AD/Kerberosanbindung KDE-NEON/Ubuntuclients inkl. Loginvertrauen

Getestet mit UCS 4.4

Der Join selbst

Bevor man beginnt muss man eine bestehende LDAP-Anbindung deaktivieren und alle Pakte mit PURGE deinstallieren und den Univention Domain-Join Client installieren.

```
apt remove nscd nslcd ldap-auth-client ldap-auth-config --purge && apt autormove --purge
```

Bei /etc/nssswitch.conf die LDAP Einträge entfernen.

ACHTUNG: Manuelle Einträge in den Pamconfigs werden beim Domainjoin überschrieben. Deshalb Einträge danach setzen.

```
add-apt-repository ppa:univention-dev/ppa
DEBIAN_FRONTEND=noninteractive apt-get install univention-domain-join
```

Der nächste Part hier scheint mittlerweile behoben zu sein. Die Resolvconf kann bleiben wie sie ist. Ich lass das hier mal zur Info noch so stehen. DNSMASQ darf keiner laufen.

Der Join mit Ubuntu/KDE-NEON funktioniert nicht out of the Box. Die Resolvconf muss mit einer anderen Funktion ihrer selbst verlinkt werden. Natürlich kann man beim Join auch die MasterIP angeben. Dann funktioniert der Join. Nur geht dann meist der Login nicht, und ein SSO am Ubuntu ist nicht möglich. Auch darf kein DNSmasq laufen.

rm /etc/resolv.conf
ln -s /run/systemd/resolve/resolv.conf /etc/.

Hier den DNSmasq Eintrag auskommentieren:

```
nano /etc/NetworkManager/NetworkManager.conf
```

Networkmanager neu starten. Ob man richtig am DNS hängt kann man ganz einfach mit folgenden Commando testen. Dies funktioniert aber nur wenn man die resolv.conf so wie oben beschrieben umlinked. Wie auch schon oben erwähnt ist dies kein Muss an Desktopsystemen.

```
host -la $(dnsdomainname)
```

oder eben direkt:

```
host -la meinedomain.wtf
```

Weiters muss noch in der Datei `/etc/lsb-release` die Zeile `DISTRIB_ID=neon` zu `DISTRIB_ID=Ubuntu` umgeschrieben werden. Verwendet man IPV6 ist in der Datei `/etc/sysctl.d/10-ipv6-privacy.conf` alles auf `0` zu stellen. Danach geht auch der Join normal.

Während dem Joinprozess wird das Netzwerk getrennt und die Config des Networkmanager mit statischen Daten überschrieben. Meist möchte man das aber auf der Automatik belassen. Daher am besten danach dem Join den Networkmanager wieder auf Automatik stellen und gesetzte DNS-Einträge löschen.

Um nun zu Joinen bedient mans sich der grafischen Oberläche, oder wenn man alle Variablen ausschließen möchte, nutzt man die CMD. Z.B. so:

```
univention-domain-join-cli --username administrator --skip-login-manager --
master-ip 192.168.1.9
```

Den Benutzer den man hier verwendet muss sich per SSH am Master einloggen können und dort Rootrechte besitzen. Nur die Mitgliedschaft in der Gruppe der „Domain Admins“ genügt nicht.

Wichtig für WLAN-Geräte

Mit WLAN den Join zu vollziehen kann je nach Hersteller ein Problem darstellen. Daher die Empfehlung den Join immer über Kabel durchführen. Oder wenn es kein Kabel gibt, vorher mit wpa_supplicant eine Verbindung herstellen.

Mit bestehendem lokalem Benutzerprofil umziehen

Es kann vorkommen das lokal ein Benutzer existiert wo man das Profil zum Teil, oder ganz mit übernehmen möchte. Ist das der Fall legt man einen temporären User lokal an und vergibt diesem Sudo-Rechte. Mit diesem Benutzer führt man dann den Joinprozess durch.

