# Integrating OpenStack with Active Directory

## (Because AD != LDAP)

Craig Jellick
cjellick@godaddy.com

Mike Dorman
mdorman@godaddy.com

Go Daddy OpenStack Cloud Platform Group

# Agenda

- OpenStack at Go Daddy
- Keystone Integration with AD
- Nova Integration with AD
- DNS Integration
- Deployment with Puppet
- Domain Controller Proxying
- PBIS Integration

# OpenStack at Go Daddy

- Internal Pilot launched in February
- Still small, but growing:
  - Made available to over 1000 T/C users
  - >200 users have created VMs
  - ~300 Active VMs
  - Windows VMs *Coming Soon™*[*]
- Production pilot *Coming Soon™*[*]

*Not actual trademarked

# Our OpenStack

- Havana 2013.2.3
- Anvil + Stackforge’s openstack-puppet
- Neutron: ML2 driver with OVS agent
- CentOS 6 on hosts and VMs
- KVM hypervisor
- No object or block storage

# Active Directory Integration

# Integration Requirements

- Large existing AD infrastructure
  - Some legacy pain points
  - Read-only*
  - Must authenticate real and service account users against AD

# Keystone Integration strategy

- LDAP Identity backend; Database Assignment backend
- 1:1 user-to-project mapping for the pilot
  - Group-based projects to follow
  - Allow some users to assign service account users to projects for API access in the short term

# LDAP Identity backend

Robust, with a few AD quirks:

- Bug #1233365: LDAP backend fails when connecting to Active Directory root DN.
- Bug #1254849: Wrong LDAP attribute used in user response bodies
- Horizon performance issues

# Quick Look at our configuration

https://gist.github.com/cjellick/e5409d9557a25e36e926

```
[identity]
driver=keystone.identity.backends.ldap.Identity
[assignment]
driver=keystone.assignment.backends.sql.Assignment
[ldap]
url=ldaps://localhost
user=CN=svc_user,OU=Svc Acount Org Unit,DC=dc1,DC=example,DC=com
query_scope=sub
...
user_tree_dn=DC=dc1,DC=example,DC=com
user_filter=(&(objectClass=organizationalPerson)(!(objectClass=computer)))
user_objectclass=organizationalPerson
user_name_attribute=sAMAccountName
user_id_attribute=sAMAccountName
...
group_tree_dn=OU=Users,OU=My Company,DC=Domain Comp 2,DC=Domain Comp 1
group_objectclass=group
group_id_attribute=cn
group_name_attribute=name
group_member_attribute=member
```

# Nova integration with AD: name uniqueness

- All VMs are register as in AD
- Server names must be:
    - Globally unique
    - Match a regex
    - Adhere to AD name length restrictions
    - Here’s a non-upstream-worthy patch to do so:
        - https://gist.github.com/cjellick/3f528923e7b961bb32da
- osapi_compute_unique_server_name_scope=global

```
--- a/nova/api/openstack/compute/servers.py
+++ b/nova/api/openstack/compute/servers.py
...
    def _validate_server_name(self, value):
-        self._check_string_length(value, 'Server name', max_length=255)
+        if isinstance(value, basestring):
+            value = value.strip()
+        name_max = CONF.els.server_name_max_length
+        self._check_string_length(value, 'Server name', max_length=name_max)
+        self._check_regex_match(value)
+        self._check_server_name_uniqueness(value)
```

# Nova Integration with AD: DNS

- Internal DNS powered by AD
- ReST API to hide those details
- Windows VMs autoregister into DNS when they join the domain
- Hook into Nova notifications topic to know when to create/delete DNS entries for Linux VMs (and delete Windows entries)

