Migration Univention Corporate Server & Kopano nach Zamba AD & Mailcow

sysops · Technische Migrationsdokumentation

UCS / Kopano → Zamba AD / mailcow / Proxmox VE · Überarbeitete Fassung: November 2026

UCS und Kopano migrieren: Zamba AD, Mailcow und Proxmox VE

Inhaltsverzeichnis

1. **Vorteile der Migration**
2. **Systemvoraussetzungen**
3. **Übersicht der Migration**
4. **Zamba Active Directory als zweiten Domain Controller installieren**
5. **DNS auf den neuen Domain Controller umstellen**
6. **Mailcow installieren**
7. **Grundkonfiguration von Mailcow**
8. **E-Mails von Kopano nach Mailcow migrieren**
9. **Kalender aus Kopano exportieren**
10. **Kontakte exportieren und importieren**
11. **Aliase und Weiterleitungen übernehmen**
12. **Mailcow produktiv schalten**
13. **Abschluss der Migration**
14. **Proxmox Mail Gateway mit Mailcow verbinden**
15. **Mailpiler als Mailarchiv integrieren**
16. **Doppelte Mails bei mehrfacher Kopano-Synchronisation vermeiden**
17. **Optionale DNS-Einträge für Autokonfiguration**

18. **Fazit**
19. **Passende Schulungen**
20. **Optional: Samba AD Health Check**

Praxisanleitung: Univention Corporate Server (UCS) und Kopano ablösen. Diese Schritt-für-Schritt-Dokumentation zeigt die Migration einer bestehenden UCS-Active-Directory-Domäne auf einen Samba-basierten Zamba-Domain-Controller sowie den Umstieg von Kopano auf mailcow mit SOGo. Dazu gehören DNS, LDAP, IMAP, Kalender, Kontakte, FSMO-Rollen und die produktive Inbetriebnahme auf Proxmox VE.

Mit **Zamba Active Directory**, **Mailcow**, **Proxmox VE**, optional **Proxmox Mail Gateway** und **Mailpiler** lässt sich eine schlanke, moderne Open-Source-Infrastruktur aufbauen, die schneller wartbar ist und sich ideal für kleine und mittlere Unternehmen eignet.

> Die neue Architektur kann Verwaltung und Updates vereinfachen. Der tatsächliche Aufwand hängt von Installation, Abhängigkeiten und Betriebsprozessen ab.
>
> Kunden, die Keycloak oder eine vollständig vom Hersteller supportete UCS-Umgebung benötigen, sollten weiterhin bei Univention bleiben.

Passende Kurse finden Sie auf cloudistboese.de unter den Themen Mailcow, Domain Controller ablösen und Alles ohne Windows.

Vorteile der Migration

- Univention UCS kann in kleinen Umgebungen ersetzt werden
- Active Directory bleibt mit Samba/Zamba erhalten
- Mailcow ersetzt Kopano als moderne Groupware
- Updates werden deutlich einfacher und schneller
- Proxmox VE ermöglicht Snapshots, Backups und einfache Wiederherstellung
- LDAP-Anbindung für Mailcow, Mailpiler und weitere Dienste bleibt möglich
- Die Umgebung wird schlanker, transparenter und besser wartbar

Systemvoraussetzungen

- Bestehender Univention Corporate Server im AD-Modus
- Kopano oder vergleichbare Groupware mit IMAP-Zugriff
- Proxmox VE 8.x oder neuer
- Linux Container mit installiertem Git
- Grundkenntnisse in DNS, Active Directory, Mailservern und Proxmox VE

Übersicht der Migration

1. Zamba Active Directory als zweiten Domain Controller installieren
2. Active Directory replizieren und prüfen
3. DNS auf den neuen Domain Controller umstellen
4. Mailcow installieren
5. LDAP-Anbindung in Mailcow konfigurieren
6. Benutzer und Mailboxen synchronisieren
7. Mails von Kopano nach Mailcow migrieren
8. Kalender und Kontakte übernehmen
9. Aliase und Weiterleitungen prüfen
10. Mailcow produktiv schalten
11. FSMO-Rollen übernehmen
12. Univention UCS demoten und abschalten
13. LDAP-Clients auf den neuen Zamba AD umstellen

Zamba Active Directory als zweiten Domain Controller installieren

Die Installation erfolgt auf dem Proxmox Host, zum Beispiel im Verzeichnis `/root`.

```
git clone -b dev https://github.com/bashclub/zamba-lxc-toolbox
cd zamba-lxc-toolbox
cp conf/zamba.conf.example conf/zamba.conf
```

zamba.conf anpassen

In der Datei `conf/zamba.conf` werden die Werte an die bestehende Umgebung angepasst.

```
LXC_TEMPLATE_STORAGE="local"
LXC_ROOTFS_SIZE="32"
LXC_ROOTFS_STORAGE="rpool-data"
LXC_SHAREFS_SIZE="100"
LXC_SHAREFS_STORAGE="rpool-data"

LXC_HOSTNAME="zmb-ad"
LXC_DOMAIN="windomain.local"
LXC_IP="10.0.0.254/24"
LXC_GW="10.0.0.1"
LXC_DNS="10.0.0.4"

LXC_BRIDGE="vmbr0"
LXC_PWD='HIER_STARKES_PASSWORT_SETZEN'

ZMB_REALM="WINDOMAIN.LOCAL"
ZMB_DOMAIN="WINDOMAIN"
ZMB_ADMIN_USER="administrator"
ZMB_ADMIN_PASS='HIER_AD_ADMIN_PASSWORT_SETZEN'
```

Hinweis: `LXC_DNS` zeigt zunächst auf den alten UCS-DNS-Server. Bei Windows-Domänen sollte vorher gegebenenfalls das Funktionslevel geprüft werden.

Zamba AD Join starten

```
bash install.sh -i 100
```

Im Menü wird **Zamba-AD-Join** gewählt.

Replikation prüfen

Nach der Installation kann der Container betreten und die Replikation geprüft werden.

```
pct enter 100
su -
wbinfo -u
wbinfo -g
```

Die Ausgabe von Benutzern und Gruppen ist ein erster Funktionstest, ersetzt aber keine Prüfung von DRS-Replikation, DNS, SYSVOL und Anmeldungen.

