**Complete RBAC Setup in Jenkins with OpenLDAP**

**Phase 1: OpenLDAP Setup with Users & Groups**

**What is OpenLDAP?**

* **OpenLDAP** is an open-source implementation of the **Lightweight Directory Access Protocol (LDAP)**.
* It’s used to **store and manage users, groups, and access control** in a centralized manner.
* Think of it as a **database for identity management**, which Jenkins (or other systems) can connect to for **authentication** and **authorization**.

**Step 1: Install OpenLDAP on EC2**

**Commands:**

sudo apt-get update

sudo apt-get install slapd ldap-utils -y

* **slapd**: OpenLDAP server daemon.
* **ldap-utils**: Command-line tools for managing and querying LDAP (ldapadd, ldapsearch, etc.).

**Step 2: Configure OpenLDAP (Interactive Setup)**

sudo dpkg-reconfigure slapd

| **Prompt** | **Explanation** | **Input** |
| --- | --- | --- |
| Omit OpenLDAP server configuration? | Select **No** to configure the server manually. | No |
| DNS domain name | This defines your **LDAP domain structure**. For devopsshack.com, the LDAP **base DN** becomes dc=devopsshack,dc=com. | devopsshack.com |
| Organization name | This is a **label** for the organization. | DevOpsShack |
| Admin password | Set a **strong password** for the **LDAP administrator (cn=admin)**. | (your password) |
| Database backend | Select **MDB (Memory-Mapped Database)**, which is the recommended backend. | MDB |
| Remove database when slapd is purged? | Prevents accidental deletion of data when removing OpenLDAP. | No |
| Move old database? | If previous configurations exist, move them aside. | Yes |

**Step 3: Base Directory Information Tree (DIT)**

**What is DIT?**

* **DIT (Directory Information Tree)** is the hierarchical structure of your LDAP database.
* It’s like a **file system hierarchy**, but for **identity data**.
* The **base DN** (Distinguished Name) here is dc=devopsshack,dc=com.

**File: base.ldif**

**Use this set of script if u haven’t created the domain during the slapd config**

*dn: dc=devopsshack,dc=com*

*objectClass: top*

*objectClass: domain*

*dc: devopsshack*

*dn: ou=users,dc=devopsshack,dc=com*

*objectClass: organizationalUnit*

*ou: users*

*dn: ou=groups,dc=devopsshack,dc=com*

*objectClass: organizationalUnit*

*ou: groups*

**If already created then use these**

dn: ou=users,dc=devopsshack,dc=com

objectClass: organizationalUnit

ou: users

dn: ou=groups,dc=devopsshack,dc=com

objectClass: organizationalUnit

ou: groups

| **Element** | **Purpose** |
| --- | --- |
| **dc=devopsshack,dc=com** | The **root** of the LDAP hierarchy. |
| **ou=users** | Organizational Unit (OU) to store **user entries**. |
| **ou=groups** | OU to store **group entries**. |

**Command: Add base DIT**

ldapadd -x -D "cn=admin,dc=devopsshack,dc=com" -W -f base.ldif

* **-D**: Specifies the bind DN (LDAP admin).
* **-W**: Prompts for the admin password.
* **-f**: File to load (base.ldif).

**Step 4: Create Users**

**File: users.ldif**

dn: uid=adminuser,ou=users,dc=devopsshack,dc=com

objectClass: inetOrgPerson

uid: adminuser

sn: Admin

cn: Admin User

userPassword: adminpass

dn: uid=devuser1,ou=users,dc=devopsshack,dc=com

objectClass: inetOrgPerson

uid: devuser1

sn: Developer1

cn: Dev User1

userPassword: devpass1

dn: uid=devuser2,ou=users,dc=devopsshack,dc=com

objectClass: inetOrgPerson

uid: devuser2

sn: Developer2

cn: Dev User2

userPassword: devpass2

dn: uid=viewer1,ou=users,dc=devopsshack,dc=com

objectClass: inetOrgPerson

uid: viewer1

sn: Viewer1

cn: Viewer User1

userPassword: viewerpass1

| **Attribute** | **Purpose** |
| --- | --- |
| **uid** | Unique identifier for the user (used for login). |
| **sn** | Surname (last name). |
| **cn** | Common name (full display name). |
| **userPassword** | Password (in plain text for simplicity, better to hash). |

