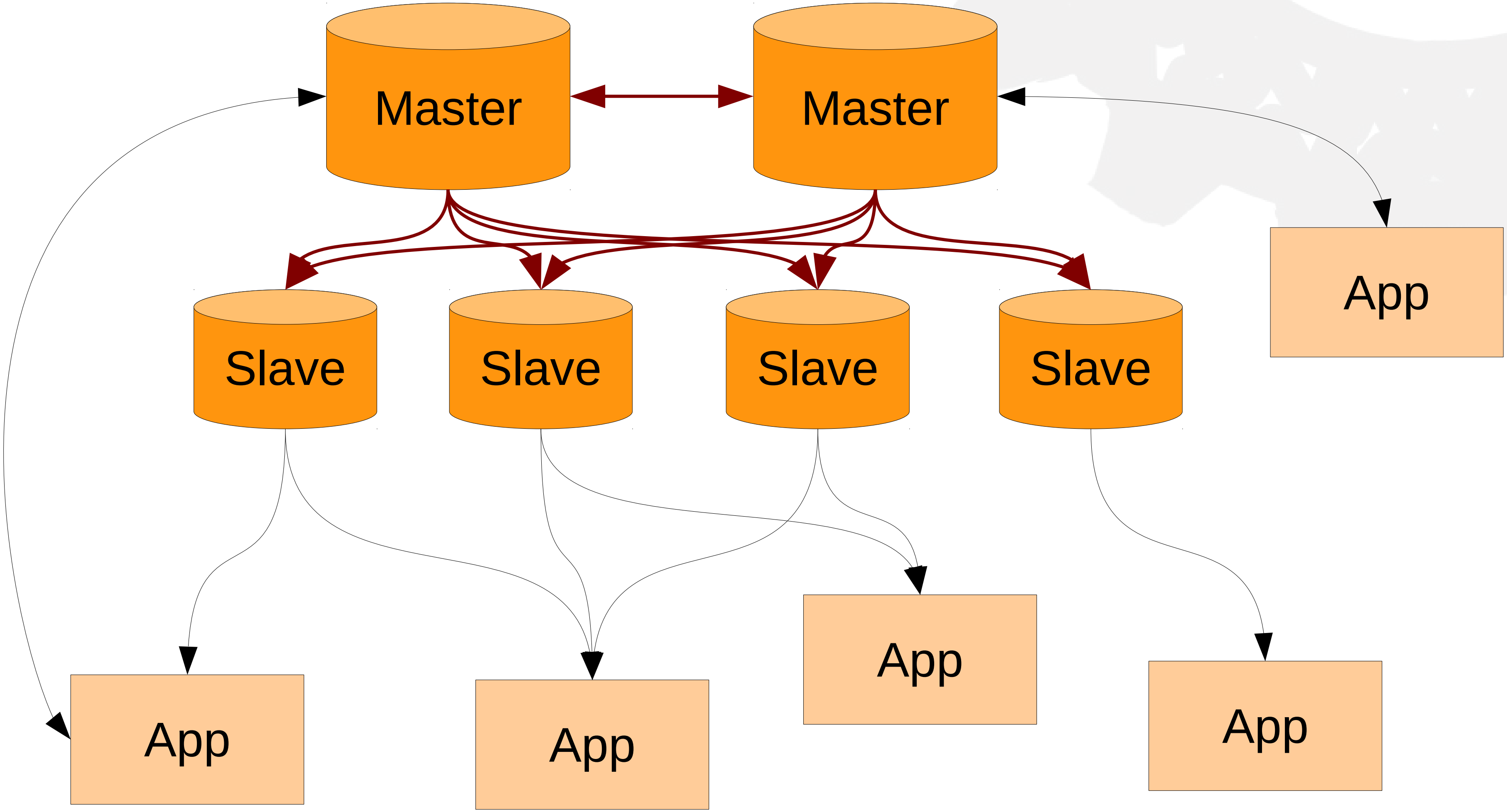
- **They are fast! Really fast. When reading.**
  - Faster than the fastest relational databases
  - But slow when writing (approx 10 times)
- **Low resource consumption**
  - Approx 10 times lower that relational DBs
- **Scaling ad nauseam**
  - 1M entries is nothing. 1B is still easy.
- **Easy to replicate the data**
  - High availability, performance, scaling

