invis Server - Client Integration

Windows - ab Windows 7

1. Hintergrund: Server-gespeicherte Profile

Im Unterschied zu Windows XP und 2000 birgt Windows 7 (wie auch der Vorgänger Vista) ein verstecktes Problem in sich, wenn mit sogenannten *Roaming Profiles* also Server-gespeicherten Profilen innerhalb einer Domänen-Struktur gearbeitet wird.

Der spezielle Systemordner „Eigene Dateien“ wurde durch sogenannte Bibliotheken (virtuelle Verzeichnisse, die sich real aus mehreren Verzeichnissen zusammensetzen können) ersetzt. Ziel des Ganzen ist die Verbesserung der Desktopsuche, also das Auffinden von Informationen auf dem eigenen System. Um eine schnelle und semantische Suche zu ermöglichen werden die beteiligten Verzeichnisse indiziert, also deren Inhalte anhand von Schlagworten und anderen Informationen in einer Index-Datenbank geführt. Das Problem dabei ist, dass die Indizierung nur auf lokalen Laufwerken möglich ist.

Die persönlichen Bibliotheken sind genau wie ihr Vorgänger der Systemordner „Eigene Dateien“ im persönlichen Benutzerprofil der Anwender gespeichert. Dieses Profil wird, so es als Roaming Profile geführt wird beim An- und Abmelden des Benutzers an der Domäne mit seinem Pendant in der Freigabe „\\profiles“ auf dem Domain-Controller synchronisiert.

Wächst die Datenmenge im Profil an, dauern An- und Abmeldung immer länger. (20 Minuten für den Anmeldevorgang habe ich schon beobachtet). Mit Windows XP und 2000 war es einfach das Ziel des Ordners „Eigene Dateien“ in ein Verzeichnis außerhalb des Profils, bestenfalls in ein persönliches Verzeichnis auf einem Fileserver zu verschieben. Genau das lässt sich mit den Bibliotheken unter W7 und Vista aufgrund der Indizierung nicht machen.

Umgehen lässt sich dieses Problem nur auf Kosten der Indizierung. Mit Einführung des Dateisystems NTFS5 (mit Windows Vista) beherrscht NTFS sogenannte Hard- und Softlinks, wie sie unter Linux- bzw. UNIX-Dateisystemen schon lange üblich sind.

Meldet sich ein neuer Benutzer erstmalig an seinem PC an, wird sein Profil und damit auch die Bibliotheksverzeichnisse erstellt. Verfügt der Benutzer über ein ihm gehörendes persönliches Verzeichnis auf einem Fileserver können dort Pendants der Verzeichnisse (Dokumente, Musik, Videos. usw.) erstellt werden. Anschließend werden die lokalen Verzeichnisse gelöscht und mit dem Kommando ***mklink*** stattdessen Verknüpfungen erzeugt die auf die Entsprechungen im persönlichen Verzeichnis zeigen. Windows wird sich nicht darüber beklagen, dass dort keine Indizierung möglich ist.

2. Vorgehensweise - Domänenbeitritt

Für den Domänenbeitritt müssen die Zugangsdaten eines zum Anlegen von Benutzerkonten berechtigten Benutzers auf dem Domain-Controller bekannt sein. Für einen Windows PDC ist dies in aller Regel der zentrale Administrator und für einen Samba PDC ist dies der Benutzer *root*.

Domänenbeitritte können nicht mit den Personal-Versionen von Windows durchgeführt werden (OK – Heise hat gezeigt, dass es mit XP Home mit ein bisschen getrickse doch geht.).