**Command: Add users**

ldapadd -x -D "cn=admin,dc=devopsshack,dc=com" -W -f users.ldif

**Step 5: Create Groups**

**File: groups.ldif**

dn: cn=jenkins-admins,ou=groups,dc=devopsshack,dc=com

objectClass: groupOfNames

cn: jenkins-admins

member: uid=adminuser,ou=users,dc=devopsshack,dc=com

dn: cn=jenkins-devs,ou=groups,dc=devopsshack,dc=com

objectClass: groupOfNames

cn: jenkins-devs

member: uid=devuser1,ou=users,dc=devopsshack,dc=com

member: uid=devuser2,ou=users,dc=devopsshack,dc=com

dn: cn=jenkins-viewers,ou=groups,dc=devopsshack,dc=com

objectClass: groupOfNames

cn: jenkins-viewers

member: uid=viewer1,ou=users,dc=devopsshack,dc=com

| **groupOfNames** | Allows defining **groups** of users with **member** attributes pointing to user DNs. |

**Command: Add groups**

ldapadd -x -D "cn=admin,dc=devopsshack,dc=com" -W -f groups.ldif

**Step 6: Verify LDAP Directory**

ldapsearch -x -b "dc=devopsshack,dc=com"

* This queries the entire directory tree and **displays all users and groups**.

**Phase 2: Jenkins RBAC with OpenLDAP**

**Step 1: Install LDAP Plugin**

1. Go to:

Manage Jenkins → Manage Plugins → Available

1. Search for:

LDAP Plugin

1. Install and **restart Jenkins**.

**Step 2: Configure LDAP Authentication**

1. Go to:

Manage Jenkins → Configure Global Security

1. Under **Security Realm**, select:

LDAP

**LDAP Connection Fields Explained:**

| **Field** | **Value** | **Explanation** |
| --- | --- | --- |
| **Server** | ldap://3.110.123.68:389 | LDAP server address and port (389 for plain LDAP). |
| **Root DN** | dc=devopsshack,dc=com | The **starting point** in the directory tree for all searches. |
| **User Search Base** | ou=users | Relative location under Root DN where **user entries** reside. |
| **User Search Filter** | (uid={0}) | Filters user logins. {0} gets replaced with the username entered at Jenkins login. |
| **Group Search Base** | ou=groups | Where Jenkins should **search for groups**. |
| **Group Membership Filter** | (member={0}) | How Jenkins finds **group memberships**. {0} gets replaced with the **user's DN** (e.g., uid=devuser1,...). |
| **Manager DN (Bind DN)** | cn=admin,dc=devopsshack,dc=com | The **LDAP admin account** used to query the directory. |
| **Manager Password** | (LDAP admin password) | Password for the **bind account**. |

1. **Test connection** using **adminuser/adminpass**.

**Step 3: Configure Jenkins Authorization (RBAC)**

**Option A: Matrix-based Security (Recommended for LDAP)**

1. Under **Authorization**, select:

Matrix-based security

1. Add **LDAP groups** with **@ prefix**:

| **Identity** | **Permissions** |
| --- | --- |
| @jenkins-admins | ✔️ Administer (all Jenkins permissions) |
| @jenkins-devs | ✔️ Job Read, ✔️ Job Build |
| @jenkins-viewers | ✔️ Job Read |

* **@** prefix tells Jenkins it’s a **group from LDAP**.

**Option B: Role Strategy Plugin (if needed)**

1. Install **Role Strategy Plugin**.
2. Set **Authorization** to **Role-Based Strategy**.
3. Define **roles** like admin, developer, viewer.
4. Assign **LDAP groups** to these roles.

**Files Recap:**

| **File** | **Purpose** |
| --- | --- |
| **base.ldif** | Creates **root DN**, **users OU**, **groups OU**. |
| **users.ldif** | Adds multiple **user entries**. |
| **groups.ldif** | Adds **groups** and **links members**. |

**Jenkins Configuration Summary:**

| **Section** | **Details** |
| --- | --- |
| **Security Realm** | **LDAP** (uses OpenLDAP for authentication) |
| **Authorization** | **Matrix-based security** (RBAC via LDAP groups) |
| **Groups** | jenkins-admins, jenkins-devs, jenkins-viewers |
| **Users** | adminuser, devuser1, devuser2, viewer1 |