Directory Service vs Relational Databases



Evolveum

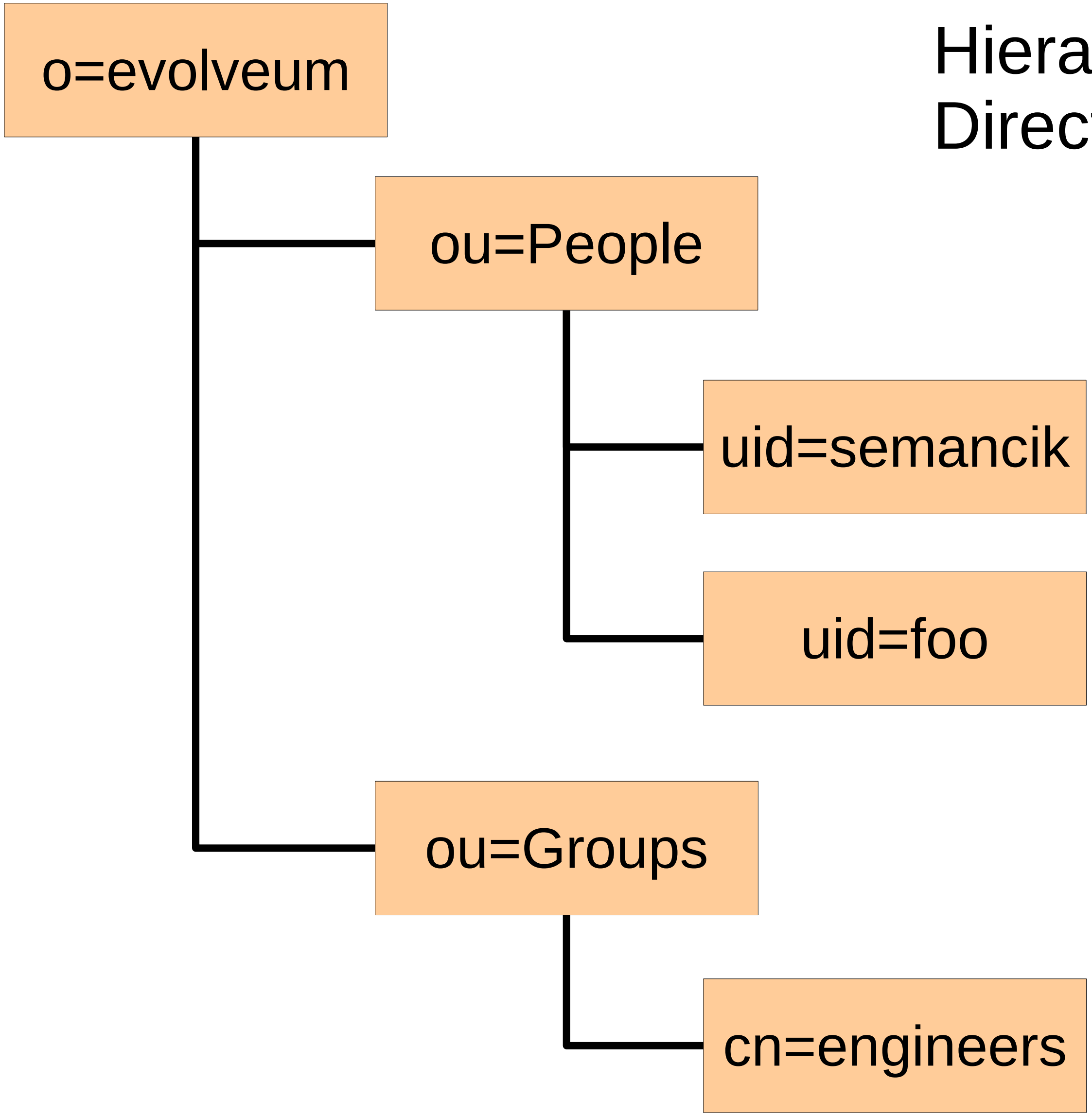
Directory Service Replication



Evolveum

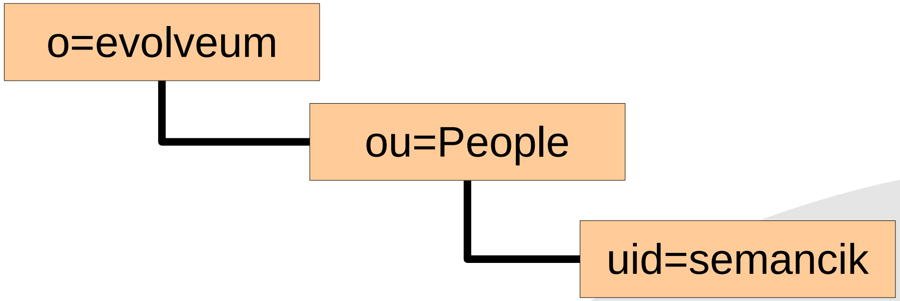
LDAP Basics

Directory Information Tree

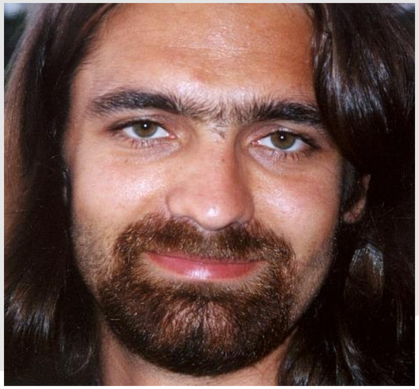


Hierarchical structure
Directory Information Tree (DIT)