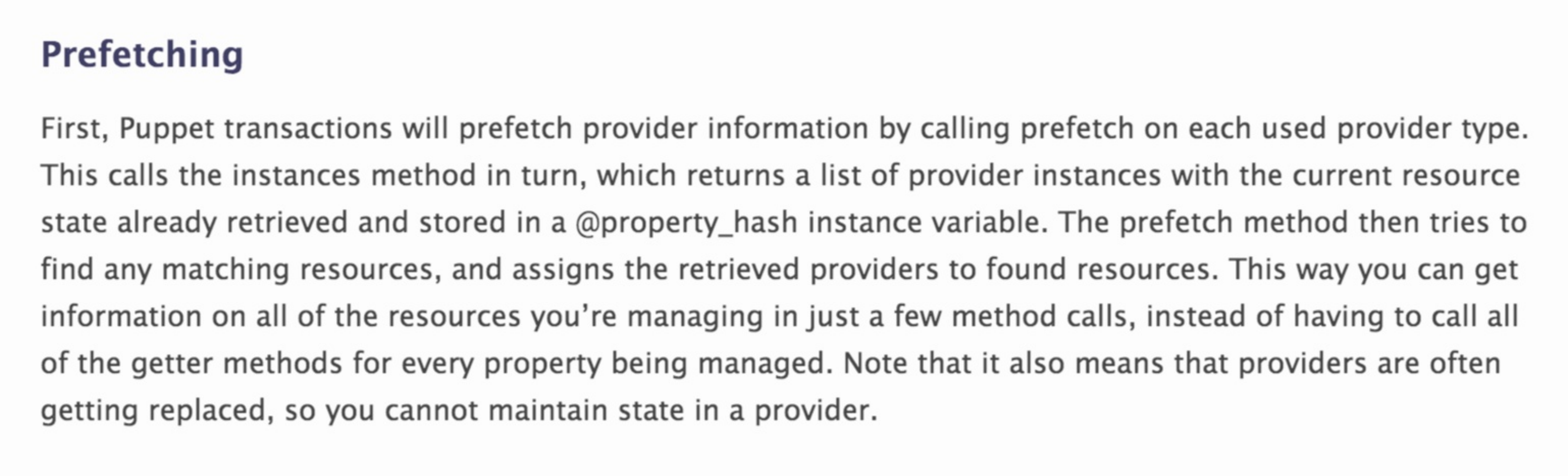
# Obligatory presentation clip art

Source: http://www.projectation.com/when-to-hand-off-the-project/

# Deployment with Puppet

- Stackforge Puppet modules
- ...with some modifications
- Mostly to implement OS SSL options