1. **(Veraltet)** Ist der PDC ein Samba-PDC mit NTLM Domäne (**invis Classic**) müssen zunächst zwei Einträge in der Win7 Registry geändert werden. Einen fertigen Patch gibt es hier. Ein Windows Neustart ist nach Einspielen des Patches nicht nötig. In Verbindung mit Active-Directory Domänen entfällt dieser Schritt. Mit Windows 10 ist der Beitritt in eine klassische NT4.0 Domäne nicht mehr möglich. Mal ehrlich, das sollte eigentlich auch in der Praxis nicht mehr geben.
2. Unter Windows 7 lässt sich die Funktion „Einer Domäne Beitreten“ am besten über die Desktop-Suche finden. Einfach *Domäne* in der Suchleiste eingeben und dann auf den entsprechenden Menüeintrag klicken.
3. Als nächstes auf die Schaltfläche „Ändern“ klicken und dann das Feld „Domäne“ aktivieren, den Domänennamen eintragen und mit OK bestätigen.
4. W7 fragt jetzt nach Benutzernamen und Passwort eines berechtigten Benutzerkontos, also *root* oder *Administrator*. Eingeben und mit OK bestätigen.
5. Stimmen die Zugangsdaten wird der Domänenbeitritt nach einigen Sekunden bestätigt. Im Falle eines Samba PDCs erscheint im Anschluss daran noch eine Fehlermeldung, da der Client nicht im DNS-Dienst des Servers eingetragen werden konnte. Diese Fehlermeldung kann ignoriert werden.
6. Jetzt ist das System neuzustarten.
7. Vor der ersten Anmeldung eines normalen Benutzers auf diesem Rechner sollte sich nach dem Neustart zunächst ein Domänenadministrator am System anmelden und die Domänen-Gruppe „Domain Users“ bzw. „Domänen Benutzer“ der lokalen Gruppe „Hauptbenutzer“ zuordnen. Ohne diesen Schritt werden einige (vor allem ältere) Programme nicht richtig funktionieren. In extremen Fällen muss die genannte Domänen-Gruppe sogar der lokalen Administratoren-Gruppe zugeordnet werden. Den Zugriff auf „lokale Gruppen“ hat MS bei Windows 7 gut versteckt: Rechtsklick auf „Computer“ im Startmenü, dann Linksklick auf „Verwaltung“. Im neuen Fenster Linksklick auf den Menüeintrag „lokale Benutzer und Gruppen“ und Doppelklick auf Gruppen. Jetzt Doppelklick auf den Eintrag „Hauptbenutzer“, dann im neuen Fenster die Schaltfläche „Hinzufügen“ klicken, gefolgt einem Klick auf die Schaltflächen „Erweitert“ und „Jetzt Suchen“ im jeweils neuen Fenster. Daraufhin erscheint eine Liste aller bekannten Benutzer und Gruppen. Hier die Gruppe der Domänenbenutzer auswählen und alles mit OK bestätigen bis alle Fenster wieder zu sind.
8. Domänenadmin abmelden und mit normalem Benutzer anmelden.
9. Jetzt können die Bibliotheken wie oben beschrieben auf den Server verlegt werden. Dies geschieht am besten per Script (Beispiel siehe unten). Das Script sollte direkt auf Laufwerk *c:* liegen und muss als Administrator ausgeführt werden. Dazu in der Suchleiste „cmd“ eingeben und den gefundenen Menüeintrag mit der rechten Maustaste anklicken und dann mit Linksklick auf „Als Administrator ausführen“ öffnen. Es wird mach Benutzername und Passwort gefragt. Hier müssen die Daten des Domänenadministrators eingegeben werden. Anschließend das Script ausführen. **Voraussetzung dafür ist, dass die Zielverzeichnisse für die zu erstellenden Verknüpfungen bereits existieren.**
10. Meldet das Script keine Fehler ist der Domänenbeitritt abgeschlossen.

3. Fehlerquellen

Es gibt einige Ursachen für ein Fehlschlagen des Domänenbeitrittes. Die Häufigste darunter ist eine vor dem Domänenbeitritt erfolgte Anmeldung am Domain-Controller unter X-beliebiger Benutzerkennung, etwa um auf ein freigegebenes Verzeichnis zuzugreifen.

In diesem Fall wird der Domänenbeitritt verweigert. Es genügt sich am PC ab- und wieder anzumelden um alle Netzwerkverbindungen zu kappen und dann den Beitritt erneut zu starten.

4. Script zum Verschieben der Bibliotheken

```
@echo off
rem Dieses Script loescht die Ordner documents, videos, downloads, music
rem und pictures aus dem Benutzerverzeichnis des aktuellen Benutzers und
rem ersetzt sie durch symbolische links
rem (c) 2010 Stefan Schaefer -- invis-server.org
rem (c) Questions: http://forum.invis-server.org
rem License: GPLv3
echo on

rmdir /S /Q c:\users\USER\documents
rmdir /S /Q c:\users\USER\downloads
rmdir /S /Q c:\users\USER\pictures
rmdir /S /Q c:\users\USER\videos
rmdir /S /Q c:\users\USER\music

mklink /d c:\users\USER\documents u:\Documents
mklink /d c:\users\USER\downloads u:\Downloads
mklink /d c:\users\USER\pictures u:\Bilder
mklink /d c:\users\USER\videos u:\Videos
mklink /d c:\users\USER\music u:\Musik
```

Dieses Script ist für den jeweiligen Benutzer anzupassen. Statt „USER“ ist der jeweilige Benutzername einzutragen und der Zielpfad für die Links ist anzupassen. Im Beispiel ist Laufwerk *u:* das vom Fileserver gemappte Home-Laufwerk des Benutzers.