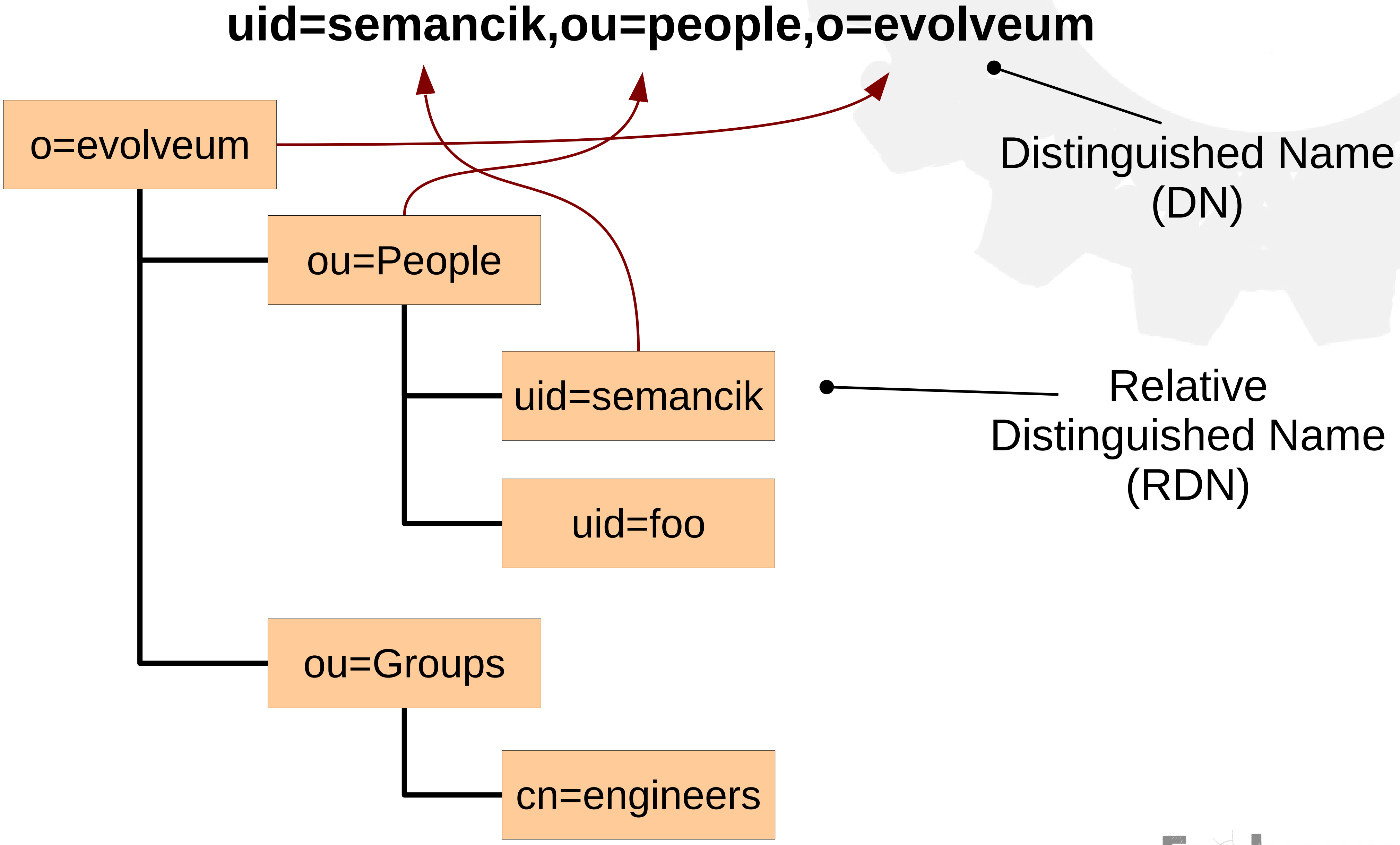
Objects & Attributes



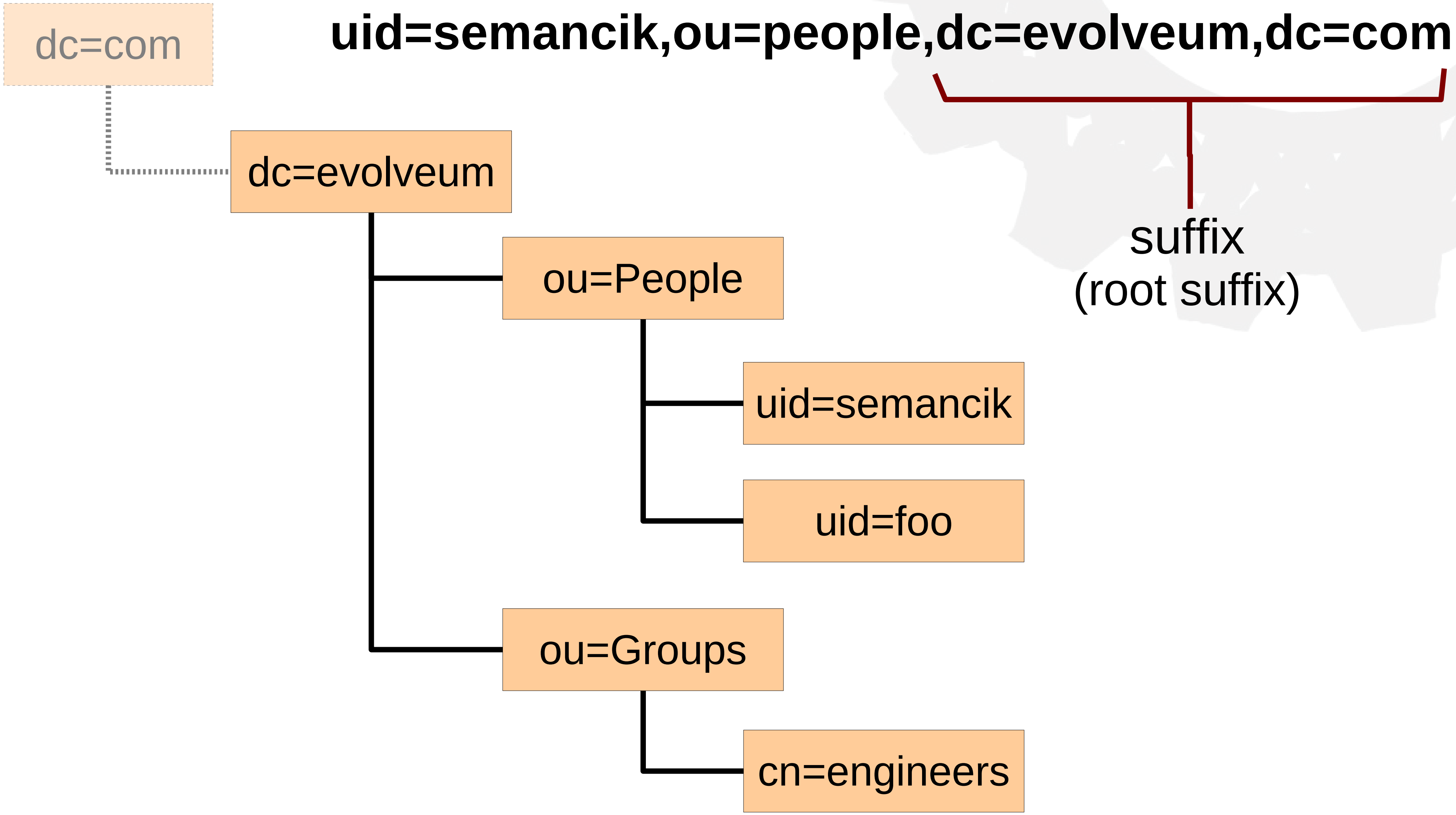
cn=Radovan Semančík
sn=Semančík
uid=semancik
objectClass=inetOrgPerson
title=Software Architect
telephoneNumber=+421 2 49100100
telephoneNumber=+421 2 49100136
preferredLanguage=en
mail=radovan.semancik@evolveum.com
jpegPhoto=