# Prefetch Scale Fail



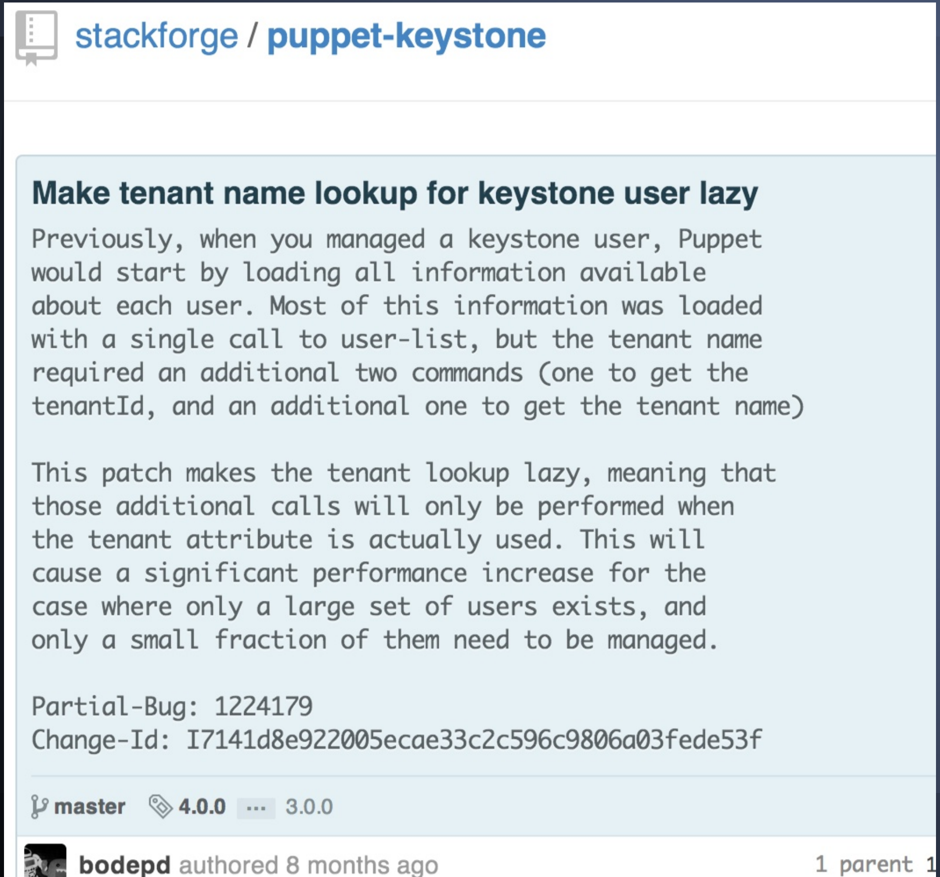
### Prefetching

First, Puppet transactions will prefetch provider information by calling prefetch on each used provider type. This calls the instances method in turn, which returns a list of provider instances with the current resource state already retrieved and stored in a @property_hash instance variable. The prefetch method then tries to find any matching resources, and assigns the retrieved providers to found resources. This way you can get information on all of the resources you're managing in just a few method calls, instead of having to call all of the getter methods for every property being managed. Note that it also means that providers are often getting replaced, so you cannot maintain state in a provider.

http://docs.puppetlabs.com/guides/provider_development.html#prefetching

- Translates to 2 keystone CLI calls for *every* user (user-get + tenant-get)

# Tenant Lazy Loading



stackforge / **puppet-keystone**

**Make tenant name lookup for keystone user lazy**

Previously, when you managed a keystone user, Puppet would start by loading all information available about each user. Most of this information was loaded with a single call to user-list, but the tenant name required an additional two commands (one to get the tenantId, and an additional one to get the tenant name)

This patch makes the tenant lookup lazy, meaning that those additional calls will only be performed when the tenant attribute is actually used. This will cause a significant performance increase for the case where only a large set of users exists, and only a small fraction of them need to be managed.

Partial-Bug: 1224179
Change-Id: I7141d8e922005ecae33c2c596c9806a03fede53f

**bodepd** authored 8 months ago

1 parent 1

- Only loads tenant info for users being managed by Puppet
- Scales for systems with 1000s of users