Achtug: Das Script darf nur ausgeführt werden, wenn die lokalen Verzeichnisse des Benutzers leer sind! Also entweder bevor der Benutzer sich selbst das erste mal anmeldet oder nachdem die darin enthaltenen Dateien auf den Server verschoben wurden. *rmdir* löscht ohne Rückfrage.

Ist der Server ein „invis Server“, wird ein bereits personalisiertes Script beim Anlegen eines neuen Benutzers über das invis Portal automatisch erzeugt und in der Freigabe *Service* im Unterverzeichnis

```
winscripts
```

abgelegt. In der Service-Freigabe wird auch der oben erwähnte Registry-Patch vorgehalten.

Linux Clients mit SSSD integrieren

Anbindung von Linux-Clients an einen invis-Server. Die folgende Beschreibung orientiert sich an **Ubuntu 12.04 LTS** und **Linux Mint 17 „Qiana“** (Getestet auch mit Linux Mint 13 „Maya“), gilt aber prinzipiell auch für andere Linux-Distributionen.

Kern der Beschreibung ist die Anbindung eines Linux-Clients an eine LDAP-gestützte Benutzerverwaltung mit Hilfe des noch recht jungen System Security Services Daemon (SSSD). Nicht alle Distributionen haben den SSSD bereits vollständig integriert, daher muss in vielen Fällen wie beim hier gezeigten Ubuntu die Einrichtung vollständig manuell durchgeführt werden.

SSSD erschließt der Linux-Welt einen Vorteil, der bisher nur Windows-Nutzern zugänglich war. Er bietet die Möglichkeit einer Offline-Anmeldung. D.h. Ein Benutzerkonto steht auf einem Client auch dann zur Verfügung, wenn keine Verbindung zum Server besteht. Spannend vor allem für die Nutzung auf Notebooks.

Um das zu erreichen baut und pflegt SSSD lokal einen Cache der Benutzerverwaltung. Nachteilig daran ist bisher, dass es nicht möglich ist ein Aktualisieren des Caches zu triggern. Daraus resultiert oft eine zeitliche Verzögerung, bis Änderungen, die an der Benutzerdatenbank des Servers vorgenommen wurden auf dem Client zur Verfügung stehen. Wird beispielsweise auf dem Server eine neuer Benutzer angelegt, können einige Minuten vergehen, bis sich der Benutzer tatsächlich anmelden kann.

Achtung: *Bevor Sie damit beginnen Linux-Clients in Ihr Netzwerk zu integrieren, müssen Sie auf Ihrem invis-Server den NFS-Dienst aktiviert haben. Wie das geht, lesen Sie* ***hier***

openSUSE Leap

Für openSUSE Leap basierte Clients stellen wir ein eigenes Setup-Paktet über unsere Repositories zur Verfügung, mit dem sich der ganze Vorgang der Client-Anbindung automatisieren lässt. Unter der aktuellen openSUSE Leap 42.3 lässt sich das entsprechende Repository wie folgt einrichten:

```
invis-client:~ # zypper ar
https://download.opensuse.org/repositories/spins:/invis:/unstable/openSUSE_L
eap_42.3/spins:invis:unstable.repo
```

Danach kann das Setup-Paket dann installiert werden:

```
invis-client:~ # zypper ref
invis-client:~ # zypper in invisAD-client-1
```

Jetzt ist lediglich das Client-Integrationsscript auszuführen. sie benötigen dafür das Passwort des Domänen-Administrators und den Namen der Domäne.

