Linux und Microsoft AD

Anleitungen rund um Active Directory

- AD Beitritt Ubuntu LTS
- Zertifikat mit AD erzeugen
- Linux System zu AD hinzufügen
- AD Beitritt Open Suse 16

AD Beitritt Ubuntu LTS

Hier ist eine **komplette, kopierfertige Anleitung für Ubuntu (22.04 LTS / 24.04 LTS)** mit **SSSD + Realmd** für **AD-Integration inkl. Offline-Login** – **ohne Winbind**, da SSSD zuverlässiger für Offline-Szenarien ist.

□□ Ubuntu AD-Integration mit Offline-Login (SSSD + Realmd)

Voraussetzungen:

- Ubuntu **22.04 LTS oder 24.04 LTS** (frisch installiert)
- Domänenname: `BEISPIEL.DOMAENE.LAN` (ersetzen!)
- AD-Admin-Benutzer: `Administrator` (oder anderer berechtigter Benutzer)
- Netzwerkverbindung zum **Domänencontroller (DC)**

1□ Pakete installieren & System vorbereiten

```
# System aktualisieren
sudo apt update && sudo apt upgrade -y

# Benötigte Pakete installieren
sudo apt install -y realmd sssd sssd-tools adcli krb5-user packagekit oddjob oddjob-mkhomedir pam_krb5 libnss-sss libpam-sss ntp

# Hostname setzen (muss im AD eindeutig sein)
sudo hostnamectl set-hostname ubuntu-client1.beispiel.domäne.lan
```

2□ Kerberos konfigurieren

```
sudo vi /etc/krb5.conf
```

Inhalt ersetzen durch:

```
[libdefaults]
    default_realm = BEISPIEL.DOMAENE.LAN
    dns_lookup_realm = true
    dns_lookup_kdc = true
    ticket_lifetime = 24h
    renew_lifetime = 7d
    forwardable = true
    rdns = false

[realms]
    BEISPIEL.DOMAENE.LAN = {
        kdc = dc1.beispiel.domäne.lan
        admin_server = dc1.beispiel.domäne.lan
    }

[domain_realm]
    .beispiel.domäne.lan = BEISPIEL.DOMAENE.LAN
    beispiel.domäne.lan = BEISPIEL.DOMAENE.LAN
```

---

3 DNS prüfen & anpassen

```
# Testen, ob der DC erreichbar ist
ping dc1.beispiel.domäne.lan
nslookup dc1.beispiel.domäne.lan

# Falls DNS nicht funktioniert, manuell anpassen:
sudo vi /etc/resolv.conf
```

Inhalt ersetzen durch:

```
nameserver 192.168.1.10  # IP des DC
search beispiel.domäne.lan
```

→ **Wichtig:** Falls `systemd-resolved` aktiv ist:

```
sudo systemctl disable --now systemd-resolved
sudo systemctl enable --now NetworkManager
```

---

4 Domäne mit `realmd` beitreten

```
# Domäne entdecken (Test)
sudo realm discover BEISPIEL.DOMAENE.LAN

# Domäne beitreten (AD-Admin-Passwort eingeben)
sudo realm join -U Administrator BEISPIEL.DOMAENE.LAN --verbose
```

→ **Erwartete Ausgabe:** `Successfully enrolled machine in realm`

---

5 SSSD für Offline-Login konfigurieren

```
sudo vi /etc/sssd/sssd.conf
```

Folgende Konfiguration einfügen:

```
[sssd]
domains = BEISPIEL.DOMAENE.LAN
config_file_version = 2
services = nss, pam, sudo, ssh

[domain/BEISPIEL.DOMAENE.LAN]
ad_domain = BEISPIEL.DOMAENE.LAN
krb5_realm = BEISPIEL.DOMAENE.LAN
realmd_tags = manages-system joined-with-adcli
cache_credentials = true          # Offline-Login aktivieren!
krb5_store_password_if_offline = true
default_shell = /bin/bash
ldap_id_mapping = true
use_fully_qualified_names = false  # Kurze Benutzernamen (z. B. "user" statt "user@domäne.lan")
fallback_homedir = /home/%u
access_provider = ad
```

```
entry_cache_timeout = 1209600     # Cache-Gültigkeit: 14 Tage
account_cache_expiration = 14     # Account-Cache: 14 Tage
```