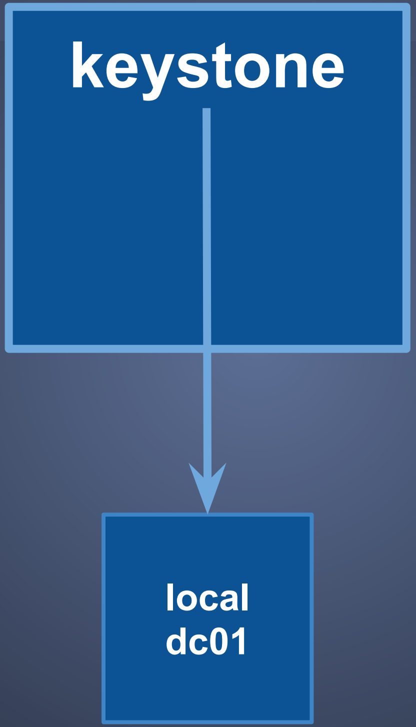
http://x.co/4ZmNb

# Which Domain Controller Do I Use?

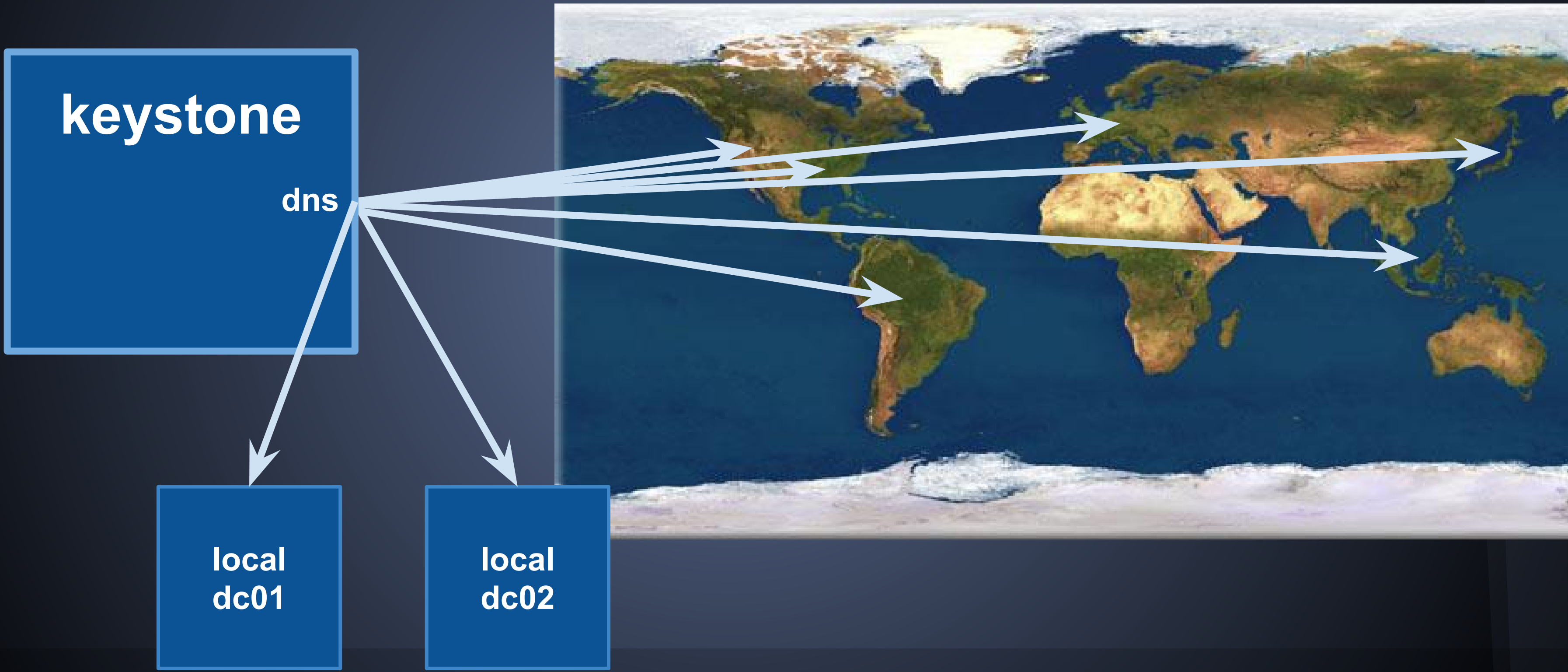
- 10’s of DCs across the company and world
- AD is site aware and can choose a close one
- LDAP is not!