Hinweis: *Sollte Ihr openSUSE Leap System als Offline-Client handeln, müssen einzelne Schritte (wie weiter unten beschrieben) noch manuell vorgenommen werden. Darunter beispielsweise das Erzeugen des VPN-Clientzertifikats. Das Script* ***joininvis*** *fragt, ob es sich bei dem System um einen PC, ein Notebook oder einen „Memberserver“ handelt. Im Falle eines Notebooks wird openvpn automatisch installiert. Auch das weiter unten beschriebene Schript* ***invisconnect*** *ist bereits im Paket enthalten.*

```
invis-client:~ # joininvis
```

Beantworten Sie die Fragen und das wars.

Manuelle Anbindung - online Clients

1. Freigaben einbinden

Zur Anbindung eines Linux-Clients an den Server müssen zunächst die Server-Freigaben **home** und **shares** per NFSv4 ins lokale Verzeichnissystem eingehängt werden. Dazu sind in der Datei

```
/etc/fstab
```

folgende Einträge vorzunehmen:

```
invis.invis-net.loc:/home /home nfs4 defaults 0 0
invis.invis-net.loc:/shares /mnt/invis/shares nfs4 defaults 0 0
```

Dabei ist zu beachten, dass die Home-Verzeichnisse ggf. vorhandener Benutzer durch das Einhängen der Home-Freigabe des Server überdeckt werden. Werden weiterhin lokale Benutzer benötigt, so sollten deren Home-Verzeichnisse vorher nach

```
/local/home
```

verschoben werden.

Weiterhin muss das Zielverzeichnis zum Einhängen der Server-Freigabe „Shares“ zunächst manuell angelegt werden.

Hinweis: *Zum Einhängen von NFS-Freigaben wird das Paket nfs-common benötigt, welches nicht immer zum Standardinstallationsumfang einer Linux-Installation gehört.*

```
linux:~ # sudo apt-get install nfs-common
```

2. Benutzerverwaltung

Die folgenden Erläuterungen beschreiben die Anbindung eines Linux-Clients an einen invis-Server, sowohl AD, als auch Classic. Für die Anbindung von openSUSE Leap (42.x) an einen invis AD Server steht in unserem Github-Repository inzwischen ein Client-Setup-Script zur Verfügung, welches die Anbindung vollständig automatisch durchführt.

https://github.com/invisserver/invisAD-client

Für die Anbindung eines Linux-Clients an einen invisAD-Server empfiehlt sich aktuell die Verwendung des SSS-Daemons. Dieser ist ggf. manuell nachzuinstallieren:

Ubuntu 16.04

```
linux:~ # sudo apt install -y sssd sssd-ad sssd-tools
```

Linux Mint 17 Qiana, 17.1 Rebecca und Linux Mint Debian

```
linux:~ # sudo apt install -y sssd sssd-ad sssd-tools libnss-sss libpam-sss
libsss-sudo libsasl2-modules-gssapi-heimdal
```

openSUSE Leap 42.x

```
linux:~ # zypper in sssd sssd-ad sssd-tools cyrus-sasl-gssapi krb5-client
```

Achtung: *Der* ***sss*** *Dienst und der auf den meisten Linux-Distributionen vorinstallierte* ***Name-Service-Cache-Daemon*** *(nscd) konkurrieren in einigen Funktionen. Daher muss* ***nscd*** *auf jeden Fall deinstalliert werden.*

openSUSE Leap 42.x

```
linux:~ # zypper rm nscd
```

Danach ist unter dem Namen

```
/etc/sssd/sssd.conf
```

eine Konfigurationsdatei für den Daemon anzulegen:

sssd-Konfiguration invis-Server Active Directory

```
[sssd]
services = nss, pam
config_file_version = 2
domains = invis-net.loc
#debug_level = 0x0370

# globale Cache Steuerung
# alle Angaben in Sekunden
# default = 120
enum_cache_timeout = 10

# default = 15
entry_negative_timeout = 5

[nss]

[pam]

[domain/invis-net.loc]
# Domain bezogene Cache Steuerung
# Alle Angaben in Sekunden
# Default = entry_cache_timeout = 5400
entry_cache_user_timeout = 10
entry_cache_group_timeout = 10

# GPO checks abschalten
```

```
# kann in spaeteren sssd Versionen auf "permissive" gesetzt werden
ad_gpo_access_control = disabled

# Using id_provider=ad sets the best defaults on its own
id_provider = ad
# In sssd, the default access provider is always 'permit'. The AD access
# provider by default checks for account expiration
access_provider = ad
auth_provider = ad

# Uncomment to use POSIX attributes on the server
# Auf false gesetzt, damit sssd die UID und GID Nummern
# des Active Directories zu verwenden, statt selbst
# ein id-Mapping vorzunehmen.
# Auf diese Art sind die UIDs und GIDs konsistent.
ldap_id_mapping = false

# Uncomment if the client machine hostname doesn't match the computer object
on the DC.
#ad_hostname = invisad.invis-net.loc

# Uncomment if DNS SRV resolution is not working
#ad_server = invisad.invis-net.loc

# Uncomment if the domain section is named differently than your Samba
domain
#ad_domain = invis-net.loc

# Aktivieren, um Benutzer und Gruppen mit getent
# anzeigen zu koennen.
# Enumeration is discouraged for performance reasons.
enumerate = true
```