> **Empfehlung:** In Proxmox sollte anschließend der alte UCS-DNS-Server aus der Hostkonfiguration entfernt werden, sobald der neue Zamba AD DNS zuverlässig funktioniert.

smb.conf anpassen

Folgende Anpassungen können in der `smb.conf` hilfreich sein:

```
vi /etc/samba/smb.conf
```

```
[global]
# DNS-Forwarder nur entsprechend der eigenen DNS-Architektur setzen:
dns forwarder = 1.1.1.1
```

DNS auf den neuen Domain Controller umstellen

DNS ist bei Active Directory fast immer der kritischste Punkt. Sobald der neue Zamba Domain Controller sauber repliziert, werden feste IP-Konfigurationen und DHCP-Einstellungen angepasst.

- Alle Server mit statischer IP erhalten den neuen Zamba AD als DNS-Server
- Im DHCP-Server wird der alte UCS-DNS durch den neuen DC ersetzt

- Alte UCS-DNS-Einträge werden später bereinigt
- RSAT DNS-Konsole zur Kontrolle verwenden

> **Merksatz:** It was DNS.

Mailcow installieren

Für Mailcow wird ein weiterer LXC verwendet. Die Installation erfolgt ebenfalls über die Zamba LXC Toolbox.

```
cp conf/zamba.conf.example conf/zamba.conf
```

Für Mailcow müssen insbesondere Hostname und Netzwerkeinstellungen angepasst werden.

```
bash install.sh -i 101
```

Im Menü wird **Mailcow** gewählt.

Mailcow aktualisieren

```
pct enter 101
su -
cd /opt/mailcow-dockerized
apt install jq
./update.sh
```

Danach wird das Mailcow-Webinterface geöffnet:

```
https://10.0.0.253/admin
```

LDAPS: Bei Port 636 ein gültiges, vertrauenswürdiges TLS-Zertifikat einsetzen. Die Option zur Ignorierung von Zertifikatsfehlern ist keine sichere Dauerlösung.

Grundkonfiguration von Mailcow

Unbedingt notwendige Schritte

- **Admin-Passwort ändern:** System / Konfiguration / Administrator bearbeiten
- **Alle Maildomains eintragen:** E-Mail / Konfiguration / Domains
- **Schwellwerte und Quotas beachten**
- **LDAP-Bind-User im AD anlegen:** zum Beispiel `bind-mailcow`

LDAP in Mailcow konfigurieren

Die LDAP-Anbindung erfolgt unter:

> System / Konfiguration / Zugang / Identity Provider

- Identity Provider: LDAP
- Host: IP des neuen Zamba AD
- Port: 636
- Benutze SSL: an
- SSL-Zertifikat prüfen: Zertifikatsfehler NICHT ignorieren; vertrauenswürdige CA einrichten und idealerweise DNS-Namen statt IP verwenden.
- Base DN: `CN=Users,DC=windomain,DC=local`
- Username Feld: `mail`
- Attribute Feld: `mail`
- Bind DN: `CN=bind-mailcow,CN=Users,DC=windomain,DC=local`
- Bind Passwort: Passwort des Bind-Users
- Attribute Mapping: Standardvorlage / Default
- Benutzer beim Login erstellen: an
- Vollsynchronisation: an
- Importiere Benutzer: an
- Sync / Import interval: für den Anfang 1 Minute

Der Erfolg wird unter **E-Mail / Mailboxen** kontrolliert.

Vor der Mailmigration: Vollständige Backups und Wiederherstellungstest durchführen. Zugangsdaten-Dateien mit `chmod 600` schützen. Passwörter in Prozessargumenten können sichtbar werden; Migrationshost und Prozesse entsprechend absichern. `--delete2`, `--delete2folders` und `--delete2duplicates` nicht ohne getestete Ziel-Sicherung aktivieren.

E-Mails von Kopano nach Mailcow migrieren

E-Mails lassen sich über IMAP nach mailcow übertragen. Bei Kopano sollten Dubletten, Ordnerzuordnung und Zeitstempel besonders sorgfältig geprüft werden. Ein Testlauf mit wenigen Postfächern ist vor der Massenmigration Pflicht.