# Two Basic Choices

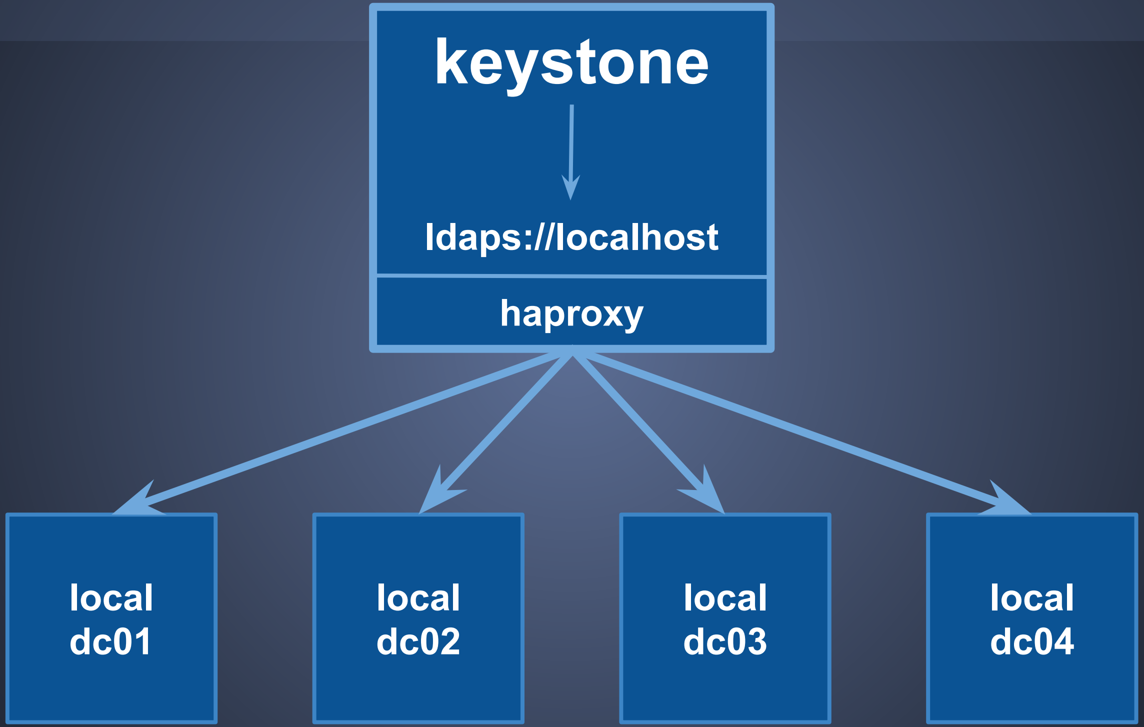
# #1



# #2



# Our Solution



keystone

ldaps://localhost

haproxy

local
dc01

local
dc02

local
dc03

local
dc04

# Some notes about inside the VM

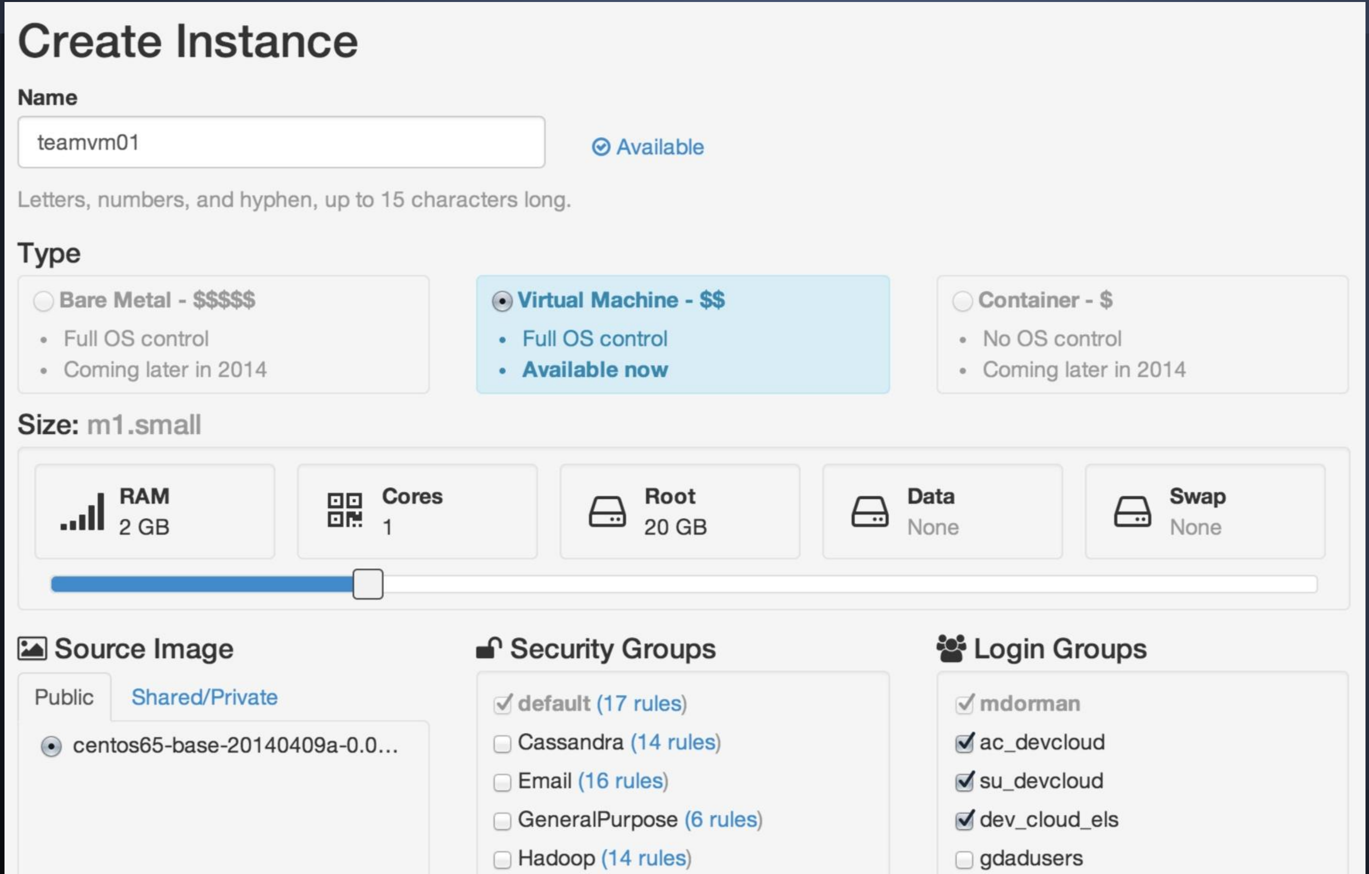
- BeyondTrust PowerBroker on Linux
- Authentication to Linux with AD credentials
- Root password management with CyberArk