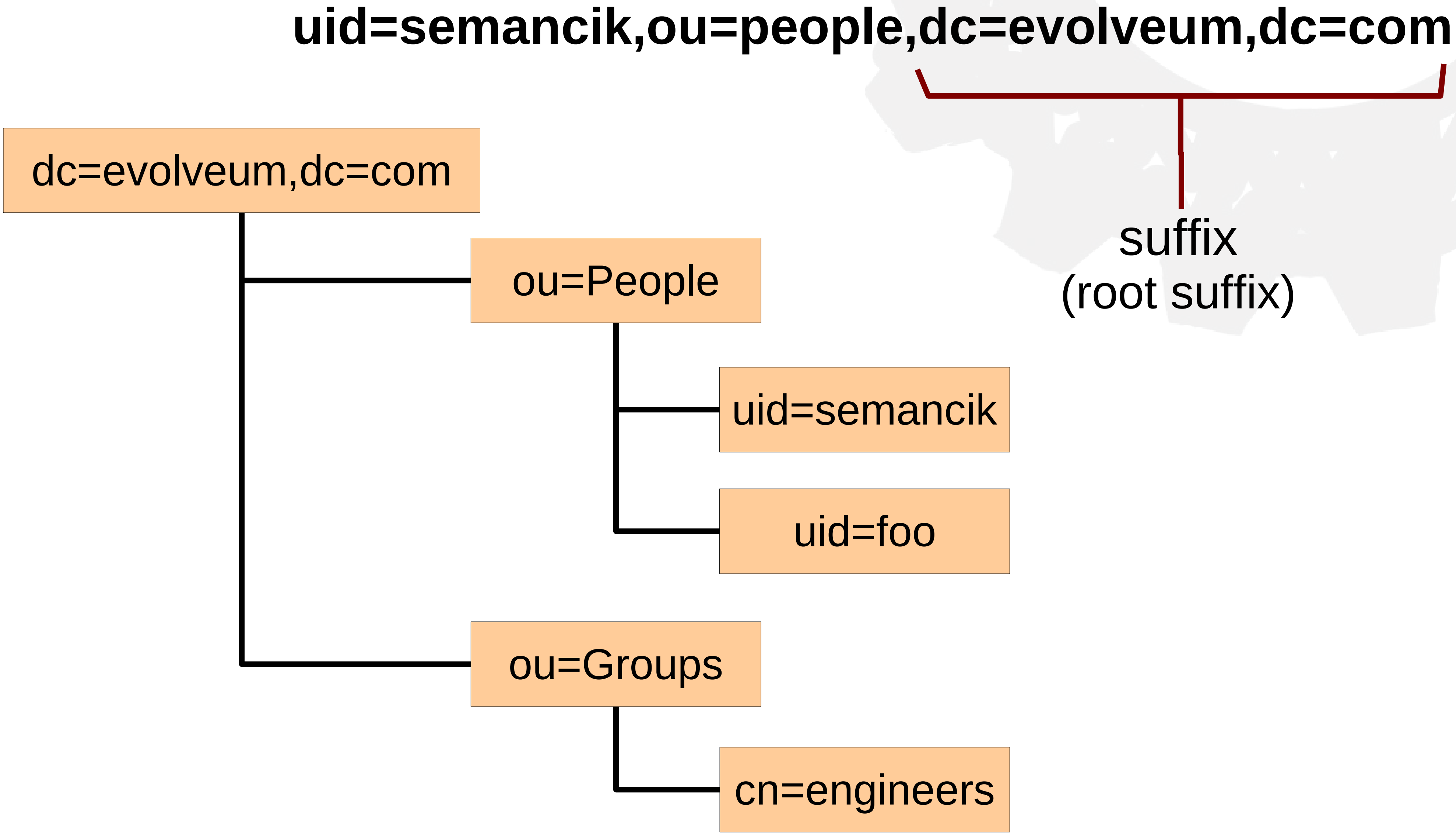
Distinguished Name



Directory Suffix



Directory Suffix



Suffix Conventions

- **o=evolveum,c=sk**

  “Traditional” X.500

- **o=evolveum.com**

  Hybrid, based on DNS domain

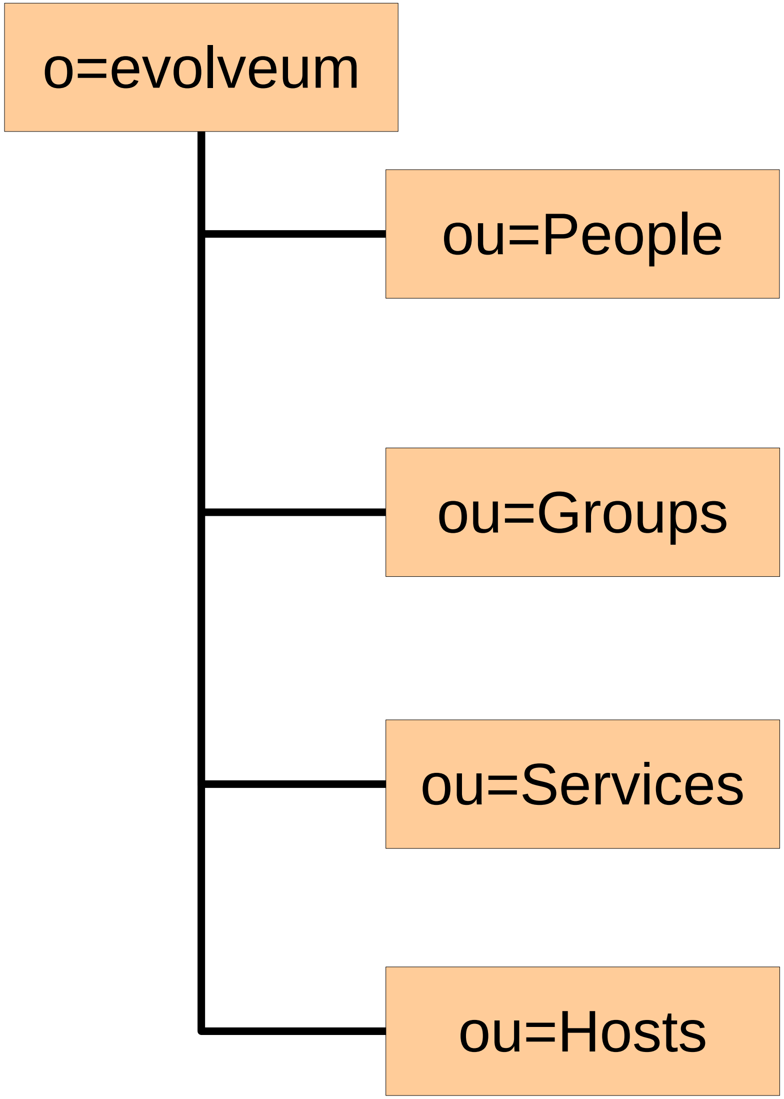
- **dc=evolveum,dc=com**

  Internet style, based on DNS domain

- **o=evolveum.com,o=isp**

  Nested, sometimes used in ISP/ASP environment

DIT Structure Conventions



o=evolveum

ou=People

ou=Groups

ou=Services

ou=Hosts

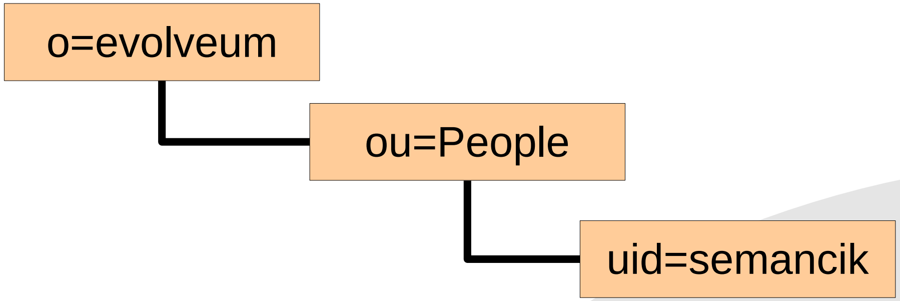
Users
objectClass=inetOrgPerson
(posixAccount)

Groups
objectClass=groupOfUniqueNames
(posixGroup)

Services (Access Manager, POSIX)
objectClass=sunService, ipService

Hosts (POSIX)
objectClass=ipHost

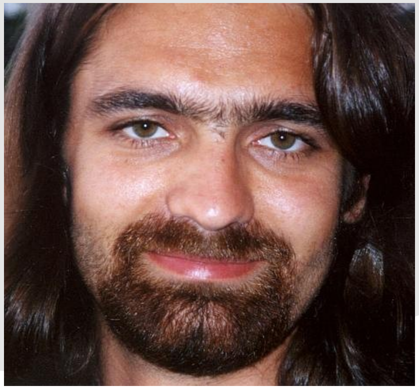
Objects & Attributes


o=evolveum

ou=People

uid=semancik

cn=Radovan Semančík
sn=Semančík
uid=semancik
objectClass=inetOrgPerson
title=Software Architect
telephoneNumber=+421 2 49100100
telephoneNumber=+421 2 49100136
preferredLanguage=en
mail=radovan.semancik@evolveum.com
jpegPhoto=



Objects and Attributes

- **Attributes are global**

  Attribute name and type is the same in all objects

- **Attributes can have multiple values**

  Multi-valued attributes is the default behavior

- **Attributes may be binary**

  E.g. jpegPhoto, userCertificate, ...

- **Attribute size is practically unlimited**

  Several megabytes for jpegPhoto is pretty normal

- **Attributes may be indexed (globally)**

LDAP Browsers

- **Apache Directory Studio**

  Java, sophisticated

- **JXplorer**

  Java, simple

- **phpLDAPadmin**

  Web-based, PHP

- **Mail clients**

  Mozilla Thunderbird, Evolution, ...

LDAP Interchange Format (LDIF)

- Textual format used for storing and exchange of data among LDAP servers
- Specified in RFC 2849
- Example:

```
dn: uid=test,ou=People,o=nlight
objectClass: top
objectClass: person
uid: test
cn: Test Testovic
sn: Testovic

dn: uid=foo,ou=People,o=nlight
objectClass: top
objectClass: person
uid: foo
cn: Foo Bar
sn: Bar
```

LDIF Example

```
dn: uid=semancik,ou=People,o=nlight
mail: radovan.semancik@nlight.eu
sn: Semancik
cn: Radovan Semancik
givenName: Radovan
uid: semancik
objectClass: top
objectClass: organizationalperson
objectClass: inetorgperson
objectClass: person
userPassword:: e1NTSEF9SzU4c1ZpOStFZS8yZlF
 zVFN6WVhNTi9obGwzQVZRVno2R3dkUHc9PQ==

dn: uid=foobar,ou=People,o=nlight
uid:foobar
```

LDIF Example

The “dn” line is always first

```
dn: uid=semancik,ou=People,dc=nlight,dc=sk
mail: radovan.semancik@nlight.eu
sn: Semancik
cn: Radovan Semancik
givenName: Radovan
uid: semancik
objectClass: top
objectClass: organizationalperson
objectClass: inetorgperson
objectClass: person
userPassword:: e1NTSEF9SzU4c1ZpOStFZS8yZlF
 zVFN6WVhNTi9obGwzQVZRVno2R3dkUHc9PQ==

dn: uid=foobar,ou=People,dc=nlight,dc=sk
uid:foobar
```

Multiple lines for multi-valued attributes

Base64 encoding

White space: continuation of previous line

Empty line: end of object

Next object

LDAP Operations

Evolveum

Basic LDAP Operations

- **search**

  Find object in the DIT, also used for reading data

- **add**

  Creating objects

- **modify**

  Changing objects

- **delete**

  Removing objects

- **bind**

  User authentication

Search Parameters

- **Base DN: The point in the tree where to start the search**
- **Scope: search scope**