Das folgende Bash-Skript verarbeitet eine Zugangsdaten-Datei mit vier Zeilen pro Postfach. Es synchronisiert zunächst ohne Löschoptionen. Achtung: --delete2 löscht auf dem ZIEL (Mailcow), nicht auf der Quelle; nur nach expliziter Prüfung und Sicherung verwenden.

```
#!/bin/bash
# ================================================================================
# Kopano -> Mailcow IMAP Migration (Sicherheits-optimierte Version)
# Synchronisiert alle E-Mails 1:1
#
# Dateiformat für mailboxes.csv (4 Zeilen pro Benutzer, Kommentarzeilen mit #
erlaubt):
#   user@source
#   password
#   user@target
#   password
# ================================================================================

set -e

usage() {
    cat <<EOF
Verwendung:
  $(basename "$0") <host1> <host2> <csv-datei>
  $(basename "$0") -h | --help
```

```
Argumente:
  host1       Hostname des Quell-IMAP-Servers (Kopano)
  host2       Hostname des Ziel-IMAP-Servers  (Mailcow)
  csv-datei   Pfad zur CSV-Datei mit Benutzern und Passwörtern

Dateiformat (4 Zeilen pro Benutzer):
  user@source        Benutzername auf dem Quell-Server (Kopano)
  password           Passwort auf dem Quell-Server
  user@target        Benutzername auf dem Ziel-Server  (Mailcow)
  password           Passwort auf dem Ziel-Server

  Zeilen die mit # beginnen werden als Kommentare ignoriert.
  Leerzeilen werden übersprungen.
  Passwörter dürfen beliebige Zeichen enthalten (auch Kommas, Anführungszeichen etc.).

Beispiel:
  $(basename "$0") imap.kopano.example imap.mailcow.example mailboxes.csv

  mailboxes.csv:
    # Alice Musterfrau
    alice@kopano.example
    P@ss,w0rd#1!
    alice@mailcow.example
    N3wS3cr3t&
    # Bob Mustermann
    bob@kopano.example
    geheim42
    bob@mailcow.example
    anderes$Passwort

Hinweise:
  - SSL wird für beide Server automatisch verwendet (Port 993).

EOF
}

# Argumente prüfen
if [[ "${1-}" == "-h" || "${1-}" == "--help" ]]; then
    usage
```

```
    exit 0
fi

if [ "$#" -ne 3 ]; then
    usage
    exit 1
fi

HOST1="$1"
HOST2="$2"
CSV_FILE="$3"

# CSV-Datei prüfen
if [ ! -f "$CSV_FILE" ]; then
    echo "FEHLER: CSV-Datei '$CSV_FILE' nicht gefunden."
    exit 1
fi
if [ ! -r "$CSV_FILE" ]; then
    echo "FEHLER: CSV-Datei '$CSV_FILE' ist nicht lesbar."
    exit 1
fi

# Prüfung ob das Tool vorhanden ist
if ! command -v kopano-migration-imap &> /dev/null; then
    echo "FEHLER: Tool 'kopano-migration-imap' nicht gefunden. Bitte installieren."
    exit 1
fi

# Benutzeranzahl ermitteln: nicht-leere, nicht-kommentierte Zeilen / 4
DATA_LINES=$(grep -c -E '^[^#[:space:]]' "$CSV_FILE" || true)
USER_COUNT=$(( DATA_LINES / 4 ))

echo "========================================"
echo "Kopano -> Mailcow Migration"
echo "========================================"
echo "Host1      : $HOST1 (Quelle - Kopano)"
echo "Host2      : $HOST2 (Ziel - Mailcow)"
echo "CSV-Datei  : $CSV_FILE"
echo "Benutzer   : $USER_COUNT"
```

```
echo "========================================="

# WARNUNG AUSGEBEN
echo "WARNING: Es werden SSL-Verbindungen zu beiden Servern aufgebaut."
echo "WARNING: Option '--delete2' löscht Nachrichten auf dem ZIELSERVER (Mailcow),
nicht auf dem Quellserver."
read -p "Möchten Sie fortfahren? [j/N] " confirm
if [[ ! "$confirm" =~ ^[Jj] ]]; then
    echo "Abgebrochen."
    exit 1
fi

# Zähler
SUCCESS_COUNT=0
FAIL_COUNT=0
LINE_NUM=0
FAILED_USERS=()

# Datei einlesen: 4 aufeinanderfolgende Nicht-Kommentar/Nicht-Leer-Zeilen = 1 Benutzer
FIELD=0
USER1="" PASS1="" USER2="" PASS2="" ENTRY_LINE=0

process_user() {
    echo
    echo "========================================="
    echo "Migriere: $USER1 -> $USER2"
    echo "========================================="

    IMAP_ARGS=(
        --host1 "$HOST1"
        --host2 "$HOST2"
        --user1 "$USER1"
        --password1 "$PASS1"
        --user2 "$USER2"
        --password2 "$PASS2"
        --ssl1
        --ssl2
        --automap
        --useheader Message-Id
```

```
        --addheader
        --usecache
        --syncinternaldates
        --skipsize
        --nofoldersizes
        --nofoldersizesatend
     #   --delete2
     #   --delete2duplicates
     #   --exclude '(?i)trash|papierkorb|gelöschte elemente|deleted items|deleted
messages|junk|spam'
     #   --maxage 1825
    )

    if kopano-migration-imap "${IMAP_ARGS[@]}"; then
        echo "SUCCESS: $USER1 -> $USER2"
        SUCCESS_COUNT=$((SUCCESS_COUNT + 1))
    else
        echo "FEHLER: Migration von $USER1 -> $USER2 fehlgeschlagen."
        FAILED_USERS+=("$USER1 -> $USER2")
        FAIL_COUNT=$((FAIL_COUNT + 1))
    fi
}

while IFS= read -r line || [ -n "$line" ]; do
    LINE_NUM=$((LINE_NUM + 1))

    # Kommentare und leere Zeilen überspringen
    [[ "$line" =~ ^[[:space:]]*# ]] && continue
    [[ -z "${line// }" ]] && continue

    FIELD=$((FIELD + 1))
    case $FIELD in
        1) USER1="$line"; ENTRY_LINE=$LINE_NUM ;;
        2) PASS1="$line" ;;
        3) USER2="$line" ;;
        4) PASS2="$line"
           FIELD=0
           process_user
           ;;
```

```
    esac

done < "$CSV_FILE"

# Warnung bei unvollständigem letzten Eintrag
if [ "$FIELD" -ne 0 ]; then
    echo "WARNUNG: Unvollständiger Eintrag ab Zeile $ENTRY_LINE ($FIELD von 4 Feldern)
— übersprungen."
    FAILED_USERS+=("Zeile $ENTRY_LINE (unvollständig, $FIELD/4 Felder)")
    FAIL_COUNT=$((FAIL_COUNT + 1))
fi

# Zusammenfassung
echo
echo "========================================="
echo "Zusammenfassung"
echo "========================================="
echo "Erfolgreich   : $SUCCESS_COUNT"
echo "Fehlgeschlagen: $FAIL_COUNT"
if [ "${#FAILED_USERS[@]}" -gt 0 ]; then
    echo
    echo "Fehlgeschlagene Benutzer:"
    for u in "${FAILED_USERS[@]}"; do
        echo "  - $u"
    done
fi
echo "========================================="

if [ "$FAIL_COUNT" -gt 0 ]; then
    exit 1
fi
```

Beispiel der mailboxes.csv

```
# ================================================================================
# Beispiel-Datei für mailboxes.csv
# Format: 4 Zeilen pro Benutzer (Kommentare und Leerzeilen werden ignoriert)
#   Zeile 1: Benutzername Quell-Server (Kopano)
#   Zeile 2: Passwort    Quell-Server
```

```
#   Zeile 3: Benutzername Ziel-Server  (Mailcow)
#   Zeile 4: Passwort    Ziel-Server
# ================================================================================

# Alice Musterfrau
alice@kopano.example
P@ss
alice@mailcow.example
N3w$3cr3t&more

# Bob Mustermann (Passwort enthält Anführungszeichen und Sonderzeichen)
bob@kopano.example
sehrkompliziert!42
bob@mailcow.example
B0b#N3u=P@ss!

# Carol Schmidt (Passwort mit Leerzeichen und Sonderzeichen)
carol@kopano.example
mein passwort mit leerzeichen & symbolen!
carol@mailcow.example
C@r0l2024#sicher
```

