Plone anbinden an LDAP/Active Directory, Add-ons für weitere Funktionalitäten

Fred van Dijk - Zest - Rotterdam/NL

Sie über Sie

(und ich über mich)

ZEST

Was könnten wir machen

- Geschichte LDAP & Plone
- Zweck von LDAP
- Installieren, Anbindung, Konfiguration
- Add'ons - collective.contentgroup
- SSO - pas.plugins.headers
- wo geht's oft falsch?

"I did this talk before"

- In English - Barcelona - PloneConf 2017
- Erste experimente mit pas.plugins.ldap
- beispiel konfiguration & umsetztung
- **damals wildcard search in users & groups**
- patches/forks benötigt für wildcard search: nicht mehr so

Warnung !!

Live Demo - Vorführ-effekt

Benutzer in Plone

- Lokale benutzerverwaltung
- Globale Rollen
- Lokale rollen
- Benutzer-felder
- Gruppen
- Permissions
- Workflow

LDAP

- Lightweight Directory Access Protocol
  (wieso lightweight)

- Zentrale hierarchische benuter & gruppenverwaltung

- LDAP-server & interfaces nach zB. ActiveDirectory (ad) oder custom servers

- Wer erinnert sich Novell NDS?

- server = procotoll = server

LDAP ähnlich wie Zope/ Plone?

- Hierarchisch
- cn=Billy, ou=Users, dc=my-domain , dc=com
- Objekt-typen (Contenttypes)
- Schemas für Typen & erlaubte Felder (Dexterity?)
- Indexes ...

LDAP in Plone geschichte/stack

- Generelles Problem: fire/configure and forget
- War fur setup A funktioniert, geht nicht für B
- Server-implementierung
- Netz-configuration
- Berechtigungen
- SSL

LDAP in Plone geschichte/stack

- **PAS - Pluggable Authentication Services**
- Products.Ploneldap
- python-ldap
- LDAPMultiplugins
- LDAPUserfolder
- plone.app.ldap

acl_users
stack
stack
stack

alt
alt
alt

ZEST

pas.plugins.ldap

- “Neu-programming” ohne 'alte' abhängkeiten
- entwickelt von by BlueDynamics Alliance
- basiert auf node and node.ext.ldap, virtual node tree
- Version 1.1.0 - 2014
- entstanden aus bda.ldap - 2007 - nicht so neu
- cache LDAP-queris mit memcached - speed vs freshness

pas.plugins.ldap

- Version 1.7.2 - februar 2020
- kein patches/forks für wildcard user/group suchen
  (* verwenden für wildcard ist manchmal notwendig)
- batching, pagination 1000 items limit from server
- leaner & meaner
- ldaps - ignorieren Certificate-checks

ZEST

Selbst experimentieren

- Lokaler LDAP-setup
- mac/linux/windows, aufpassen mit lokalen schon existierenden packages / package manager

- Docker/containers!
  https://github.com/osixia/docker-openldap

Selbst experimentieren

- Jetzt lokal für dieser Talk:
- OS X Brew "install openldap"
  /etc/open/ldapdap/slapd.conf.example
  /usr/local/opt/openldap/libexec/slapd

  > slapd -d1 -f /path/to/slapd.conf -h "ldap://
  127.0.0.1:8389/""

Selbst experimentieren

- Clients - LDAP Abragen

  - Apache Directory Studio
    Wird 'alt': Java 6-8 notwendig

    auf Mac OS X Mojave/catalina
    brew cask install adoptopenjdk
    xattr -d com.apple.quarantine /Library/Java/JavaVirtualMachines/adoptopenjdk-13.0.2.jdk/
    brew cask install apache-directory-studio

- phpLDAPadmin - Docker
  https://github.com/osixia/docker-phpLDAPadmin

Demo lokaler Aufbau

- slapd
- Apache Directory Studio
- my-domain.com
- Bootstrapping: admin-user in slapd.conf

DN, UID, CN, OU, DC

- Verzweigung
  - UID= User ID
  - CN = Common Name
  - OU = OrganizationalUnit
  - DC = Domain Component

Schema's / objectClass

- inetOrgPerson
- organizationalUnit
- groupOfNames

Attributen

- cn
- uid
- mail
- gecos
- fullname
- memberOf
- member

Wass verfügbar ist.....

hängt ab vom Server

ZEST

Bootstrapping lokal

- Apache Directory Studio mit admin password: leer

- Im terminal command line mit ldif files objekte hinzufügen

Terminal: ldapadd, ldapsearch

- Beispiele:
  - ldapadd -H ldap://localhost:8389 -w secret -D cn=Manager,dc=my-domain,dc=com -f jane-user.ldif
  - ldapadd -H ldap://localhost:8389 -w secret -D cn=Manager,dc=my-domain,dc=com -f reviewer-group.ldif
  - ldapsearch -H ldap://localhost:8389 -w secret -D cn=Manager,dc=my-domain,dc=com -b "ou=users,dc=my-domain,dc=com" -s sub "(objectclass=inetOrgPerson)"

