SEARCH GUARD

ACTIVE DIRECTORY & LDAP AUTHENTICATION

Search Guard

01.

LDAP VS ACTIVE DIRECTORY

- **LDAP (Lightweight Directory Access Protocol)**

→ “an open, vendor-neutral, industry standard application protocol for accessing and mainta
distributed directory information services over an Internet Protocol (IP) network”

- **Active Directory**

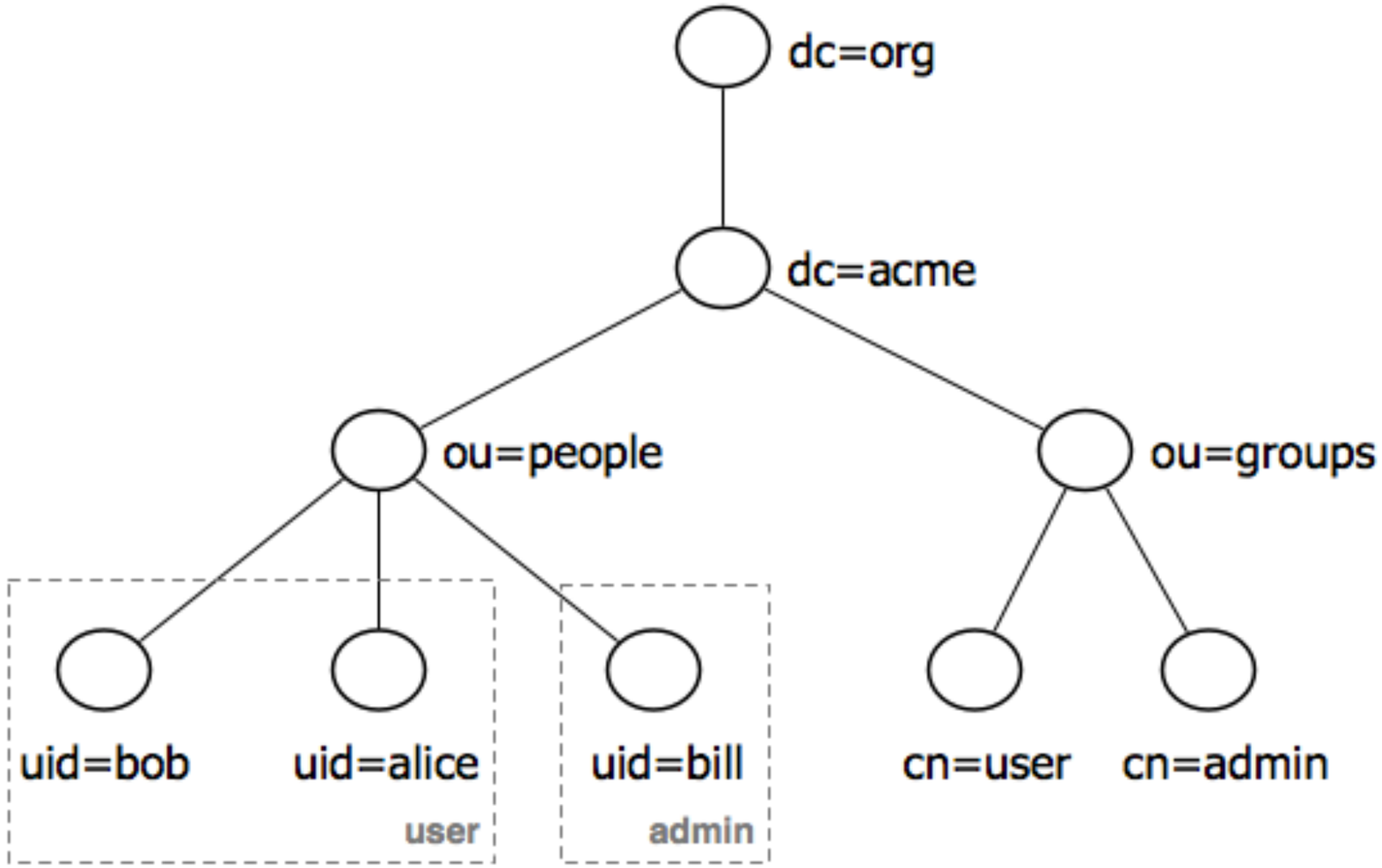
→ “a service that provides LDAP based authentication with Kerberos based authorization.”

- **Both provide a tree-based directory service: “Directory Information Tree” (DIT)**

→ “Directory Information Tree” (DIT)

- **Typically stored users and groups, amongst other objects**
- **Objects identified by their Distinguished Name (DN)**

DIRECTORY INFORMATION TREE



02.

LDAP SUPPORT IN SEARCH GUARD

- **The LDAP backend can be used for**
  - → Authentication (verify user credentials)
  - → Authorization (fetch a users backend roles)
  - → For Active Directory and LDAP likewise
- **One or multiple LDAP servers supported**
  - → For high availability
  - → For connecting to multiple directories
- **In Windows environments often combined with Kerberos**

03.

LDAP AUTHENTICATION BACKEND

```
ldap:
  http_enabled: true
  transport_enabled: true
  order: 0
  http_authenticator:
    type: basic
    challenge: false
  authentication_backend:
    type: ldap
    config:
      enable_ssl: true
      verify_hostnames: true
      hosts:
        - ldap.example.com:636
      bind_dn: cn=admin,dc=example,dc=com
      password: password
      userbase: 'ou=people,dc=example,dc=com'
      # Filter to search for users (currently in the whole subtree beneath userbase)
      # {0} is substituted with the username
      usersearch: '(sAMAccountName={0})'
```

04.