Damit **sssd** starten kann müssen unabhängig von der verwendeten invis-Server Version die Zugriffsrechte angepasst werden:

```
linux:~ # sudo chmod 0600 /etc/sssd/sssd.conf
```

Jetzt kann **sssd** gestartet werden:

```
linux:~ # sudo service sssd start
```

Es dauert einige Minuten, bis die Benutzer und Gruppen des ADs von sssd via PAM angezeigt werden. Abgefragt werden die Benutzer und Gruppen mit folgenden Kommandos:

```
linux:~ # getent passwd
linux:~ # getent group
```

Hinweis: *Wenn sssd auch nach verstreichen mehrerer Minuten weder Benutzer noch Gruppen anzeigt und im System-Log die nachfolgende Meldung angezeigt wird, kann die Ursache eine falsche*

*Reverse-DNS Auflösung des AD DC Host-Namens sein. Die IP-Adresse des Domain-Controllers muss bei einer Reverse-DNS Anfrage **zwingend** immer als erstes den Hostnamen des Domain-Controllers liefern.*

```
GSSAPI Error: Unspecified GSS failure.  Minor code may provide more
information (Server not found in Kerberos database)
```

Grundsätzlich ist der SSSD relativ sparsam mit dem Schreiben von Fehlerprotokollen, was natürlich dei Fehlersuche nicht gerade erleichtert. Um den Loglevel auf ein brauchbares Niveau anzuheben hat sich in den einzelnen Sektionen der Konfigurationsdatei folgende Einstellung bewährt:

```
debug_level = 0x0370
```

Der Debug-Level kann für die einzelnen Sektionen „[sssd]“, „[nss]“, „[pam]“ und „[domain/...]“ auch individuell gesetzt werden. Weitere Infos zum Debug-Level sind in der Manpage des Dienstes zu finden (https://jhrozek.fedorapeople.org/sssd/1.13.1/man/sssd.conf.5.html)

Name-Service-Switch anpassen

Damit PAM weiss, dass Benutzerkonten sowohl in der lokalen „/etc/passwd“ als auch im Active-Directory bzw. OpenLDAP des invis-Servers geführt werden, muss die Datei

```
/etc/nsswitch.conf
```

wie folgt angepasst werden:

```
...
passwd: compat sss
group:  compat sss
...
```

Beide gezeigten Einträge müssen um den Wert „sss“ ergänzt werden.

PAM konfigurieren

Auch die PAM Module für die Benutzeranmeldung am System müssen an den **sssd** angepasst werden. Dies kann im einfachsten Fall mit Hilfe des Tools ***pam-config*** vorgenommen werden:

```
linux:~ # pam-config --add --sss
```

Linux Mint (und daher vermutlich auch Ubuntu, man entschuldige hier meine Unwissenheit...) verfügen alternativ über das Programm ***pam-auth-update*** mit dem die PAM-Konfiguration recht komfortabel generiert bzw. aktualisiert werden kann. Bei dessen Verwendung ist (so nicht bereits geschehen, einfach der Punkt „SSS authentication“ im Auswahlmenü zu aktivieren.