Berechtigungen setzen:

```
sudo chmod 600 /etc/sssd/sssd.conf
```

---

6 PAM für Home-Verzeichnisse & Offline-Login anpassen

```
# Automatische Home-Verzeichnisse aktivieren
sudo pam-auth-update --enable mkhomedir

# Manuell prüfen (falls nötig)
sudo vi /etc/pam.d/common-session
```

Folgende Zeile hinzufügen (falls nicht vorhanden):

```
session required pam_mkhomedir.so skel=/etc/skel umask=0022
```

---

7 Dienste neu starten

```
sudo systemctl restart sssd
sudo systemctl enable --now oddjobd
```

---

8 Test: Online-Anmeldung (Cache füllen)

```
# Testbenutzer anmelden (ersetze "testuser" mit einem AD-Benutzer)
su - testuser
exit

# Kerberos-Ticket prüfen
```

```
klist
```

→ **Erwartet:** Ein gültiges Ticket für `testuser@BEISPIEL.DOMAENE.LAN`.

---

9 Test: Offline-Login

```
# Netzwerk trennen (z. B. Kabel ziehen oder WLAN deaktivieren)
sudo systemctl stop NetworkManager

# Offline-Login testen
su - testuser  # Sollte funktionieren!
id testuser    # Sollte Benutzerdaten anzeigen
klist -l       # Sollte gecachte Tickets zeigen
```

Fehlersuche (falls nötig)

Problem	Befehl zur Diagnose
Anmeldung fehlgeschlagen	`journalctl -u sssd -f`
Keine Benutzer auflösbar	`getent passwd testuser`
Kerberos-Fehler	`kinit testuser` (manueller Test)
DNS-Probleme	`dig dc1.beispiel.domäne.lan`
SSSD-Cache löschen	`sudo systemctl stop sssd; sudo rm -f /var/lib/sss/db/cache_*.ldb; sudo systemctl start sssd`

Fertig!

- Ubuntu ist jetzt in der **AD-Domäne** und unterstützt **Offline-Login**.
- **Wichtig:** Jeder Benutzer muss sich **mindestens einmal online anmelden**, bevor er offline arbeiten kann.

Optional: Automatische Ticket-Erneuerung

Falls Kerberos-Tickets zu schnell ablaufen:

```
sudo vi /etc/krb5.conf
```

Folgende Zeilen anpassen:

```
[libdefaults]
    ticket_lifetime = 24h
    renew_lifetime = 7d
    forwardable = true
```

Dann **SSSD neu starten**:

```
sudo systemctl restart sssd
```

□□ Warum SSSD statt Winbind?

Kriterium	SSSD	Winbind
Offline-Login	□ Zuverlässig (cached credentials)	□ Oft problematisch
Kerberos-Integration	□ Voll unterstützt	⚠ Teilweise manuelle Konfiguration nötig
Performance	□ Schneller (lokaler Cache)	⚠ Langsamer (mehr DC-Abfragen)
Gruppenrichtlinien (GPO)	□ Nicht unterstützt	□ Teilweise möglich
Einfachheit	□ `realmd` macht viel automatisch	⚠ Manuelle Konfiguration nötig

---

□□ Fazit

- **SSSD + Realmd** ist die **beste Wahl für Ubuntu mit Offline-Login**.
- **Winbind** ist nur nötig, wenn **GPOs** oder **Samba-Freigaben** im Vordergrund stehen.
- **Smartcards?** → Dann zusätzlich `pam_pkcs11` installieren und konfigurieren.

Falls du **Samba-Freigaben** oder **Gruppenrichtlinien** brauchst, lass es mich wissen – dann ergänze ich die Anleitung! □□

Zertifikat mit AD erzeugen

Wenn du eine **klassische AD-Zertifizierungsstelle** hast und **nur mit Windows-Bordmitteln** ein Zertifikat anfordern willst, geht das sehr stabil mit **certreq + certutil + openssl**. Damit bekommst du auch die drei Dateien (`.crt`, `.key`, `.chain.pem`), die Nginx Proxy Manager erwartet.