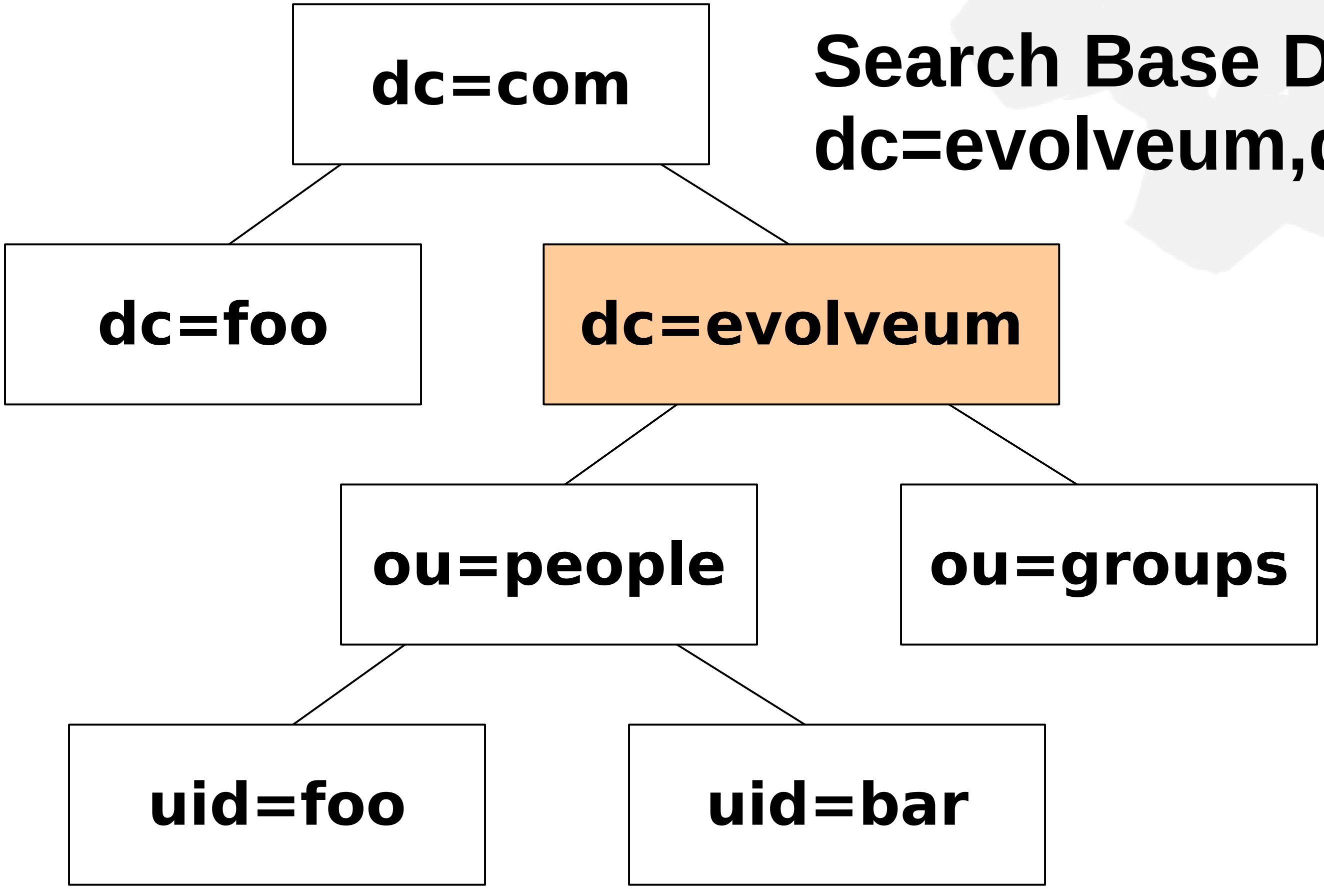
  base: just the base object (used for reading data)

  one: just one level under the base object

  subtree: entire subtree under the base object
- **Filter: attribute conditions**

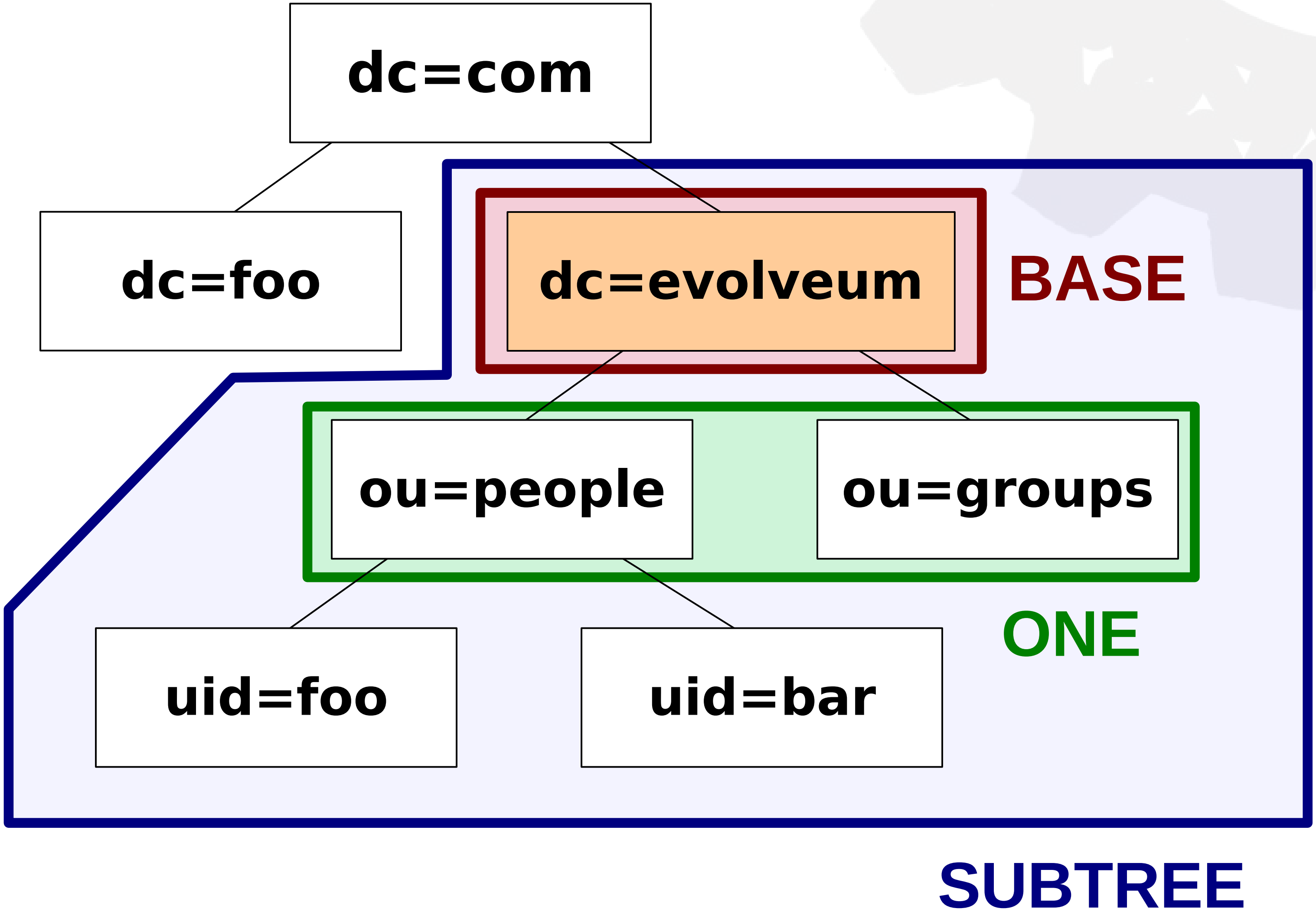
  Using expressions to form complex conditions
- **Attribute list (optional)**

Search: scope



Search Base DN:
dc=evolveum,dc=com

Search: scope



Search Filter (string data)

- `(uid=semancik)`
- `(cn=*sem*)` - substring filter
- `(uid=*)` - presence filter
- `(createTimestamp>=20050101000000Z)`

- `(objectClass=*)` - special filter, matches every object

Compound Search Filter

- `(&(givenName=Radovan)(sn=Semancik))`
- `(|(cn=sem*)(cn=zem*))`
- `(&(objectClass=person)(cn=FooBar))`

-
  ```
  (&
     (|
        (objectClass=person)
        (objectClass=organization)
     )
     (!(l=Bratislava))
  )
  ```

ldapseach command-line tool

-
```
ldapsearch [-b <baseDN>] [-s <scope>] <filter>
                                        [ <attrs> ]
```

- **Examples:**

```
ldapsearch -b 'ou=people,o=nlight' '(objectclass=*)'

ldapsearch -b 'o=nlight' -s one '(ou=*)'
```