- This is good for a bunch of reasons!

http://www.beyondtrust.com/Products/PowerBrokerUnixLinux/

http://www.cyberark.com/product-detail/enterprise-password-vault

# User Access Control with Metadata



## Create Instance

**Name**

teamvm01

✓ Available

Letters, numbers, and hyphen, up to 15 characters long.

### Type

○ **Bare Metal - $$$$$**
- Full OS control
- Coming later in 2014

● **Virtual Machine - $$**
- Full OS control
- **Available now**

○ **Container - $**
- No OS control
- Coming later in 2014

### Size: m1.small

| | | | | |
|---|---|---|---|---|
| **RAM**<br>2 GB | **Cores**<br>1 | **Root**<br>20 GB | **Data**<br>None | **Swap**<br>None |

### Source Image

Public | Shared/Private

● centos65-base-20140409a-0.0...

### Security Groups

☑ **default** (17 rules)

☐ Cassandra (14 rules)

☐ Email (16 rules)

☐ GeneralPurpose (6 rules)

☐ Hadoop (14 rules)

### Login Groups

☑ **mdorman**

☑ ac_devcloud

☑ su_devcloud

☑ dev_cloud_els

☐ gdadusers

# User Access Control with Metadata

```
"meta": {
    "project_name": "user-mdorman",
    "created_by": "mdorman",  ← ssh key configured for this guy
    "login_users": "DC1\\mdorman"
    "login_groups": "DC1\\ac_devcloud,DC1\\su_devcloud,
DC1\\dev_cloud_els",
    "sudo_users": "DC1\\mdorman",
    "sudo_groups": "DC1\\ac_devcloud,DC1\\su_devcloud,
DC1\\dev_cloud_els",
}
```

# User Access Control with Metadata

/etc/login.groups:

```
DC1\ac_devcloud
DC1\dev_cloud_els
DC1\mdorman
DC1\su_devcloud
```

/etc/sudoers.d/openstack-users:

```
mdorman ALL = ALL
%ac_devcloud ALL = ALL
%su_devcloud ALL = ALL
%dev_cloud_els ALL = ALL
```

# This is your Linux, on AD

- Linux VMs get “joined” to the domain
- Name uniqueness requirement
- External clean-up hooks on VM termination

# Thank You!

cjellick@godaddy.com
mdorman@godaddy.com

x.co/ADneLDAP