Leider haben nicht alle Linux-Distributionen entsprechende Tools im Gepäck. Ist dies der Fall und PAM wird nicht beim Installieren neuer PAM-Module automatisch von der verwendeten Linux-Distribution angepasst, müssen die relevanten PAM-Konfigurationen händisch angepasst werden. Dies sind in aller

Regel 4 Dateien im Verzeichnis:

```
/etc/pam.d
```

Hier Beispielhaft die Dateien aus Linux Mint 17.3, relevant sind immer die Zeilen in denen auf **pam_sss.so** Bezug genommen wird:

common-account

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

common-auth

```
...
# here are the per-package modules (the "Primary" block)
auth    [success=2 default=ignore]      pam_unix.so nullok_secure
auth    [success=1 default=ignore]      pam_sss.so use_first_pass
# here's the fallback if no module succeeds
auth    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
auth    required                        pam_permit.so
# and here are more per-package modules (the "Additional" block)
auth    optional        pam_ecryptfs.so unwrap
auth    optional                        pam_cap.so
# end of pam-auth-update config
```

common-password

```
...
# here are the per-package modules (the "Primary" block)
password        requisite                       pam_pwquality.so retry=3
password        [success=2 default=ignore]      pam_unix.so obscure
use_authtok try_first_pass sha512
password        sufficient                      pam_sss.so use_authtok
# here's the fallback if no module succeeds
password        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
password        required                        pam_permit.so
# and here are more per-package modules (the "Additional" block)
password        optional        pam_gnome_keyring.so
password        optional        pam_ecryptfs.so
# end of pam-auth-update config
```

common-session

```
...
# 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_unix.so
session optional                        pam_sss.so
session optional        pam_systemd.so
session optional        pam_ecryptfs.so unwrap
session optional                        pam_ck_connector.so nox11
# end of pam-auth-update config
```

Kerberos Konfiguration für Client-Anbindung an invis-Server ActiveDirectory

Zur Anbindung eines Linux-Clients an einen invis-Server ActiveDirectory ist es mit der bloßen sssd-Konfiguration nicht getan. Client-Seitig wird darüber hinaus eine Kerberos-Konfiguration sowie eine Kerberos-Keymap benötigt.

Die Konfigurationsdatei ist recht simpel aufgebaut. Legen Sie, so nicht bereits vorhanden die Datei

```
/etc/krb5.conf
```

an:

```
[libdefaults]
    default_realm = INVIS-NET.LOC
    dns_lookup_realm = true
    dns_lookup_kdc = true
```

Um an eine gültige Client-Keytab zu gelangen sind mehrere Schritte notwendig. Zunächst muss der Linux-PC der Active-Directory Domäne beitreten, das dazu benötigte Kommando ***net*** dürfte in den meisten Linux-Distributionen zur Standard-Installation gehören, es ist Bestandteil des Paketes „samba-client“. Vor dem Domänenbeitritt ist eine minimale Samba-Konfiguration anzulegen. Tragen Sie in die Datei

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

folgendes ein:

```
[global]
   security = ads
   realm = INVIS-NET.LOC
   workgroup = INVIS-NET

   kerberos method = secrets and keytab

   client signing = yes
   client use spnego = yes
```

Weitere Einträge sind nicht notwendig. Es folgt der Domänenbeitritt:

```
linux:~ # kinit administrator
net ads join -k
```

Sie benötigen dafür das Passwort des Domänen-Administrators. Gelingt der Domänenbeitritt, wird automatisch für den Client eine Keytab Datei als

```
/etc/krb5.keytab
```

angelegt.

Sollte diese Keytab versehentlich gelöscht werden kann sie auf dem Server unter Verwendung des ***samba-tools*** wiederhergestellt werden:

```
linux:~ # samba-tool domain exportkeytab --principal=CLIENTHOSTNAME$ >
krb5.keytab
```

Kopieren Sie diese Datei dann einfach nach

```
/etc
```

auf dem Client.

Hinweis: *Wenn es sich beim Client-PC um einen Dualboot-system mit Windows und Linux Betrriebssystem handelt, genügt es den Domänenbeitritt mit Windows vorzunehmen und anschließend wie zuvor gezeigt eine Keytab zu generieren und in das Linux-Betriebssystems zu kopieren. Alle anderen Konfigurationsschritte müssen auf dem Linux-Client trotzdem durchgeführt werden.*

3. Test der Konfiguration

Mit

```
linux:~ # getent passwd
```

kann überprüft werden, ob die Benutzerkonten aus dem LDAP-Verzeichnis des Servers zur Verfügung stehen. Dabei ist zu beachten, dass es einen Moment dauern kann, bis sssd alle Benutzer und Gruppen der Domäne anzeigt.