Kalender aus Kopano exportieren

Kopano stellt Kalender über CalDAV bereit. Beispiele:

`http://kopano:8080/caldav/<user>/<folder-name>/`	Kalender oder Aufgabenordner im Benutzerpostfach
`http://kopano:8080/caldav/<user>/<sub-folder>/`	Selbst erstellter Unterkalender
`http://kopano:8080/caldav/<other-user>/<folder-name>/`	Freigegebener Kalender eines anderen Benutzers
`http://kopano:8080/caldav/public/<foldername>/`	Öffentlicher Kalender

`http://kopano:8080/caldav/<user>/`	Standardkalender des Benutzers

Die exportierten Dateien können anschließend im SOGo-Frontend oder in einem Mailclient importiert werden.

Kontakte exportieren und importieren

Kontakte exportieren

Der Export kann über den alten Mailclient erfolgen. Alternativ kann ein Python-Script mit Kopano-MAPI genutzt werden.

```
sudo apt install python3-kopano
mkdir export
wget https://raw.githubusercontent.com/bashclub/trmm-scripts/refs/heads/main/kopano-export-contacts.py
python3 kopano-export-contacts.py
```

Das Script exportiert Kontakte in Unterordner pro Benutzer, da Kopano kein echtes CardDAV für diesen Zweck bereitstellt.

Kontakte importieren

- Outlook kann beim Export oder Import Probleme verursachen
- SOGo importiert Kontakte oft nur einzeln
- eM Client war in der Praxis häufig die beste Lösung für den Import vieler VCF-Dateien per Drag & Drop

Beispiel für CardDAV-Anbindung:

```
https://kunde.dyndns.org/SOGo/dav/user@domain.de/Contacts/personal/
```

Aliase und Weiterleitungen übernehmen

Aliase finden

```
univention-ldapsearch -LLL | grep @ | grep mailAlternative
```

Aliase werden in Mailcow unter **E-Mail / Aliasse / Alias hinzufügen** eingetragen.

Weiterleitungen finden

```
univention-ldapsearch -LLL | grep mailForward
```

> Weiterleitungen je nach Art als mailcow-Adressumschreibung/Alias oder als benutzerbezogene Sieve-Regel in SOGo anlegen. Die bisherigen UCS-Regeln vorab vollständig inventarisieren.

Mailcow produktiv schalten

NAT und Portweiterleitungen

Auf der Firewall, zum Beispiel OPNsense, können Aliase verwendet werden.

```
p_mailcow: 80 443 465 993 4190
h_mailcow: 10.0.0.253
```

Port Forwarding:

```
any > WAN IP > p_mailcow > h_mailcow
```

Mailcow neu starten

```
pct enter 101
cd /opt/mailcow-dockerized

# Erst prüfen, ob eine bestehende Weiterleitung angepasst werden muss:
ls -l data/conf/nginx/redirect.conf

# Konfiguration prüfen und anschließend Dienste starten:
docker compose config --quiet
docker compose up -d
```

Optional: Greylisting deaktivieren

```
vi /opt/mailcow-dockerized/data/conf/rspamd/local.d/greylist.conf
```

```
enabled = false;
```

Optional: Let’s Encrypt Einstellungen

```
vi /opt/mailcow-dockerized/mailcow.conf
```

```
SKIP_LETS_ENCRYPT=n
SKIP_HTTP_VERIFICATION=y
```

```
docker compose up -d
```

Let’s Encrypt kontrollieren

```
docker compose logs --tail=200 -f acme-mailcow
```

Alternativ eigenes Zertifikat verwenden

Eigene Zertifikate können hier abgelegt werden:

```
/opt/mailcow-dockerized/data/assets/ssl
```

- `cert.pem`
- `key.pem`

Danach DNS-Werte beim Provider kontrollieren:

> E-Mail / Domains / DNS

Reihenfolge beim DC-Wechsel: Den alten UCS nicht vor der FSMO-Übertragung und dem Nachweis funktionierender DNS-, SYSVOL- und AD-Replikation abschalten. Einen erreichbaren DC regulär demoten; „remove-other-dead-server“ ist nur für einen tatsächlich nicht mehr erreichbaren DC vorgesehen.

Abschluss der Migration

- Synchronisationen nach erfolgreicher Migration deaktivieren
- Optional Proxmox Mail Gateway auf Mailcow verweisen
- Alten Univention Corporate Server abschalten
- FSMO-Rollen auf Zamba AD übertragen
- Alte UCS-Fragmente bereinigen
- LDAP-Clients auf neuen Zamba AD umstellen

FSMO-Rollen übernehmen

```
pct enter 100
su -

# Vorher Replikation/DNS prüfen, dann FSMO übertragen:
samba-tool fsmo transfer --role=all -Uadministrator
samba-tool fsmo show

# UCS anschließend kontrolliert herunterstufen, solange erreichbar.
# Nur bei dauerhaft ausgefallenem DC nach eigener Prüfung:
# samba-tool domain demote --remove-other-dead-server=UCS

# Cronjob erst nach validierter SYSVOL-Umstellung entfernen:
# rm /etc/cron.d/sysvol-sync
```

Bereinigung

- RSAT DNS-Konsole nach alten Fragmenten durchsuchen
- Alte DNS-Einträge entfernen
- Optional Metadata Cleanup durchführen

LDAP-Clients anpassen

Für bestehende LDAP-Clients werden neue Bind-User pro Rolle angelegt. Anschließend werden die LDAP-Parameter angepasst.

- IP: neuer Zamba AD
- LDAP-Port pro Anwendung prüfen: häufig 636/LDAPS; UCS-spezifische Altports nicht ungeprüft übernehmen.
- UID auf `sAMAccountName` ändern
- Bind-DN anhand der realen AD-Objekte und des LDAP-Baums ermitteln; nicht pauschal uid durch cn ersetzen.