Möchte man nur einen Teil des alten Home's mit übernehmen, so benennt man dieses Verzeichnis einfach um. z.B. `mv /home/myuser /home/myuser_old`. Nach dem Login mit seinem neuen Domainbenutzer wird wieder ein frisches Profil angelegt. ACHTUNG bei KDE. Es gibt dort eigene Mechanismen zum Erstellen von Profilen. Wer das `/etc/skel (Benutzerprofilvorlage)` gerne kopiert haben möchte, muss sich als erstes mit SSH am lokalem Gerät einloggen. z.B. `ssh myuser@localhost` Dann wird immer SKEL kopiert.

Einfache Sudorechte

UCS und Sudo auf Ubuntu-Clients

Loginvertrauen

Dieser Teil ist optional, bitte lese diesen genau durch und entscheide selbst was für dich relevant ist.

Bindet man einen Linuxclient per LDAP an kann man alle Benutzer mit einem Realm bestimmen. Somit können sich auch nur diese User auf der Maschine einloggen. Bedient man sich aber der Kerberosanbindung, können sich alle User in der Domäne auf Clients einloggen. Dies ist sehr unschön. Für Ubuntu/Linuxclients die also die volle Domänenanbindung mit Kerberos und SSO geniesen, bedient man sich ganz einfach PAM.

Zuerst erstellt man sich ein File:

```
nano /etc/security/access-login.conf
```

Mit folgendem Inhalt:

access-login.conf

```
# Warning: This file is auto-generated and might be overwritten by
#          univention-config-registry.
#          Please edit the following file(s) instead:
# Warnung: Diese Datei wurde automatisch generiert und kann durch
#          univention-config-registry ueberschrieben werden.
#          Bitte bearbeiten Sie an Stelle dessen die folgende(n)
Datei(en):
#
#       /etc/univention/templates/files/etc/security/access-login.conf
#

+:Administrators,root,Domain Admins,firmen-mitarbeiter-
user,backuppc:ALL
-:ALL:ALL
```

Das heißt nur Mitglieder von den genannten Gruppen `Administrators,root,Domain Admins,firmen-mitarbeiter-user` dürfen sich einloggen. Die nächste Zeile besagt „alles andere verweigern“. Dies gilt für alles Services, also auch IMAP, SSH, usw. Man kann auch einzelne Services verwenden. Da diese Config aber nur Clientmaschinen betrifft, sollte das so genügen. Diese Config wird dann mit Foreman ausgerollt. Gut. Also nächstes wird das File noch in der Datei `/etc/pam.d/common-account` definiert. Hierfür fügen wir diese eine Zeile ein.

common-account

```
#
# /etc/pam.d/common-account - authorization settings common to all
services
#
# This file is included from other service-specific PAM config files,
# and should contain a list of the authorization modules that define
# the central access policy for use on the system.  The default is to
# only deny service to users whose accounts are expired in /etc/shadow.
#
# As of pam 1.0.1-6, this file is managed by pam-auth-update by
default.
# To take advantage of this, it is recommended that you configure any
```

```
# local modules either before or after the default block, and use
# pam-auth-update to manage selection of other modules.  See
# pam-auth-update(8) for details.
#
# here are the per-package modules (the "Primary" block)
account [success=1 new_authtok_reqd=done default=ignore]
pam_unix.so
# here's the fallback if no module succeeds
account requisite                       pam_deny.so
# prime the stack with a positive return value if there isn't one
already;
# this avoids us returning an error just because nothing sets a success
code
# since the modules above will each just jump around
account required                        pam_permit.so
# and here are more per-package modules (the "Additional" block)
account sufficient                      pam_localuser.so

account required pam_access.so accessfile=/etc/security/access-
login.conf listsep=, maxent=0x400001

account [default=bad success=ok user_unknown=ignore]    pam_sss.so
# end of pam-auth-update config
```

Ab dem Zeitpunkt ist die Config gültig.