Hat alles funktioniert, muss dafür gesorgt werden, dass **sssd** automatisch beim Systemstart startet:

```
linux:~ # sudo update-rc.d sssd defaults
```

Manuelle Anbindung - offline Clients mit OpenVPN

Allem voran, sollte klar sein, dass ein solches Setup nur dann wirklich Sinn macht, wenn der invis-Server mit einer akzeptablen DSL-Anbindung (>= 16000kBit/s) ausgestattet ist. Zu bedenken ist hier, dass der Upload der begrenzende Faktor ist und der liegt bezogen auf eine 16000er ADSL Leitung bei nur 1000kBit/s. Alles darunter wird sehr zäh! Bei einer 6000er DSL Leitung schrumpft der Upload beispielsweise schon auf magere 256kBit/s.

Die nachfolgende beschriebene Vorgehensweise ist derzeit noch vollständig manuell durchzuführen und birgt durchaus Potential für Fehler. Problematisch ist vor allem die Reihenfolge der einzelnen Schritte.

Hintergründe

Geeignet ist das folgende Setup für Notebooks die unterwegs genutzt werden sollen oder den PC im Home-Office.

Ziel des Setups ist es von Ferne via OpenVPN und NFS vollständigen Zugriff auf den Datenbestand und die Webapplikationen des invis-Servers zu erhalten. Für diesen Zweck wird lokal ein Benutzerkonto benötigt, da zum Zeitpunkt des Logins die Verbindung zum Server noch nicht zur Verfügung steht. D.h. es wird vor allem ein lokales Home-Verzeichnis benötigt, da ohne ein Solches, ein Login an einer graphischen Oberfläche unter Linux nicht möglich ist.

Geht man wie oben beschrieben von einer Ubuntu- oder Mint-Installation aus, wird zum Anlegen dieses Home-Verzeichnisses wie auch zur Konfiguration des SSSD zunächst ein Systemverwalter-Konto benötigt, da ein Root-Zugang unter Ubuntu normalerweise nicht zur Verfügung steht.

Mit diesem Konto kann der Zugriff auf die Benutzerverwaltung des invis-Servers eingerichtet werden. Für ein Notbook kann das der Einfachheit halber direkt im Netzwerk des invis-Servers erfolgen, der PC im Home-Office wiederum benötigt zunächst eine OpenVPN-Verbindung zum invis-Netzwerk. Das Konto wird später nicht mehr benötigt.

Ziel ist unter Anderem auch der Zugriff auf das eigene Home-Verzeichnis auf dem invis-Server. Da dieses zum Zeitpunkt der lokalen Anmeldung aber noch nicht zur Verfügung steht, kann es nicht wie

bei lokalen PCs nach „/home“ gemountet werden.

Damit sind die Schwierigkeiten grob umrissen.

Hinweis: *Wir gehen davon aus, dass die Einrichtung an einem PC erfolgt, der vollständig vom invis-Server Netzwerk getrennt ist, da dies alle Schwierigkeiten berücksichtigt.*

1. VPN Zugang vorbereiten

Auf dem invis-Server muss zunächst ein Schlüssel für die VPN-Verbindung des Clients erzeugt werden. Dazu benötigen Sie lediglich den Hostnamen des Clients, den Sie mit

```
linux:~ hostname -f
```

auf dem Client ermitteln können.

Die Erzeugung des Schlüssels erfolgt auf dem invis-Server unter Verwendung des Scripts ***inviscerts***. Sie benötigen neben dem Hostnamen des Clients auch das Passwort Ihrer CA.

```
invis:~ # inviscerts vpn
```

Das Script fragt nach einem „Export Passwort“ für das zu erzeugende Schlüsselpaar. Schlüssel und Zertifikat werden in einer PKCS12 Datei zusammengefasst und diese mit einem Passwort verschlüsselt. Sie sollten hier ein möglichst sicheres Passwort verwenden und dieses nur dem Inhaber des Clients nennen.

Die fertige Datei finden Sie im Unterverzeichnis

```
/srv/shares/service/VPN-Clients/Zertifikate
```

unter dem Namen:

```
clienthostname.p12
```

Diese Datei ist auf sicherem Weg (z.B. SCP) auf den Client zu übertragen.