Proxmox Mail Gateway mit Mailcow verbinden

Greylisting in Mailcow deaktivieren

```
vi /opt/mailcow-dockerized/data/conf/rspamd/local.d/greylist.conf
```

```
enabled = false;
```

```
docker compose restart rspamd-mailcow
```

Routing in Mailcow

- Konfiguration / Routing / Networks
- `ipvompmg:26`
- Greylisting deaktivieren

Senderabhängige Transport Maps

- Für alle Maildomains, die PMG nutzen
- ID 1: `ipvompmg:26`

Proxmox Mail Gateway

- Configuration / Mail Proxy
- Relay Domain setzen
- Transport zur Mailcow konfigurieren
- Kein MX für internen Transport verwenden

Beim DNS-Provider werden die PMG-Werte statt der direkten Mailcow-Werte gesetzt.

Mailpiler als Mailarchiv integrieren

Mailpiler kann ebenfalls über die LXC Toolbox installiert werden.

Mailcow für Archivweiterleitung vorbereiten

- System / Konfiguration / Einstellungen / Weiterleitungs-Hosts
- IP des Mailpilers eintragen

BCC an das Archiv

- System / Konfiguration / Routing / Transport
- Ziel: `piler.windomain.local`
- Next Hop: `10.0.0.x`

BCC-Maps einrichten

Unter:

> E-Mail / Konfiguration / Adressumschreibung / BCC-Maps

Für jede Maildomain jeweils eingehend und ausgehend:

- Lokales Ziel: `sysops.tv`
- BCC-Ziel: `piler@piler.windomain.local`
- Domain: `sysops.tv`

Kontrolle im Mailpiler:

```
tail -f /var/log/mail.log
```

> LDAP-Zuordnung und Alias-Auflösung in Mailpiler nach der Umstellung mit echten Benutzerkonten überprüfen.

Doppelte Mails bei mehrfacher Kopano-Synchronisation vermeiden

Die IMAP-Kompatibilität von Kopano ist in der Praxis oft problematisch. Wenn Mails doppelt oder dreifach erscheinen, kann eine gezielte Bereinigung und erneute Synchronisation helfen.

```
apt install kopano-migration-imap
nano migrate.sh
```

```
#!/bin/bash
# Sicherer Diagnoseablauf statt pauschaler Ordnerlöschung
set -euo pipefail
cd /opt/mailcow-dockerized
USER_MAIL="${1:?Aufruf: $0 user@example.org}"

echo "Ordnerübersicht von $USER_MAIL:"
docker compose exec -T dovecot-mailcow doveadm mailbox list -u "$USER_MAIL"

echo "Anzahl E-Mails pro Ordner prüfen; keine Löschung erfolgt."
echo "Duplikate nur nach Backup und dokumentierter Einzelfallprüfung bereinigen."
```

Für die Dublettenanalyse die oben gezeigte nicht-destruktive Diagnose verwenden:

```
bash diagnose-mailcow-folders.sh benutzer@example.org
```

> Das Beispiel nimmt keine Papierkörbe oder Spam-Ordner mit. Dies wird über `--exclude` gesteuert.

Optionale DNS-Einträge für Autokonfiguration

Für Thunderbird, eM Client und weitere Programme können SRV-Records gesetzt werden, damit Mail, Kalender und Kontakte automatisch gefunden werden.

```
_autodiscover._tcp  IN  SRV  0 0 443 q.deine-domain.de.
_caldavs._tcp       IN  SRV  0 1 443 q.deine-domain.de.
_carddavs._tcp      IN  SRV  0 1 443 q.deine-domain.de.
_imaps._tcp         IN  SRV  0 1 993 q.deine-domain.de.
_sieve._tcp         IN  SRV  0 1 4190 mail.deine-domain.de.
_smtps._tcp         IN  SRV  0 1 465 q.deine-domain.de.
_submissions._tcp   IN  SRV  0 1 465 q.deine-domain.de.
```

Fazit

Für überschaubare AD- und Groupware-Umgebungen kann eine Migration von UCS und Kopano auf Zamba AD, mailcow/SOGo und Proxmox VE sinnvoll sein. Zuvor sind genutzte UCS-Apps, Freigaben, Berechtigungen, Kalenderfunktionen, Clients und Supportanforderungen individuell abzugleichen.

- Active Directory bleibt erhalten
- Kopano wird durch Mailcow ersetzt
- Updates werden deutlich schneller
- Proxmox VE ermöglicht einfache Snapshots und Backups
- Mailpiler und Proxmox Mail Gateway können integriert werden
- Der Betrieb wird transparenter und besser kontrollierbar

Passende Schulungen

Wenn Sie die Migration unter Anleitung durchführen möchten, finden Sie passende Live-Schulungen auf **cloudistboese.de**:

- Windows Domain Controller mit Samba/Zamba ablösen

- Open-Source Mailserver mit Mailcow betreiben
- Alles ohne Windows
- Proxmox produktiv mit ZFS betreiben
- Replikationen und Backups trojanersicher

Häufige Fragen zur UCS- und Kopano-Migration

Kann man Univention UCS durch Samba/Zamba AD ersetzen?

Ja, sofern die Umgebung primär Active Directory bereitstellt und alle zusätzlich genutzten UCS-Funktionen vorab berücksichtigt werden. DNS, SYSVOL, Replikation und FSMO-Rollen müssen beim Wechsel geprüft werden.

Wie werden E-Mails von Kopano nach mailcow migriert?

Typischerweise über IMAP-Synchronisation, ergänzt um getrennte Exporte für Kalender und Kontakte. Bei Kopano können Ordnerzuordnungen und Duplikate zusätzliche Tests erfordern.

Ist mailcow ein vollständiger Ersatz für Kopano?

mailcow stellt den Mailserver bereit und integriert SOGo als Webmail- und Groupware-Oberfläche. Unterschiede bei Freigaben, Delegation, Mobilgeräten und Outlook-Anbindung sollten vorab getestet werden.