□□ Schritt-für-Schritt mit Windows Bordmitteln

1. Request-Datei erstellen

Erstelle eine INF-Datei, z. B. `request.inf`:

```
[Version]
Signature="$Windows NT$"

[NewRequest]
Subject = "CN=app.contoso.com"
KeySpec = 1
KeyLength = 2048
Exportable = TRUE
MachineKeySet = TRUE
SMIME = FALSE
PrivateKeyArchive = FALSE
UserProtected = FALSE
UseExistingKeySet = FALSE
ProviderName = "Microsoft RSA SChannel Cryptographic Provider"
ProviderType = 12
RequestType = PKCS10
KeyUsage = 0xa0

[Extensions]
```

```
2.5.29.17 = "{text}"
_continue_ = "dns=app.contoso.com&"
_continue_ = "dns=www.app.contoso.com"

[EnhancedKeyUsageExtension]
OID=1.3.6.1.5.5.7.3.1 ; Server Authentication
```

2. CSR erzeugen

```
certreq -new request.inf request.csr
```

3. Zertifikat bei der AD-CA beantragen

Falls dein Client Mitglied der Domäne ist und du Berechtigungen hast:

```
certreq -submit -config "CA-SERVER\CA-NAME" request.csr cert.cer
```

- `CA-SERVER\CA-NAME` ist deine AD-CA (findest du in der CA-Konsole).
- Ergebnis: `cert.cer` (dein Serverzertifikat).
- Private Keys liegen im Windows-Keystore.

4. In PFX exportieren

Zuerst Zertifikat in den Store importieren:

```
certreq -accept cert.cer
```

Dann per `certutil` mit privatem Schlüssel exportieren:

```
certutil -exportPFX -p SuperGeheimesPasswort My "CN=app.contoso.com" app.pfx
```

Jetzt hast du alles in `app.pfx`.

5. Aufsplitten in 3 Dateien

Windows kann kein `.pem` direkt erzeugen → hier kommt **OpenSSL** ins Spiel (kannst du von slproweb.com installieren).

```
# privater Schlüssel extrahieren
openssl pkcs12 -in app.pfx -nocerts -out privkey.pem -nodes -password pass:SuperGeheimesPasswort

# Zertifikat extrahieren
openssl pkcs12 -in app.pfx -clcerts -nokeys -out cert.pem -password pass:SuperGeheimesPasswort

# Kette (Root + Intermediate) extrahieren
openssl pkcs12 -in app.pfx -cacerts -nokeys -out chain.pem -password pass:SuperGeheimesPasswort
```

Damit hast du:

- `privkey.pem` → der private Schlüssel
- `cert.pem` → dein Serverzertifikat
- `chain.pem` → die Zwischen-/Root-Zertifikate

Diese drei Dateien kannst du dann im **Nginx Proxy Manager** unter „Custom SSL Certificate“ hochladen.

Zusammenfassung

- Mit **`certreq`** → CSR erzeugen und Zertifikat von AD CS holen
- Mit **`certutil`** → in PFX exportieren
- Mit **`openssl`** → in die drei Dateien (`privkey.pem`, `cert.pem`, `chain.pem`) zerlegen

Das ist der „saubere“ Microsoft-Weg, kein Gefrickel.

Soll ich dir ein komplettes **PowerShell-Skript** bauen, das all diese Schritte (CSR → AD-CA → PFX → PEMs) automatisch abwickelt? Dann müsstest du nur einmal CN/Domains eintragen und hättest am Ende direkt die 3 Dateien im gewünschten Ordner.

Linux System zu AD hinzufügen

Allgemeines

Es gibt zwar eine relativ einfache Methode einer AD Domäne beizutreten. Dies ist hier sehr gut beschrieben. Wer jedoch die volle Funktionalität von AD nutzen möchte, kommt bei der Methode sehr schnell an seine Grenzen. Unter Open Suse 15.x / Tubleweed ist das kein Problem. Yast auf und Domäne beitreten wählen. Etwas warten, die Admin Benutzerdaten eingeben und schon ist die Sache erledigt. Bei Ubuntu ist die Sache nicht ganz so simpel. Dieser Artikel bezieht sich auf Ubuntu 20.04 und 21.04. Sollte so aber auch in späteren Versionen funktionieren.