Sicherheit HOME

Möchte man das ein Home mit sämtlichen Daten von „/etc/skel“ beim Ersten Login angelegt wird muss man Pam konfigurieren.

```
nano /etc/pam.d/common-session
...
session         required        pam_mkhomedir.so skel=/etc/skel umask=0007
...
```

Hier ist drauf zu achten dass, das ganze and der richtigen Stelle steht. Beispiel Datei mit UCS Anbindung:

common-session

```
#
# /etc/pam.d/common-session - session-related modules common to all
services
#
# This file is included from other service-specific PAM config files,
# and should contain a list of modules that define tasks to be
```

```
performed
# at the start and end of sessions of *any* kind (both interactive and
# non-interactive).
#
# As of pam 1.0.1-6, this file is managed by pam-auth-update by
default.
# To take advantage of this, it is recommended that you configure any
# local modules either before or after the default block, and use
# pam-auth-update to manage selection of other modules.  See
# pam-auth-update(8) for details.

# here are the per-package modules (the "Primary" block)
session [default=1]                     pam_permit.so
# here's the fallback if no module succeeds
session requisite                       pam_deny.so
# prime the stack with a positive return value if there isn't one
already;
# this avoids us returning an error just because nothing sets a success
code
# since the modules above will each just jump around
session required                        pam_permit.so
# The pam_umask module will set the umask according to the system
default in
# /etc/login.defs and user settings, solving the problem of different
# umask settings with different shells, display managers, remote
sessions etc.
# See "man pam_umask".
session optional                        pam_umask.so
# and here are more per-package modules (the "Additional" block)
session required    pam_mkhomedir.so umask=0007 skel=/etc/skel
session required        pam_unix.so
session optional                        pam_sss.so
session optional        pam_mount.so
session optional        pam_systemd.so
# end of pam-auth-update config
```

Somit wird das Home wie folgt mit dem Benutzerprofil von `/etc/skel` erstellt:

```
ls -l /home
insgesamt 36
drwx------ 18 backuppc backuppc        4096 Sep 20  2018 backuppc/
drwxrwx--- 45 peter     peter-maingroup  4096 Mai 20 10:41 peter/
drwxrwx--- 22 hans     hans-maingroup  4096 Jun 14 10:19 hans/
drwxrwx--- 24 susi     susi-maingroup  4096 Jun 14 12:54 susi/
```

SAML in Firefox und Chrome

In der erweiterten Firefox Konfiguration, diese ist erreichbar über die Eingabe von **about:config** in

der Firefox Adresszeile, muss bei der Option **network.negotiate-auth.trusted-uris** und **network.negotiate-auth.delegation-uris** die Adresse des Identity Providers eingetragen werden, also in der Standardeinstellung **ucs-sso.domainname**.

In Chrome und Chromium lässt sich das viel einfacher per Files managen.

Chromium:

Folgende Datei ist zu erstellen:

```
nano /etc/chromium-browser/default
```

Mit diesem Inhalt:

```
# Default settings for chromium-browser. This file is sourced by /bin/sh
from
# /usr/bin/chromium-browser

# Options to pass to chromium-browser
CHROMIUM_FLAGS="--auth-server-whitelist=ucs-sso.mydomainname --auth-
negotiate-delegate-whitelist=ucs-sso.mydomainname"
```

Chrome

```
mkdir -m 755 -p /etc/opt/chrome/policies/managed
```

```
nano /etc/opt/chrome/policies/managed/kerberos.json
```

```
{
     "AuthServerWhitelist": "*.sso.uni-erlangen.de,*.sso.fau.de",
     "AuthNegotiateDelegateWhitelist": "*.sso.uni-erlangen.de,*.sso.fau.de"
}
```

Oder nur eine Domäne:

```
{
     "AuthServerWhitelist": "ucs-sso.mydomainname",
     "AuthNegotiateDelegateWhitelist": "ucs-sso.mydomainname"
}
```

```
chmod o+rx /etc/opt/chrome/policies/managed/kerberos.json
```

Sofern die UCR-Variable für Kerberos laut Doku gesetzt wurde ist der Login ohne Passwort mit Kerberosticket ab sofort aktiv. Die Timeouts des Webinterface sollte man wohl von 5 Minuten auf z.B. 12 Stunden mit „ucr set“ stellen. ;)

Quellen:

- https://www.anleitungen.rrze.fau.de/betriebssysteme/linux/kerberos-und-websso/
- https://docs.software-univention.de/handbuch-4.4.html#domain:saml