Optional: Samba Active Directory nach der UCS-Migration prüfen

Nach der Übernahme der FSMO-Rollen und der Abschaltung des alten Univention Domain Controllers kann ein **Samba AD DC Health Check** helfen, verbliebene UCS-Referenzen und Probleme in der Active-Directory-Konfiguration zu erkennen. Das folgende Prüfsystem betrachtet Computer- und Serverobjekte, FSMO-Rollen, DRS-Replikation, NTDS-Settings, DNS- und SRV-Einträge sowie die Datenbankkonsistenz.

> **Wichtig:** Das Skript ist für die **Nachkontrolle einer abgeschlossenen Migration** ausgelegt und verändert weder Active Directory noch DNS. Es erwartet in der gezeigten Fassung einen einzigen verbleibenden DC. Warnungen zu alten Objekten sollten fachlich geprüft werden; alte Einträge nicht ungeprüft löschen. Auch ein fehlerfreier Bericht ersetzt kein Backup und keinen Funktionstest mit Domänenclients.

Konfiguration und Ausführung

Vor der Ausführung die Variablen `DOMAIN`, `WORKGROUP`, `CURRENT_DC`, `OLD_DC`, `OLD_GUID` und die abgeleiteten Distinguished Names an die eigene Domäne anpassen. Im Beispiel sind dies `lermann.lan`, `ZMB` und `UCS`. Das Skript auf dem neuen Samba-/Zamba-Domain-Controller als `root` ausführen. Benötigt werden unter anderem `samba-tool`, `ldbsearch` und `dig`.

```
nano /root/samba-ad-check.sh
chmod 700 /root/samba-ad-check.sh
/root/samba-ad-check.sh
```

Das Skript fragt das Administratorpasswort interaktiv ab und liefert am Ende eine Zusammenfassung aus erfolgreichen Prüfungen, Warnungen und Fehlern (Exitcodes 0, 1 bzw. 2). Die Werte und Prüfannahmen sind umgebungsspezifisch; insbesondere sind die Ausgaben zu DNS-NS-Einträgen und zu Replikationspartnern bei mehreren DCs gesondert zu bewerten.