Pakete installieren

```
sudo apt-get install -y krb5-user libpam-krb5 winbind  samba  smbclient libnss-winbind libpam-winbind
```

Während der Installation wird nach dem Realm für Kerberos 5 gefragt. Dieser ist der Domänen Name von Active Directory. Wenn der Rechner als **rechner01.ad.einedomain.org** lautet ist der Realm **einedomain.org**.

Kerberos einrichten

In der Datei **/etc/krb5.conf** mit folgendem Inhalt versehen, wobei **ad.eigenedomain.org** wieder der bei der Installation eingegebene Name ist. Bitte auch auf die Großschreibung achten, da MIT Kerberos da ein kleinwenig heikel ist:

```
[logging]
    default = FILE:/var/log/krb5.log


[libdefaults]
```

```
    ticket_lifetime = 24000
    clock_skew = 300
    default_realm = AD.EIGENEDOMAIN.ORG

[realms]
    EXAMPLE.COM = {
        kdc = pdc.eigenedomain.org:88
        admin_server = pdc.eigenedomain.org:464
        default_domain = AD.EIGENEDOMAIN.ORG
    }

[domain_realm]
    .ad.eigenedomain.org = AD.EIGENEDOMAIN.ORG
    ad.eigenedomain.org = AD.EIGENEDOMAIN.ORG
```

Sobald die Datei gespeichert ist kann dies getestet werden. Auch hier wieder auf die groß geschriebene Domain achten:

```
kinit administrator@AD.EIGENEDOMAIN.ORG
```

Sollte jetzt keine Ausgabe kommen, war die Operation erfolgreich. Wenn jedoch eine Meldung wie die kommt:

```
kinit: KDC-Antwort entsprach nicht den Erwartungen bei Anfängliche Anmeldedaten werden geholt.
```

ist etwas schief gelaufen. In den meisten Fällen ist der Fehler dann entweder ein Fehler in den Rechnernamen oder dass der Realm entweder bei kinit oder in der Konfiguration nicht groß geschrieben waren.

Samba konfigurieren

Als nächstes bringen wird die bestehende Beispielkonfiguration in Sicherheit (diese behalte ich ganz gerne, weil in den Dateien sehr viel Dokumentiert ist):

```
sudo mv /etc/samba/smb.conf /etc/samba/smb.conf.sample
```

Nun legen wird die Datei **/etc/samba/smb.conf** neu an:

```
[global]
  security = ads
```

```
realm = AD.EIGENEDOMAIN.ORG
password server = IPADRESSE     #IP des Domain Controllers
workgroup = AD
idmap uid = 10000-20000
idmap gid = 10000-20000
winbind enum users = yes
winbind enum groups = yes
winbind cache time = 10
winbind use default domain = yes
template homedir = /home/%U
template shell = /bin/bash
client use spnego = yes
client ntlmv2 auth = yes
encrypt passwords = yes
restrict anonymous = 2
domain master = no
local master = no
preferred master = no
os level = 0
```

Jetzt kann der Rechner der Domäne hinzugefügt werden:

```
sudo net ads join -U administrator
```

Sollte hier die Fehlermeldung kommen, dass der DNS Eintrag nicht aktualisiert werden konnte, ist das nicht schlimm. Hierzu die Datei **/etc/hosts** aktualiesieren, dass die den vollen Domain Namen enthält:

```
192.168.1.2         rechner01.eigenedomain.org       rechner01
```

Nun müssen die RPC Dienste noch verbunden werden:

```
sudo net rpc join -U administrator
```

Um die Änderungen zu übernehmen muss nun Winbind neu gestartet werden:

```
sudo systemctl restart winbind
```

Um zu prüfen ob alles wie gewünscht funktioniert können nun anfragen an den AD Server gestellt werden. Mit volgendem Befehl werden alle Benutzer aufgelistet:

```
wbinfo -u
```

