# Homelab AD–Linux Integration

**Scope:** Document the design, playbooks, and troubleshooting we used over the last few days to join 10 Rocky/EL9 Linux servers to **CORP.LOCAL** (Windows Server 2022 AD), lock down access with SSSD, and grant controlled sudo via a dedicated AD group.

## 1) Environment & Goals

**Windows / AD** - Domain: corp.local (realm CORP.LOCAL) - DC: ws2022-dc.corp.local - OU structure (minimum used): - OU=Groups,DC=corp,DC=local - OU=Linux,OU=Servers,DC=corp,DC=local (for computer accounts) - Security Groups created: - **linux-admins** (Global, Security) - Users: - **oluadmin** (UPN oluadmin@corp.local) — member of **linux-admins**

**Linux** - ~10 hosts (Rocky/EL9 family) joined to AD. - SSSD + realmd/adcli for domain join. - Login format: %U@corp.local. - Controlled sudo via /etc/sudoers.d/99-linux-admins.

**Objectives** 1. Consistent AD join for all hosts. 2. Ensure time/DNS pre-reqs (chrony, DC resolver). 3. SSSD configured and enabled. 4. Restrict interactive logins to the **linux-admins** AD group. 5. Provide **limited passwordless** sudo to linux-admins (package mgmt + systemd/journal), with a switch to full sudo if needed for lab convenience.

## 2) AD Preparations (Windows side)

These were executed once on the DC (PowerShell / AD module).

# Ensure an OU for groups exists (or use the default CN=Users)  
$DomainDN = "DC=corp,DC=local"  
if (-not (Get-ADOrganizationalUnit -LDAPFilter '(ou=Groups)' -SearchBase $DomainDN -ErrorAction SilentlyContinue)) {  
 New-ADOrganizationalUnit -Name "Groups" -Path $DomainDN -ProtectedFromAccidentalDeletion $false  
}  
  
# Create the security group that will get Linux admin rights  
New-ADGroup -Name "linux-admins" -SamAccountName "linux-admins" -GroupScope Global -GroupCategory Security -Path "OU=Groups,$DomainDN"  
  
# Create (or use existing) user who should be a Linux admin  
New-ADUser -Name "Olu Admin" -SamAccountName "oluadmin" -UserPrincipalName "oluadmin@corp.local" -Enabled $true \  
 -AccountPassword (Read-Host -AsSecureString "Set a password for oluadmin")  
  
# Add members to the group  
Add-ADGroupMember -Identity "linux-admins" -Members "oluadmin"  
  
# Verify  
Get-ADGroupMember "linux-admins" | Select Name,SamAccountName

## 3) Ansible Inventory & Variables

**Group:** ad\_join\_targets — contains the 10 Linux hosts.

**Vars (group\_vars/ad\_join\_targets.yml)**

ad\_domain: corp.local # DNS domain  
ad\_dcs: ["ws2022-dc.corp.local"] # preferred DC list  
ad\_computer\_ou: "OU=Linux,OU=Servers,DC=corp,DC=local" # where to place computer accounts  
  
# Domain-join credentials (reference a Vault var; do not store in clear text)  
ad\_join\_user: Administrator@CORP.LOCAL  
ad\_join\_password: "{{ vault\_ad\_join\_password }}"

**Vault example**

ansible-vault create group\_vars/ad\_join\_targets.vault.yml

vault\_ad\_join\_password: "<secure-password>"

echo '---  
# pull in vault  
' >> group\_vars/ad\_join\_targets.yml

## 4) Playbooks (Linux side)

### 4.1 Domain join & SSSD baseline (playbooks/ad\_domain\_join.yml)

High-level tasks: - Ensure time in sync (chrony enabled), and DNS points to the DC. - Install required packages: realmd adcli sssd sssd-tools oddjob oddjob-mkhomedir samba-common-tools. - authselect to enable mkhomedir on login (EL9): - authselect select sssd with-mkhomedir. - Check current realm membership; join if needed via **adcli** with preferred OU and DC. - Ensure/enable SSSD, confirm realm list and keytab present (/etc/krb5.keytab).

**Common fixes we applied** - SSSD inactive due to missing config: create /etc/sssd/sssd.conf via realmd or template, then systemctl enable --now sssd. - klist -k shows *Permission denied* when run as non‑root (expected); use sudo klist -k. - Kerberos *Client ‘HOST$’ not found* after a failed join: re-run join to refresh the machine account and keytab.