Vollständiges Prüfsystem: samba-ad-check.sh

```
#!/usr/bin/env bash
#
# Samba AD DC Health Check
# Version 2.0
#
# Read-only: keine Aenderungen an AD oder DNS.
#
# Ausfuehrung:
#   chmod 700 /root/samba-ad-check.sh
#   /root/samba-ad-check.sh
#

set -uo pipefail
umask 077
export LC_ALL=C

# ============================================================
# KONFIGURATION
# ============================================================

DOMAIN="lermann.lan"
WORKGROUP="LERMANN"
CURRENT_DC="ZMB"
OLD_DC="UCS"

OLD_GUID="8be79f79-1f42-4775-accd-cde27e5eaf11"

ADMIN_USER="Administrator"
DNS_SERVER="127.0.0.1"
SAMDB="/var/lib/samba/private/sam.ldb"

BASE_DN="DC=lermann,DC=lan"
CONFIG_DN="CN=Configuration,$BASE_DN"
```

```
SITE_DN="CN=Sites,$CONFIG_DN"

# ============================================================
# FARBEN UND ZAEHLER
# ============================================================

if [[ -t 1 && -z "${NO_COLOR:-}" ]]; then
    GREEN=$'\033[32m'
    YELLOW=$'\033[33m'
    RED=$'\033[31m'
    BLUE=$'\033[36m'
    RESET=$'\033[0m'
else
    GREEN=""
    YELLOW=""
    RED=""
    BLUE=""
    RESET=""
fi

OK=0
WARN=0
ERR=0

section() {
    printf '\n%s========== %s ==========%s\n' \
        "$BLUE" "$1" "$RESET"
}

ok() {
    printf '%s[OK]%s %s\n' "$GREEN" "$RESET" "$1"
    ((OK+=1))
}

warn() {
    printf '%s[WARNUNG]%s %s\n' \
        "$YELLOW" "$RESET" "$1"
    ((WARN+=1))
}
```

```
error() {
    printf '%s[FEHLER]%s %s\n' \
        "$RED" "$RESET" "$1"
    ((ERR+=1))
}

# ============================================================
# VORAUSSETZUNGEN
# ============================================================

if [[ "$EUID" -ne 0 ]]; then
    echo "Bitte als root ausfuehren."
    exit 1
fi

for cmd in samba-tool ldbsearch dig grep awk mktemp; do
    if ! command -v "$cmd" >/dev/null 2>&1; then
        echo "[FEHLER] Fehlender Befehl: $cmd"
        exit 1
    fi
done

if [[ ! -r "$SAMDB" ]]; then
    echo "[FEHLER] Samba-Datenbank nicht lesbar: $SAMDB"
    exit 1
fi

# ============================================================
# PASSWORT EINMAL ABFRAGEN
# ============================================================

echo "================================================"
echo " SAMBA AD DOMAIN CONTROLLER HEALTH CHECK"
echo "================================================"
echo "Domain       : $DOMAIN"
echo "Aktiver DC   : $CURRENT_DC"
echo "Alter DC     : $OLD_DC"
echo "DNS-Server   : $DNS_SERVER"
```

```
echo "Datum        : $(date)"
echo

read -r -s -p \
    "Passwort fuer ${WORKGROUP}\\${ADMIN_USER}: " \
    PASSWORD
echo

if [[ -z "$PASSWORD" ]]; then
    echo "[FEHLER] Kein Passwort eingegeben."
    exit 1
fi

# Passwort ueber Dateideskriptor 3 an Samba uebergeben.
# Kein Passwort in Prozessargumenten oder temporaeren Dateien.

samba_dns() {
    PASSWD_FD=3 samba-tool dns "$@" \
        -U "${WORKGROUP}\\${ADMIN_USER}" \
        3<<<"$PASSWORD"
}

cleanup() {
    unset PASSWORD
}
trap cleanup EXIT

# ============================================================
# HILFSFUNKTIONEN
# ============================================================

check_old_reference() {
    local DATA="$1"
    local NAME="$2"

    if grep -Eiq \
        "(^|[^[:alnum:]_-])${OLD_DC}([.]|[^[:alnum:]_-]|$)" \
        <<< "$DATA"; then
        warn "$NAME enthaelt Referenzen auf $OLD_DC"
```

```
    else
        ok "$NAME ohne Referenzen auf $OLD_DC"
    fi
}

dns_lookup() {
    local NAME="$1"
    local TYPE="$2"
    local REQUIRE_DC="$3"
    local RESULT

    echo
    echo "--- $NAME ($TYPE) ---"

    if ! RESULT=$(dig @"$DNS_SERVER" \
        "$NAME" "$TYPE" \
        +short +time=3 +tries=2); then

        error "DNS-Abfrage fehlgeschlagen: $NAME"
        return
    fi

    if [[ -z "$RESULT" ]]; then
        warn "Kein DNS-Eintrag: $NAME"
        return
    fi

    echo "$RESULT"

    check_old_reference "$RESULT" "DNS $NAME"

    if [[ "$REQUIRE_DC" == "yes" ]]; then

        if grep -Fiq \
            "${CURRENT_DC}.${DOMAIN}." <<< "$RESULT"; then
            ok "DNS verweist auf $CURRENT_DC"
        else
            error "DNS verweist nicht auf $CURRENT_DC"
        fi
```

```
    else
        ok "DNS-Eintrag vorhanden"
    fi
}

# ============================================================
# 1. COMPUTEROBJEKTE
# ============================================================

section "COMPUTEROBJEKTE"

if COMPUTERS=$(samba-tool computer list 2>&1); then

    if grep -Fxiq "${CURRENT_DC}\$" <<< "$COMPUTERS"; then
        ok "Computerobjekt $CURRENT_DC vorhanden"
    else
        error "Computerobjekt $CURRENT_DC fehlt"
    fi

    if grep -Fxiq "${OLD_DC}\$" <<< "$COMPUTERS"; then
        warn "Altes Computerobjekt $OLD_DC vorhanden"
    else
        ok "Altes Computerobjekt $OLD_DC entfernt"
    fi

else
    error "Computerliste nicht abrufbar"
    echo "$COMPUTERS"
fi

# ============================================================
# 2. FSMO-ROLLEN
# ============================================================

section "FSMO-ROLLEN"

if FSMO=$(samba-tool fsmo show 2>&1); then

    echo "$FSMO"
```

```
    OWNERS=$(grep -i 'Role owner:' <<< "$FSMO" || true)
    OWNER_COUNT=$(grep -ic 'Role owner:' <<< "$FSMO" || true)

    if [[ "$OWNER_COUNT" -lt 5 ]]; then
        error "Zu wenige FSMO-Rollen erkannt: $OWNER_COUNT"

    elif grep -vi "CN=${CURRENT_DC}," \
        <<< "$OWNERS" >/dev/null; then

        error "Mindestens eine FSMO-Rolle liegt nicht auf $CURRENT_DC"

    else
        ok "Alle $OWNER_COUNT angezeigten Rollen liegen auf $CURRENT_DC"
    fi

else
    error "FSMO-Abfrage fehlgeschlagen"
    echo "$FSMO"
fi

# ============================================================
# 3. REPLIKATION
# ============================================================

section "DRS REPLIKATION"

if REPL=$(samba-tool drs showrepl 2>&1); then

    echo "$REPL"

    if grep -Eq 'NTDS DN:|DSA object GUID:.*' \
        <<< "$REPL"; then

        NEIGHBORS=$(grep -c 'NTDS DN:' <<< "$REPL" || true)

        if [[ "$NEIGHBORS" -eq 0 ]]; then
            ok "Keine Replikationspartner (Single-DC)"
        else
```

```
            warn "$NEIGHBORS Replikationspartner vorhanden"
        fi
    fi

    if grep -Eiq \
        'failed, result|consecutive failure' <<< "$REPL"; then
        error "Replikationsfehler vorhanden"
    else
        ok "Keine Replikationsfehler"
    fi

    check_old_reference "$REPL" "DRS"