- **Additional parameters**

```
-h <ldapServerHostName>

-p <ldapServerPort>

-D <bindDN>
```

ldapsearch

```
$ ldapsearch -b "dc=evolveum,dc=com" -s sub
             "(uid=semancik)"

dn: uid=semancik,ou=People,dc=evolveum,dc=com
mail: semancik@evolveum.com
sn: Semancik
cn: Radovan Semancik
givenName: Radovan
uid: semancik
objectClass: top
objectClass: organizationalperson
objectClass: inetorgperson
objectClass: person
```

Authentication: bind

- **Directory server authenticates every incoming connection using objects in the DIT**

  It means that DN is used as user identifier

- **This operation is called *bind***

- **Password is usually used (*simple bind*)**

  But there are other options (SASL)

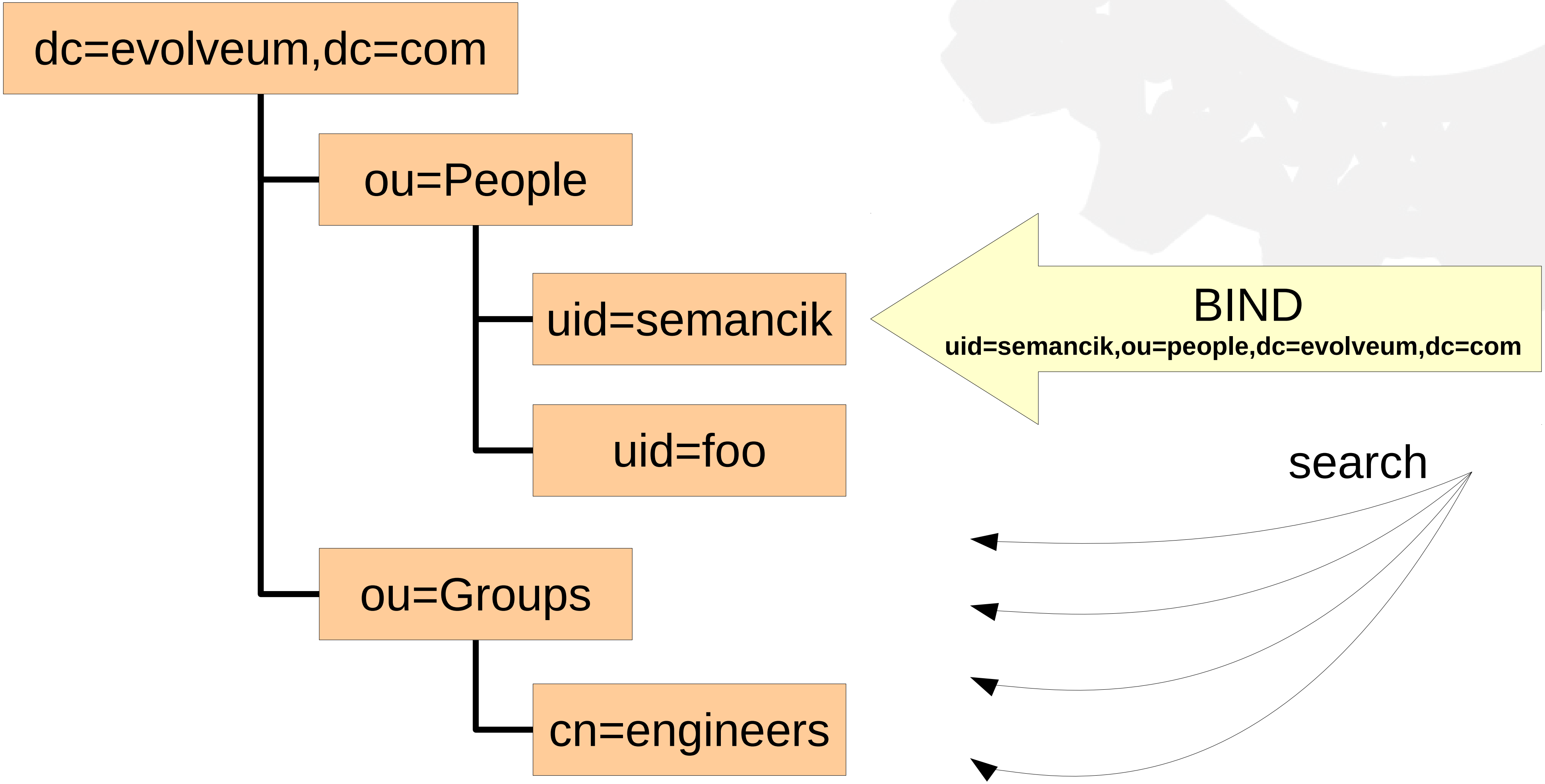
- **Example**

  User “semancik” *binds* to the server using the DN:

  `uid=semancik,ou=people,dc=nlight,dc=sk`

  This object must exists. The password (hash) stored in **`userPassword`** atribute is used.

LDAP Bind


dc=evolveum,dc=com
ou=People
uid=semancik
uid=foo
ou=Groups
cn=engineers
BIND
uid=semancik,ou=people,dc=evolveum,dc=com
search

ldapsearch with bind

```
$ ldapsearch -D “uid=admin,ou=people,dc=nlight,dc=sk”
             -w password -b "dc=nlight,dc=sk" -s sub
             "(uid=semancik)"

dn: uid=semancik,ou=People,dc=nlight,dc=sk
mail: semancik@nlight.eu
sn: Semancik
cn: Radovan Semancik
givenName: Radovan
uid: semancik
objectClass: top
objectClass: organizationalperson
objectClass: inetorgperson
objectClass: person
userPassword:: e1NTSEF9SzU4cS8y...obkUHc9PQ==
```

`userPassword` Attribute

`userPassword:` {*MECH*}*base64encodedHash*

```
userPassword: {SSHA}x2J+RZAmEdE5I7nw5Qi4zRuMBAb1CVVhpyMzIQ==
```