### 4.2 Conventional admin model (playbooks/linux\_admins\_conventional.yml)

Key elements implemented on all joined hosts:

* **SSSD access control**
* [domain/corp.local]  
  access\_provider = simple  
  simple\_allow\_groups = linux-admins@corp.local
* **Sudoers drop-in** at /etc/sudoers.d/99-linux-admins with *limited* passwordless commands:
* Cmnd\_Alias PKG = /usr/bin/dnf, /usr/bin/yum, /usr/bin/apt, /usr/bin/apt-get  
  Cmnd\_Alias SYS = /usr/bin/systemctl, /usr/bin/journalctl  
  %linux-admins@corp.local ALL=(ALL:ALL) NOPASSWD: PKG, SYS
* Switch to full sudo for the lab by replacing content with: %linux-admins@corp.local ALL=(ALL:ALL) NOPASSWD: ALL
* **Service state**: sssd enabled and started on all hosts.
* **Verification** tasks in the playbook:
  + systemctl is-active sssd
  + realm list (summary)
  + sudo -lU oluadmin@corp.local | head
  + sudo -n true && echo SUDO\_OK

## 5) Verification & Troubleshooting

**Quick checks (per host)**

adcli testjoin -D corp.local && echo JOIN\_OK || echo NOT\_JOINED  
realm list  
systemctl is-active sssd && systemctl is-enabled sssd  
sudo klist -k | head -3  
getent passwd "oluadmin@corp.local"  
getent group "linux-admins@corp.local"  
sudo -lU "oluadmin@corp.local" | head  
sudo -n true && echo SUDO\_OK

**Common issues we resolved** - **SSSD inactive (dead)** with ConditionPathExists=/etc/sssd/sssd.conf: create the domain section and restart; ensure permissions 0600 owned by root. - **Permission denied while starting keytab scan**: run klist -k as root. - **Couldn’t enumerate keytab** immediately after join: re-run join or systemctl restart sssd to refresh; confirm /etc/krb5.keytab existence. - **Jinja recursion error** in Ansible (maximum recursion depth exceeded): avoid self-referential defaults (we replaced ad\_domain → internal ad\_dns\_domain).

## 6) Final State (end of engagement)

* All 10 Linux hosts are joined to **CORP.LOCAL** and show consistent realm list summaries.
* sssd is **active** and **enabled** on every host.
* Interactive logins are **restricted** to AD group **linux-admins** via simple\_allow\_groups.
* **oluadmin@corp.local** (member of linux-admins) has **passwordless limited sudo** (dnf/apt + systemctl/journalctl).
* /etc/sudoers.d/99-linux-admins validated via visudo -cf.

## 7) Runbooks

### 7.1 Onboard a new Linux host

1. Put the host in the ad\_join\_targets group.
2. Ensure it resolves the DC and has time sync:
   * /etc/resolv.conf → DC IP; chronyd enabled.
3. Run join + baseline:

* ansible-playbook -i inventory playbooks/ad\_domain\_join.yml -l <newhost>  
  ansible-playbook -i inventory playbooks/linux\_admins\_conventional.yml -l <newhost>

1. Verify (adcli testjoin, realm list, sudo -lU).

### 7.2 Rotate domain-join password

* Update group\_vars/ad\_join\_targets.vault.yml → vault\_ad\_join\_password.
* Re-run join task with --tags join (or full playbook) to refresh keytabs if needed.

### 7.3 Refresh machine keytab (if computer account changed)

sudo adcli update --domain=corp.local --login-user=Administrator@CORP.LOCAL  
sudo systemctl restart sssd

### 7.4 Emergency / Break-glass (optional)

* Temporarily add Domain Admins@corp.local to simple\_allow\_groups and/or sudoers; revert after maintenance.

## 8) Files & Commands Reference

**Playbooks** - playbooks/ad\_domain\_join.yml — join + prerequisites, sssd enable - playbooks/linux\_admins\_conventional.yml — access control + sudo + verification

**Ad‑hoc health sweep**

ansible ad\_join\_targets -b -m shell -a '  
 echo "SSSD: $(systemctl is-active sssd) / $(systemctl is-enabled sssd)" ;  
 test -s /etc/sssd/sssd.conf && echo "sssd.conf: present" || echo "sssd.conf: MISSING" ;  
 sudo klist -k | head -n3 ;  
 realm list | awk "NR==1,/^$/" ;  
 getent group "linux-admins@corp.local" || true ;  
 sudo -lU "oluadmin@corp.local" | head -n 20 ;  
 sudo -n true && echo SUDO\_OK || echo SUDO\_FAIL  
'

## 9) Notes & Lessons Learned

* Ensure **DNS** and **time** are correct *before* joining — most failures traced back here.
* Use a **dedicated AD group** for Linux admin rights instead of Domain Admins; easier to audit and revoke.
* Keep Ansible variables **non‑recursive**; avoid var: "{{ var | default(...) }}" with the same name on both sides.
* Validate sudoers with visudo -cf in the playbook to prevent lockouts.

## 10) Next Ideas (optional)

* Add MFA or SSH‑CA integration for admin logins.
* Ship SSSD & sudo logs to a central syslog (rsyslog/Elastic) for auditing.
* Create a small **CI** that runs the verification playbook nightly and alerts on drift.

*Prepared for the homelab by documenting the 3–5 day integration sprint. Feel free to annotate this doc; we can add screenshots or export to PDF on request.*