Einspielen testdaten ldif

user-jane.ldif

```
dn: uid=jane,ou=users,dc=my-domain,dc=com
changetype: add
uid: jane
cn: fred
sn: 3
objectClass: top
objectClass: posixAccount
objectClass: inetOrgPerson
loginShell: /bin/bash
homeDirectory: /home/jane
uidNumber: 14583106
gidNumber: 14564109
userPassword: {SSHA}j3lBh1Seqe4rqF1+NuWmjhvtAni1JC5A
mail: jane@example.org
gecos: Jane UserToo
```

reviewer-group.ldif

```
dn: cn=reviewers,ou=groups,dc=my-domain,dc=com
changetype: add
objectclass: top
objectclass: groupOfNames
member: cn=jane,ou=users,dc=my-domain,dc=com
cn: reviewers
```

initial.ldif

```
dn: ou=users,ou=my-domain,dc=com
changetype: add
objectclass: top
objectclass: organizationalUnit
ou: users

dn: uid=billy,ou=users,dc=my-domain,dc=com
changetype: add
uid: billy
cn: billy
sn: 3
objectClass: top
objectClass: posixAccount
objectClass: inetOrgPerson
loginShell: /bin/bash
homeDirectory: /home/billy
uidNumber: 14583102
gidNumber: 14564100
userPassword: {SSHA}j3lBh1Seqe4rqF1+NuWmjhvtAni1JC5A
mail: billy@example.org
gecos: Billy User
```

- Importierfolge für testbenutzer über ldifs ist wichtig:
- Wenn der ou=users noch nicht da ist, kann man kein benutzer under ou=users anlegen!

ZEST

Setup mit Plone

- Demo:
- Add'on hinzufügen
- Warten ..... (timeout anbinding default settings)
- konfiguration plugin

Benuter, Gruppen, Rollen

- Many Group/User settings controlpanel
- Suchen mit wildcards: *
- Direct vergeben - Global Sharing
- Direct vergeben - Local Sharing
- Gruppen lokal oder aus LDAP?

Caching

- memcached
- key - value store
- lokaler server, 9277
- freshness vs speed
- Beispiel

Programmierbare installation

- collective.ldapsetup (github.com/collective)

- GenericSetup

- Script für migrieren von plone.app.ldap nach pas.plugins.ldap

collective.contentgroups

Membrane ohne Membrane :-)

Dixit Maurits van Rees:

- Plone PAS plugin for content as groups.
- Use case: create content item that works as a group.
- dexterity behavior
- No Products.membrane, no Products.remember, no dexterity.membrane.
- No separate membrane_catalog.
- Only groups, not users.
- No multiple inheritance, just AccessControl.users.BasicUser.

ZEST

Verwenden für

- Delegieren von ad-hoc Gruppen-verwaltung
- Benutzer & Gruppen kommen aus LDAP
- Oder gruppen sind lokal in Plone verwaltet
- Editors/Sekretariat hat/darf kein zugriff auf LDAP management
- UND Benutzer dürfen auch kein Zugriff haben auf Plone Users/Groups (alles oder nichts)

collective.contentgroups

Demo

ZEST

Wo geht's oft falsch mit LDAP

- Raten - Glücksspiel
- Es git immer schon ein User Directory - customisations
- Raten von Objecten / Attributen
- Authentizierung vs Authorizierung
- DN vs UID vs SAMAccountName (AD)
- Filtern auf Objekten: was kommt zurück:
  sind das vollständige benutzerobjekte?
- Caching (nicht)
- Caching (zu viel)
- Group Members vs MemberOf auf Benutzerobjekt

SSO - Single Sign On

- Authentizierung
- Benutzerdaten noch notwendig <- pas.plugins ldap
- SSO in Plone/PAS - oder 'bevor' Plone
- **pas.plugins.headers**
- (älter: Products.WebserverAuth)

ZEST

SSO - Single Sign On

- **pas.plugins.headers**
- **Komplexität in Plone vermeiden**
- 2020 werden wir's verwenden für samlv2 integrationen:
- Benutzername/UID kommt aus Apache/Nginx
- 'trusted headers'
- Jedes request, oder:
  - header -> session-aufbau ->__ac cookie -> Cookie auth

Danke

@fredvd (Github/Twitter)

f.van.dijk@zestsoftware.nl