System Auth konfigurieren

Kerberos und Winbind laufen jetzt. Jetzt muss das System noch dazu überredet werden, die Auth über Winbind laufen zu lassen. Dazu muss die Datei **/etc/nsswitch.conf** bearbeitet werden.

```
passwd:         compat winbind
group:          compat winbind
shadow:         compat
```

Nun muss Winbind wieder neu gestartet werden:

```
sudo systemctl restart winbind
```

PAM Konfiguration

Zuletzt muss ein Domänen Benutzer noch in die Lage versetzt werden mit dem System zu interagieren. Hierzu folgende Zeile in die Datei **/etc/security/group.conf** einfügen:

```
* ; * ; * ; Al0000-2400 ; floppy, audio, cdrom, video, usb, plugdev, users
```

Nun noch dafür sorgen, dass die Home Verzeichnisse angelegt werden. dazu folgenden Befehl ausführen und den Eintrag "Create home directory on login" aktivieren:

```
sudo pam-auth-update
```

Die folgenden abschließenden Befehle sorgen für die Übernahme der letzten Änderungen:

```
sudo systemctl stop winbind
sudo systemctl restart smbd
sudo systemctl start winbind
```

Einhängen von Netzlaufwerken mit Benutzerrechten

Dazu müssen noch weitere Pakete installiert werden:

```
sudo apt-get install libpam-mount cifs-utils
```

Abschliesende Arbeiten

Damit Domänenbenutzer **sudo** verwenden können, müssen diese entweder in die entsprechende Gruppe, oder es muss eine Datei im Ordner **/etc/sudoers.d/** angelegt werden. Ein Beispiel:

```
%wksadmins      ALL=(ALL:ALL)     ALL
"%domain admins"    ALL= (ALL:ALL) ALL
```

Die WKSAdmins Gruppe verwende ich um den Benutzern auf den lokalen Maschinen Admin Rechte zu geben ohne, dass sie diese in der Domain hätten ;).

Auch wenn das nicht unbedingt notwendig ist, sollte nun der Rechner neu gestartet werden. Dann sollten die Domänen Benutzer in der Lage sein, sich anzumelden. Ein guter Artikel zu dem Thema kann hier gefunden werden.

AD Beitritt Open Suse 16

Hier ist eine **kompakte, kopierfertige Anleitung** für die **Domänenintegration mit Offline-Login** unter **openSUSE Leap 16 / SLE 16**. Einfach **Schritt für Schritt abarbeiten** – von der Installation bis zum Test der Offline-Anmeldung.

□□ Anleitung: openSUSE Leap 16 in Active Directory einbinden (mit Offline-Login)

Voraussetzungen:

- Domänenname: `BEISPIEL.DOMAENE.LAN` (ersetzen!)
- Domänen-Admin: `Administrator` (oder anderer Berechtigter)
- Netzwerkverbindung zum Domänencontroller (DC)

1□ Pakete installieren

```
sudo zypper refresh
sudo zypper install --no-confirm realmd sssd sssd-ad sssd-tools adcli krb5-client oddjob oddjob-mkhomedir
pam_krb5 pam_winbind samba-client
```

2□ Hostname und DNS prüfen

```
# Hostname setzen (muss im AD eindeutig sein)
sudo hostnamectl set-hostname client1.beispiel.domäne.lan

# DNS prüfen (muss den DC auflösen können)
ping dc1.beispiel.domäne.lan
nslookup dc1.beispiel.domäne.lan
```

→ Falls DNS nicht funktioniert:

```
sudo vi /etc/resolv.conf
```

Eintrag hinzufügen:

```
nameserver 192.168.1.10  # IP des DC
search beispiel.domäne.lan
```

3️⃣ Domäne mit `realmd` beitreten

```
# Domäne entdecken (Test)
sudo realm discover BEISPIEL.DOMAENE.LAN

# Domäne beitreten (Passwort des AD-Admins eingeben)
sudo realm join -U Administrator BEISPIEL.DOMAENE.LAN --verbose
```