- **One-way hash, difficult to reverse**

  SSHA

  SHA

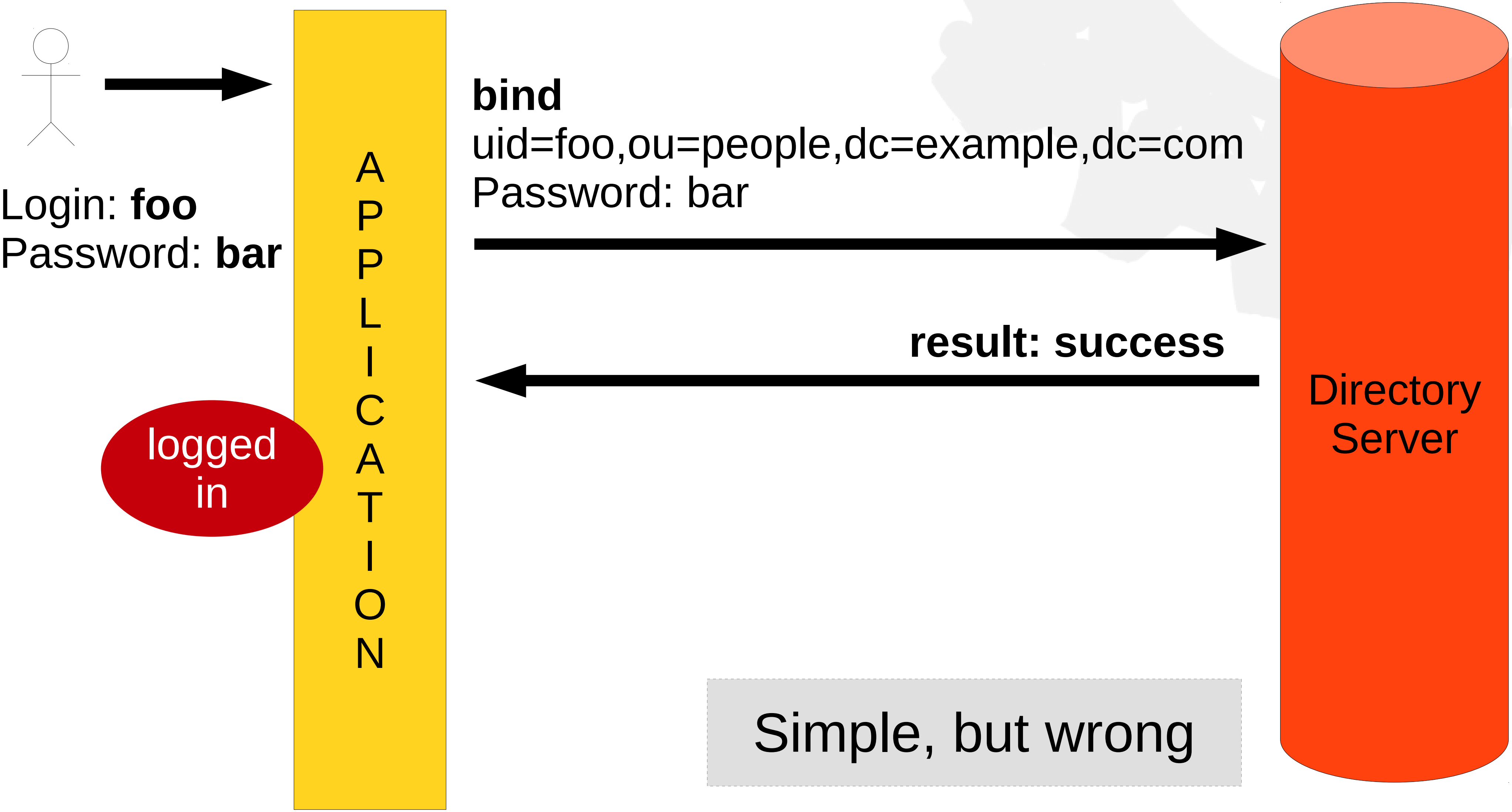
  crypt

  ...

- **Server may hash the password automatically**

  ... if not use `slappasswd`

LDAP-Based Authentication



Login: **foo**
Password: **bar**

APPLICATION

bind
uid=foo,ou=people,dc=example,dc=com
Password: bar

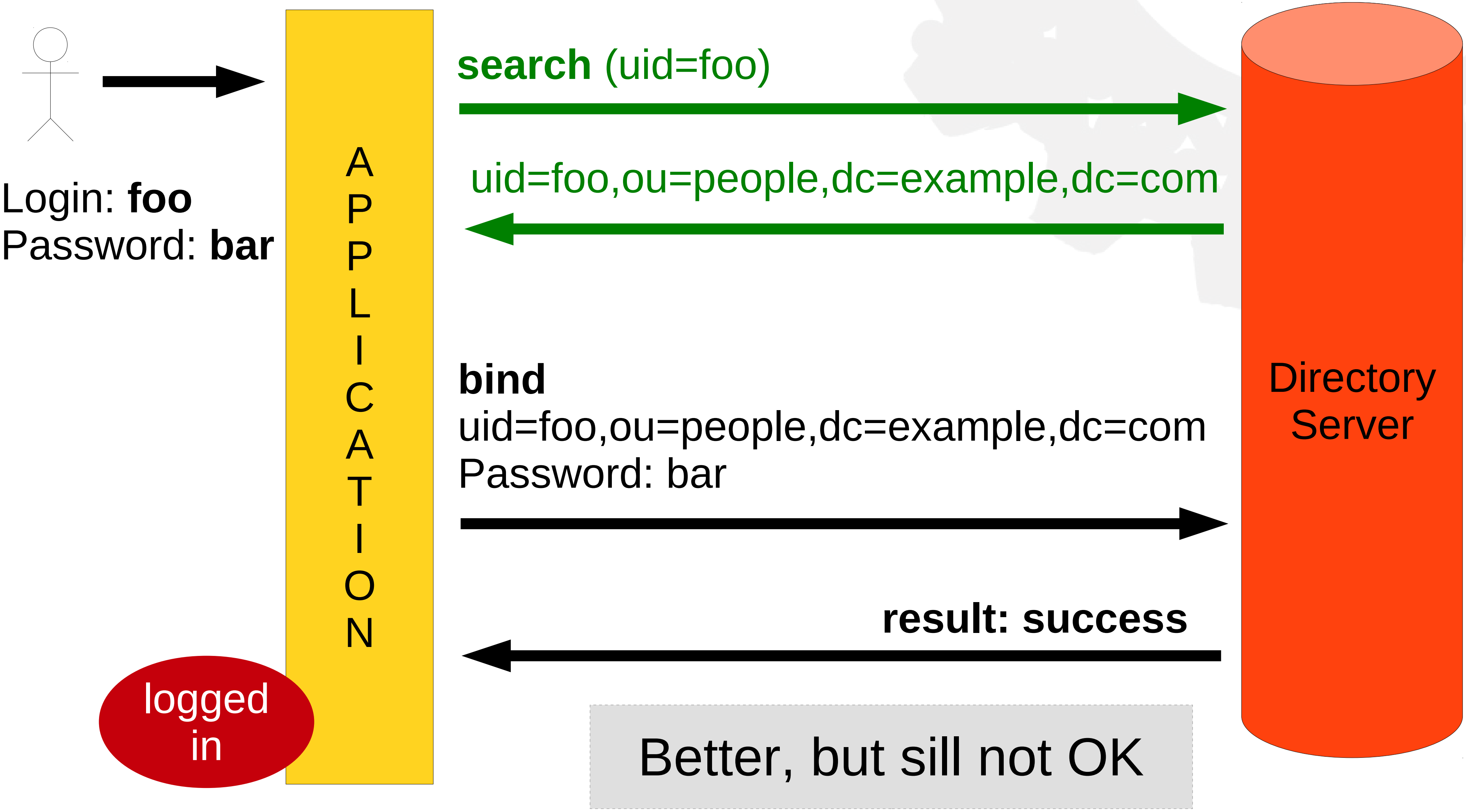
result: success

logged in

Directory Server

Simple, but wrong

LDAP-Based Authentication



Login: **foo**
Password: **bar**

APPLICATION

search (uid=foo)