else
    error "DRS-Abfrage fehlgeschlagen"
    echo "$REPL"
fi

# ============================================================
# 4. SERVEROBJEKTE
# ============================================================

section "AD SERVEROBJEKTE"

if SERVERS=$(ldbsearch -H "$SAMDB" \
    -b "$SITE_DN" \
    '(objectClass=server)' \
    dn cn dNSHostName 2>&1); then

    echo "$SERVERS"

    if grep -Fq \
        "CN=${CURRENT_DC},CN=Servers," <<< "$SERVERS"; then
        ok "Serverobjekt $CURRENT_DC vorhanden"
    else
        error "Serverobjekt $CURRENT_DC fehlt"
    fi

    check_old_reference "$SERVERS" "Serverobjekte"
```

```
else
    error "Serverobjekte nicht lesbar"
    echo "$SERVERS"
fi

# ============================================================
# 5. NTDS SETTINGS
# ============================================================

section "NTDS SETTINGS"

if NTDS=$(ldbsearch -H "$SAMDB" \
    -b "$SITE_DN" \
    '(objectClass=nTDSDSA)' \
    dn objectGUID 2>&1); then

    echo "$NTDS"

    if grep -Fq \
        "CN=NTDS Settings,CN=${CURRENT_DC}," \
        <<< "$NTDS"; then
        ok "NTDS Settings von $CURRENT_DC vorhanden"
    else
        error "NTDS Settings von $CURRENT_DC fehlen"
    fi

    check_old_reference "$NTDS" "NTDS Settings"

else
    error "NTDS Settings nicht lesbar"
    echo "$NTDS"
fi

# ============================================================
# 6. DNS RPC AUTHENTIFIZIERUNG
# ============================================================

section "DNS RPC AUTHENTIFIZIERUNG"
```

```
DNS_RPC_OK=0

if ZONES=$(samba_dns zonelist "$DNS_SERVER" 2>&1); then

    DNS_RPC_OK=1
    ok "DNS RPC Anmeldung erfolgreich"
    echo "$ZONES"

    if grep -Fq "pszZoneName" <<< "$ZONES" &&
       grep -Fq ": $DOMAIN" <<< "$ZONES"; then
        ok "DNS-Zone $DOMAIN vorhanden"
    else
        error "DNS-Zone $DOMAIN fehlt"
    fi

    if grep -Fq ": _msdcs.$DOMAIN" <<< "$ZONES"; then
        ok "DNS-Zone _msdcs.$DOMAIN vorhanden"
    else
        error "DNS-Zone _msdcs.$DOMAIN fehlt"
    fi

else
    error "DNS RPC Anmeldung fehlgeschlagen"
    echo "$ZONES"
fi

# ============================================================
# 7. DNS NS / LDAP / KERBEROS / GC
# ============================================================

section "DNS SRV UND NS"

dns_lookup "$DOMAIN" NS yes

dns_lookup \
    "_ldap._tcp.dc._msdcs.$DOMAIN" \
    SRV yes
```

```
dns_lookup \
    "_kerberos._tcp.$DOMAIN" \
    SRV yes

dns_lookup \
    "_gc._tcp.$DOMAIN" \
    SRV yes

# ============================================================
# 8. DNS ALTER UCS
# ============================================================

section "ALTER DNS HOST $OLD_DC"

if [[ "$DNS_RPC_OK" -eq 1 ]]; then

    if OLD_DNS=$(samba_dns query "$DNS_SERVER" \
        "$DOMAIN" "$OLD_DC" ALL 2>&1); then

        echo "$OLD_DNS"

        if grep -Eiq \
            'A:|AAAA:|CNAME:|NS:|SRV:' \
            <<< "$OLD_DNS"; then

            warn "Alter DNS-Eintrag $OLD_DC vorhanden"
        else
            ok "Kein aktiver DNS-Record fuer $OLD_DC erkannt"
        fi

    else
        if grep -Eiq \
            'NAME_DOES_NOT_EXIST|NAME_ERROR|WERR_DNS_ERROR_NAME_DOES_NOT_EXIST' \
            <<< "$OLD_DNS"; then

            ok "Alter DNS-Host $OLD_DC nicht vorhanden"

        else
            warn "DNS-Abfrage fuer $OLD_DC nicht eindeutig"
```

```
            echo "$OLD_DNS"
        fi
    fi

else
    warn "UCS-DNS-RPC-Pruefung wegen Authentifizierungsfehler uebersprungen"
fi

# ============================================================
# 9. ALTE DSA GUID IM DNS
# ============================================================

section "ALTE DSA GUID"

GUID_NAME="${OLD_GUID}._msdcs.${DOMAIN}"

if GUID_RESULT=$(dig @"$DNS_SERVER" \
    "$GUID_NAME" CNAME \
    +short +time=3 +tries=2); then

    if [[ -z "$GUID_RESULT" ]]; then
        ok "Alter DSA-GUID-CNAME nicht vorhanden"
    else
        warn "Alter DSA-GUID-CNAME vorhanden"
        echo "$GUID_RESULT"
    fi
else
    error "DSA-GUID-DNS-Abfrage fehlgeschlagen"
fi

# ============================================================
# 10. AD DATENBANK
# ============================================================

section "AD DATENBANKKONSISTENZ"

DBCHECK=$(samba-tool dbcheck --cross-ncs 2>&1)
DB_RC=$?
```

```
echo "$DBCHECK"

DB_ERRORS=$(grep -Eo \
    'Checked [0-9]+ objects \([0-9]+ errors\)' \
    <<< "$DBCHECK" | tail -n 1 || true)

if [[ -n "$DB_ERRORS" ]]; then

    ERROR_COUNT=$(sed -n \
        's/.*(\([0-9][0-9]*\) errors).*/\1/p' \
        <<< "$DB_ERRORS")

    if [[ "$ERROR_COUNT" -eq 0 ]]; then
        ok "AD-Datenbank ohne Fehler"
    else
        error "AD-Datenbank: $ERROR_COUNT Fehler"
    fi

elif [[ "$DB_RC" -ne 0 ]]; then
    error "AD-Datenbankpruefung fehlgeschlagen"
else
    warn "DB-Check-Ergebnis nicht eindeutig"
fi

# ============================================================
# 11. UCS-REFERENZEN IM DBCHECK
# ============================================================

section "UCS-ALTREFERENZEN IN DER DATENBANK"

if grep -Fq "$OLD_GUID" <<< "$DBCHECK"; then
    warn "Alte UCS-GUID wird im DB-Check noch referenziert"
else
    ok "Keine alte UCS-GUID im DB-Check gefunden"
fi

if grep -q 'msDS-NC-Replica-Locations' <<< "$DBCHECK"; then
    warn "Verwaiste msDS-NC-Replica-Locations gefunden"
else
```

```
    ok "Keine verwaisten Replica-Locations gemeldet"
fi

# ============================================================
# 12. ZUSAMMENFASSUNG
# ============================================================

section "GESAMTBERICHT"

printf '%-20s %s\n' "Domain:" "$DOMAIN"
printf '%-20s %s\n' "Aktiver DC:" "$CURRENT_DC"
printf '%-20s %s\n' "Alter DC:" "$OLD_DC"
printf '%-20s %s\n' "Datum:" "$(date)"

echo
printf '%sOK:           %d%s\n' "$GREEN" "$OK" "$RESET"
printf '%sWARNUNGEN:    %d%s\n' "$YELLOW" "$WARN" "$RESET"
printf '%sFEHLER:       %d%s\n' "$RED" "$ERR" "$RESET"
echo

if (( ERR > 0 )); then
    echo "ERGEBNIS: FEHLER - Administrator muss pruefen"
    exit 2
elif (( WARN > 0 )); then
    echo "ERGEBNIS: WARNUNGEN - Nachkontrolle erforderlich"
    exit 1
else
    echo "ERGEBNIS: ALLE PRUEFUNGEN ERFOLGREICH"
    exit 0
fi
```

Weiterbildung und Praxisworkshops: **cloudistboese.de**

Version #22
Erstellt: 2025-06-23 13:04:26 UTC von Christian Zengel (sysops GmbH)
Zuletzt aktualisiert: 2026-11-13 13:06:23 UTC von Christian Zengel (sysops GmbH)