→ **Erwartete Ausgabe:** `Successfully enrolled machine in realm`

4️⃣ SSSD für Offline-Login konfigurieren

```
sudo vi /etc/sssd/sssd.conf
```

Folgende Zeilen anpassen/ergänzen:

```
[sssd]
domains = BEISPIEL.DOMAENE.LAN
config_file_version = 2
services = nss, pam, sudo, ssh

[domain/BEISPIEL.DOMAENE.LAN]
ad_domain = BEISPIEL.DOMAENE.LAN
krb5_realm = BEISPIEL.DOMAENE.LAN
realmd_tags = manages-system joined-with-adcli
cache_credentials = true          # Offline-Login aktivieren
```

```
krb5_store_password_if_offline = true
default_shell = /bin/bash
ldap_id_mapping = true
use_fully_qualified_names = false  # Anmeldung mit "benutzername" statt "benutzername@domäne.lan"
fallback_homedir = /home/%u
access_provider = ad
entry_cache_timeout = 1209600     # Cache-Gültigkeit: 14 Tage (in Sekunden)
account_cache_expiration = 14     # Account-Cache: 14 Tage
```

Berechtigungen setzen:

```
sudo chmod 600 /etc/sssd/sssd.conf
```

5️⃣ PAM für Home-Verzeichnis und Offline-Login konfigurieren

```
# Automatische Erstellung von Home-Verzeichnissen aktivieren
sudo pam-config --add --mkhomedir

# PAM-Konfiguration prüfen
sudo vi /etc/pam.d/common-session
```

Folgende Zeile hinzufügen (falls nicht vorhanden):

```
session required pam_mkhomedir.so skel=/etc/skel umask=0022
```

6️⃣ Dienste neu starten

```
sudo systemctl restart sssd
sudo systemctl enable --now oddjobd
```

7️⃣ Test: Online-Anmeldung (Cache füllen)

```
# Testbenutzer anmelden (ersetze "testuser" mit einem AD-Benutzer)
su - testuser
exit

# Kerberos-Ticket prüfen
klist
```

→ **Erwartet:** Ein gültiges Ticket für `testuser@BEISPIEL.DOMAENE.LAN`.

---

8 Test: Offline-Login

```
# Netzwerk trennen (z. B. Kabel ziehen oder WLAN deaktivieren)
sudo systemctl stop NetworkManager

# Offline-Login testen
su - testuser  # Sollte funktionieren!
id testuser    # Sollte Benutzerdaten anzeigen
klist -l       # Sollte gecachte Tickets zeigen
```

---

9 Fehlersuche (falls nötig)

Problem	Befehl zur Diagnose
Anmeldung fehlgeschlagen	`journalctl -u sssd -f`
Keine Benutzer auflösbar	`getent passwd testuser`
Kerberos-Fehler	`kinit testuser` (manueller Test)
DNS-Probleme	`dig dc1.beispiel.domäne.lan`

Wichtige Dateien & Befehle im Überblick

Zweck	Datei/Befehl
SSSD-Konfiguration	`/etc/sssd/sssd.conf`
Kerberos-Einstellungen	`/etc/krb5.conf`

Zweck	Datei/Befehl
PAM-Konfiguration	`/etc/pam.d/common-session`
Cache löschen	`sudo systemctl stop sssd; sudo rm -f /var/lib/sss/db/cache_*.ldb; sudo systemctl start sssd`
Ticket-Cache prüfen	`klist -l`
Benutzerdaten prüfen	`id testuser`

Fertig!

- Der Rechner ist jetzt in der Domäne und unterstützt **Offline-Login**.
- **Wichtig:** Jeder Benutzer muss sich **mindestens einmal online anmelden**, bevor er offline arbeiten kann.

Optional: Automatische Ticket-Erneuerung konfigurieren

Falls Kerberos-Tickets zu schnell ablaufen:

```
sudo vi /etc/krb5.conf
```

Folgende Zeilen anpassen:

```
[libdefaults]
    ticket_lifetime = 24h
    renew_lifetime = 7d
    forwardable = true
```

Dann **SSSD neu starten**:

```
sudo systemctl restart sssd
```

Tipp: Falls **Smartcards** oder **Zertifikatsauthentifizierung** benötigt werden, muss zusätzlich `pam_pkcs11` oder `sssd` mit `certificate`-Backend konfiguriert werden. Lass es mich wissen, falls du das brauchst!

Viel Erfolg!