CONNECTION SETTINGS

- **Hosts**
  - → Authentication (verify user credentials)
  - → Authorization (fetch a users backend roles)
  - → For Active Directory and LDAP likewise
- **One or multiple LDAP servers supported**
  - → For high availability
  - → For connecting to multiple directories
- **In Windows environments often combined with Kerberos**

05.

CONNECTION SETTINGS

- **LDAPS and TLS support**
  - → Hostname verification & DNS lookups
  - → Client certificate authentication
  - → Support for a separate root CA, if different from Search Guard root CA
- **Bind settings**
  - → Anonymous bind
  - → bind_dn / password

06.

USER AUTHENTICATION

- **Search Guard performs LDAP queries for user authentication**
- **userbase**

→ Subtree that stores user information, specified by full DN

- **usersearch**

→ LDAP query to find the user

→ {0} is a placeholder and substituted with the user`s name

- **username_attribute**

→ attribute of the LDAP entry that contains the username

→ If not specified, the full DN is used

07.

USER AUTHENTICATION

LDAP Entry:

```
dn: CN=hr_employee,CN=Users,DC=test,DC=local
objectClass: person
cn: AD_hr_employee
distinguishedName: CN=AD_hr_employee,CN=Users,DC=test,DC=local
displayName: AD_hr_employee
memberOf: CN=HumanResources_Employees,OU=Groups,DC=test,DC=local
name: AD_hr_employee
sAMAccountName: hr_employee
```

Configuration:

```
config:
  hosts:
    - ldap.example.com:636
  bind_dn: cn=admin,dc=test,dc=local
  password: password
  userbase: 'CN=Users,DC=test,DC=local'
  # Filter to search for users (currently in the whole subtree beneath userbase)
  # {0} is substituted with the username
  usersearch: '(sAMAccountName={0})'
```

08.

LDAP AUTHORIZATION BACKEND

```
authorization_backend:
  type: ldap # NOT FREE FOR COMMERCIAL USE
  config:
    … (tls, hostnames and bind_dn as before) …
    rolebase: 'ou=groups,dc=example,dc=com'
    # Filter to search for roles (currently in the whole subtree beneath rolebase)
    # {0} is substituted with the DN of the user
    # {1} is substituted with the username
    # {2} is substituted with an attribute from user's directory entry(userroleattribute)
    # Specify the name of the attribute which value should be substituted with {2} above
    userroleattribute: null
    rolesearch: '(uniqueMember={0})'
    # Roles as an attribute of the user entry
    userrolename: memberOf
    # The attribute in a role entry containing the name of that role
    rolename: cn
    # Resolve nested roles transitive (roles which are members of other roles on ...)
    resolve_nested_roles: true
```

09.

USER AUTHORIZATION

- **Connection settings are identical to authentication section**
- **Basic principle for fetching roles is similar to authentication**

→ Configure role subtree

→ Defined LDAP query for retrieving roles

→ Configure the attribute of the LDAP entry that is used as the role name

→ Configure support for nested roles

- **Alternative: Roles as direct user attributes**
- **Both approaches can be combined**

10.

USING THE ROLE SUBTREE

- **rolesearch LDAP query can contain three placeholders**

→ # {0} is substituted with the DN of the user

→ # {1} is substituted with the username

→ # {2} is substituted with an attribute value from the user's directory entry

→ use "userroleattribute" to specify the name of this attribute

- **E.g.: rolesearch: "(uniqueMember={0})"**
- **Configure role name attribute**

→ E.g.: rolename: "cn"

11.

ATTRIBUTE BASED ROLES

- **Roles can be stored as user attributes**

→ Attributes of the LDAP user entry in the user subtree

- **Search Guard can extracts these roles**

→ userrolename: “myroleattribute”

- **Attribute values can be:**

→ DN pointing to an LDAP role

→ This role must exist in the role subtree

→ Arbitrary, non-DN values

→ These values are returned as-is

12.

NESTED ROLES

- **Roles on LDAP can be nested**

→ Roles which are members of roles

- **Search Guard can be configured resolve nested roles**

→ resolve_nested_roles: <truelfalse>

- **Depending on the nesting level, can have a performance impact**

→ One LDAP call for each level

13.

ADVANCED FEATURES

- **Exclude users from role lookups**

→ E.g., service users like logstash or Kibana

- **Exclude roles from nested role lookups**

→ Performance optimization

→ Only resolve nested roles when necessary

→ Wildcards and regular expressions are supported

14.

RESOURCES

- **Search Guard website**

→ https://search-guard.com/

- **Documentation**

→ https://docs.search-guard.com

- **Community Forum**

→ https://groups.google.com/d/forum/search-guard

- **GitHub**

→ https://github.com/floragunncom

SEARCH GUARD

SEND US A MESSAGE:

info@search-guard.com

Search Guard

floragunn GmbH
Tempelhofer Ufer 16
D-10963 Berlin, Germany

E-Mail: info@search-guard.com
Web: search-guard.com

Managing Directors: Claudia Kressin, Jochen Kressin
Registergericht: Amtsgericht Charlottenburg
Registernummer: HRB 147010 B
E-Mail: info@floragunn.com