uid=foo,ou=people,dc=example,dc=com

bind
uid=foo,ou=people,dc=example,dc=com
Password: bar

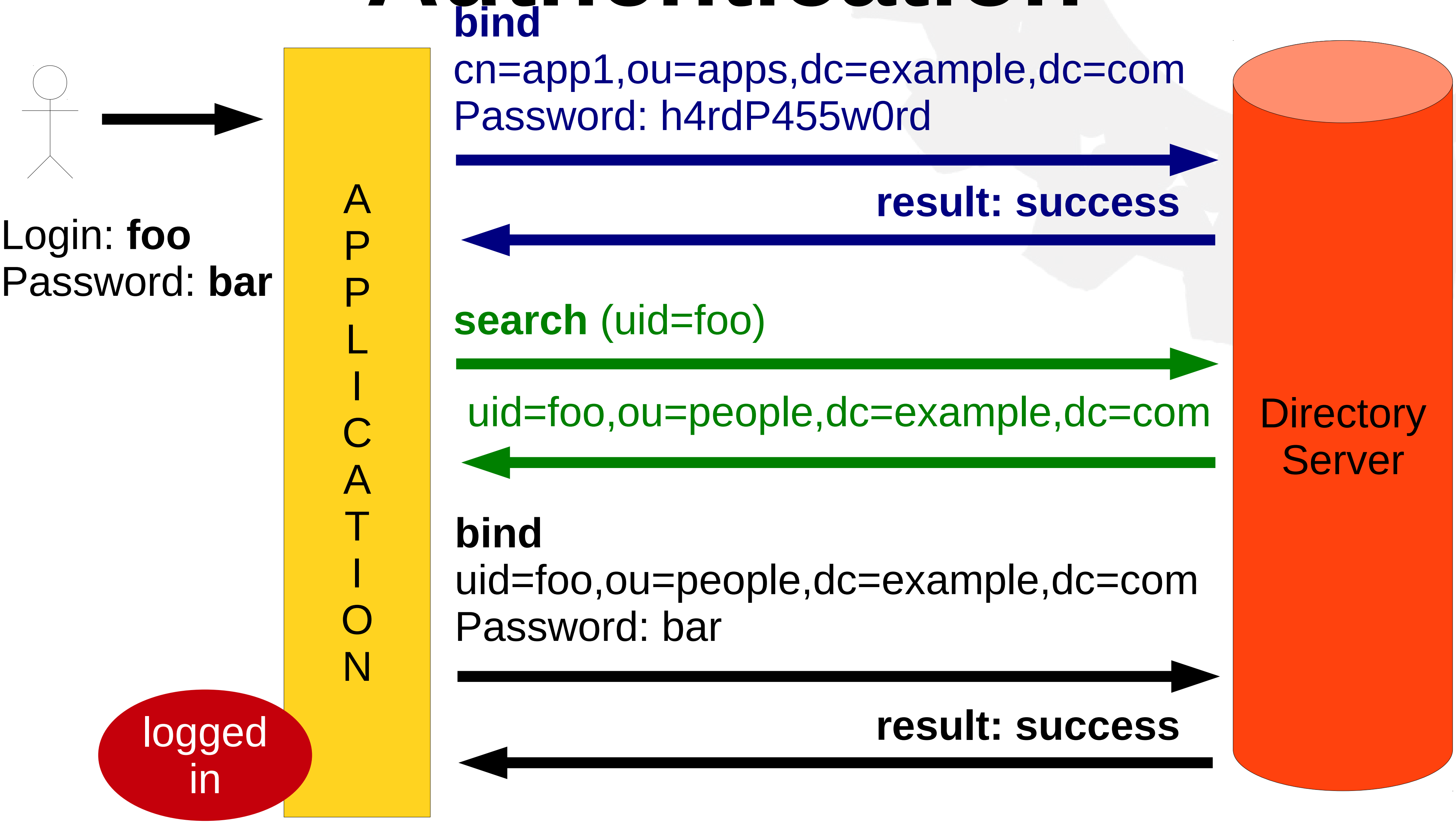
result: success

logged in

Directory Server

Better, but sill not OK

LDAP-Based Authentication



Login: **foo**
Password: **bar**

APPLICATION

bind
cn=app1,ou=apps,dc=example,dc=com
Password: h4rdP455w0rd

result: success

search (uid=foo)

uid=foo,ou=people,dc=example,dc=com

bind
uid=foo,ou=people,dc=example,dc=com
Password: bar

result: success

logged in

Directory Server

Evolveum

There is (much) more ...

- **LDAP groups**
- **LDAP server configuration**
- **LDAP schema**
- **Replication**
- **Security (ACLs, authentication)**
- **Linux as LDAP client (PAM & NSS or SSS)**
- This presentation is just a (very) short excerpt from the full training

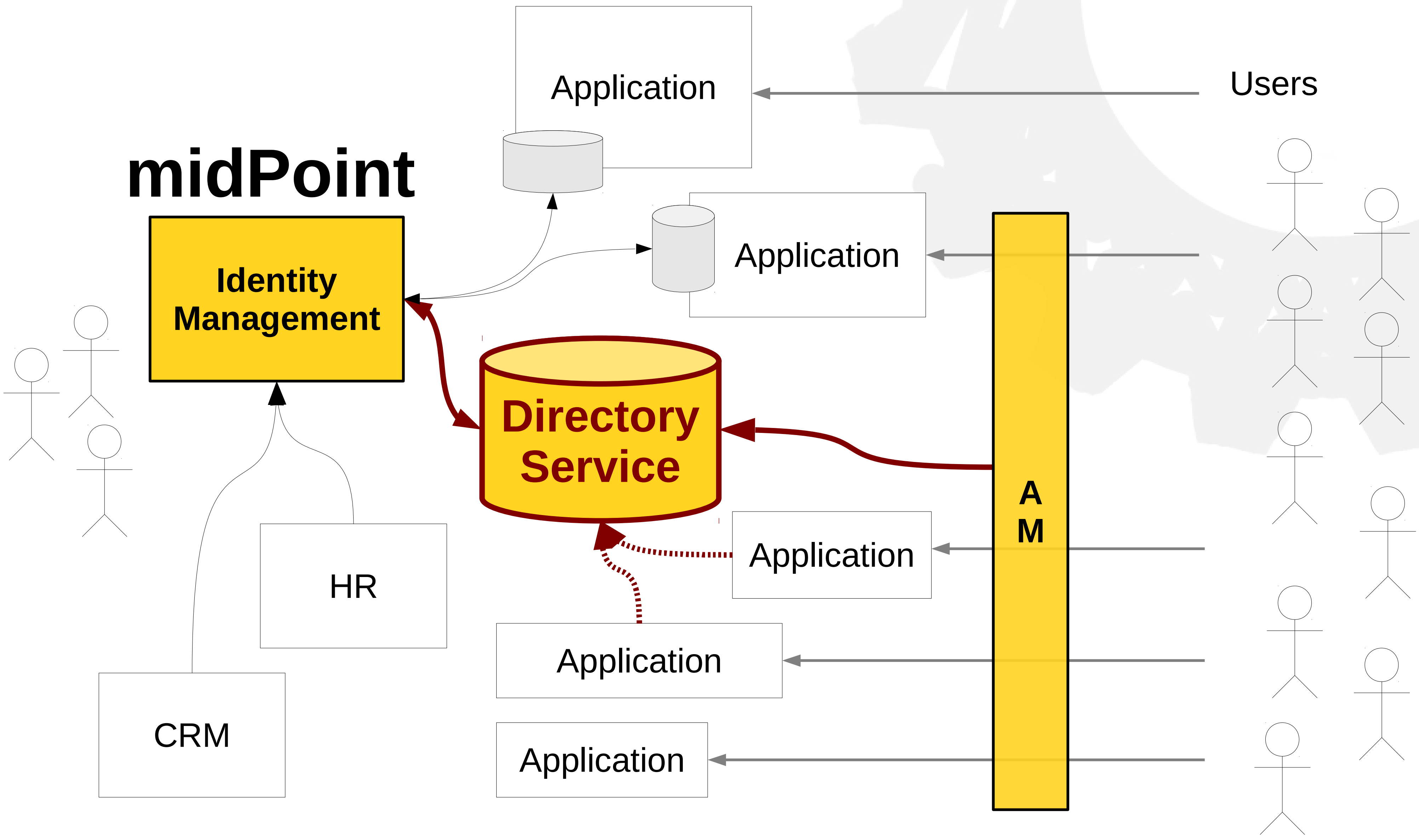
Directory Service Limitations

- **Object-oriented, no tables, no joins**
- **Standard data model limitations**
- **Management tools are … cumbersome**
- **LDAP is a database, not authentication server**
- **Single directory myth**
- **You will need more components**

  Directory server(s)

  Access management (SSO)

  Identity management


Application
Users
midPoint
Application
Identity Management
Directory Service
A M
HR
Application
Application
CRM
Application

Evolveum

Questions and Answers

Thank You

Radovan Semančík

www.evolveum.com