2. VPN Zugang herstellen

Melden Sie sich mit Ihrem lokalen Systemverwalter-Konto an und installieren Sie OpenVPN:

```
linux:~ # sudo apt-get install openvpn
```

Nach der Installation benötigen Sie sowohl die zuvor erstellte PKCS12-Datei und eine OpenVPN-Konfiguration. Eine entsprechende Vorlage finden Sie (ab invis-Server Version 9.2 Build 194) unter:

```
/srv/shares/service/VPN-Clients/Linux
```

In dieser Datei müssen Sie lediglich den Namen der PKCS12-Datei anpassen und den im Internet

gültigen Hostnamen Ihres invis-Servers an zwei Stellen kontrollieren:

```
....
# Authentifizierung und Verschluesselung
tls-client
auth SHA512
cipher AES-256-CBC
....
# The hostname/IP and port of the server.
# You can have multiple remote entries
# to load balance between the servers.
remote server.ihredomain.de
....
# Einfacher ist eine PKCS#12 Datei
pkcs12 /etc/openvpn/ihre-domain.loc/clientfqdn.p12
....
```

Kopieren Sie die fertige Datei auf dem Client nach

```
/etc/openvpn
```

und legen Sie unterhalb des Verzeichnisses einen Unterordner unter dem Namen Ihres lokalen Netzwerknamen des invis-Servers an und kopieren Sie die Schlüsseldatei dort hinein:

```
linux:~ # sudo mkdir /etc/openvpn/ihre-domain.loc
linux:~ # sudo cp clientfqdn.p12 /etc/openvpn/ihre-domain.loc
```

Danach kann eine Testverbindung aufgebaut werden.

```
linux:~ # sudo /usr/sbin/openvpn --config /etc/openvpn/vpn-client.ovpn --
daemon --log ovpnsession.log
```

Eine erfolgreiche Verbindung ist Voraussetzung für die Anbindung an die Benutzerverwaltung des invis-Server.

3. SSSD einrichten

Die Einrichtung des SSSD erfolgt in gleicher Weise wie im im Abschnitt „Linux Clients mit SSSD integrieren“ beschrieben.

Wichtig ist in der SSSD-Konfigurationsdatei für diesen Zweck vor allem die folgende Zeile:

```
cache_credentials = true
```

Sie sorgt dafür, dass ein Offline-Login überhaupt möglich ist.

Ist die Anbindung gelungen, was mit

```
linux:~ # getent passwd
```

überprüft werden kann, kann die Anmeldung des invis-Benutzers getestet werden. ***getent*** zeigt alle Benutzerkonten des Servers an, suchen Sie sich aus der Liste den Benutzernamen und die User-ID (UID) des Benutzers den Sie auf dem zu konfigurierenden PC benutzen möchten und legen Sie für Ihn ein leeres Homeverzeichnis an und übereignen es Ihm:

```
linux:~ # sudo mkdir /home/benutzername
linux:~ # sudo chown benutzername."Domain Users" /home/benutzername
```

Sie können jetzt versuchen temporär die Identität des gewünschten Benutzers anzunehmen:

```
linux:~ # su - benutzername
```

Durch das Angeben des „Minuszeichens“ wechseln Sie mit ***su*** in das Home-Verzeichnis des Benutzers. Testen Sie, ob es Ihnen jetzt möglich ist darin Dateien und Ordner anzulegen.

4. Benutzer als Systemverwalter einrichten

Um eine OpenVPN Verwindung aufzubauen muss der Benutzer, der es versucht über Root-Rechte verfügen, da es nur so möglich ist automatisch eine Route ins Zielnetz zu setzen. Um dies auf einem Ubuntu-System zu erreichen, muss der gewünschte Benutzer als „sudoer“ (Systemverwalter) geführt werden. Das geht nicht unbedingt mit den Ubuntu Systemwerkzeugen, da diese in unseren Tests die über SSSD verfügbaren Benutzerkonten nicht immer angezeigt haben.

Einfacher ist es den neuen Benutzer manuell in

```
/etc/group
```

in die gleichen Gruppen einzutragen, in denen auch der bisherige Systemverwalter Mitglied ist. Dies muss vom bisherigen Systemverwalter vorgenommen werden. (Ach wie schön ist openSUSE mit freigeschaltetem root-Konto 😉)

5. Abschluss

Sie können sich jetzt mit dem zuvor getesteten invis-Benutzerkonto am System anmelden. Dabei wird eine neue leere Desktop-Umgebung aufgebaut. Kopieren Sie entweder per SCP oder mit Hilfe eines USB-Sticks das Script ***invisconnect*** vom invis-Server auf Ihren PC. Es liegt genau, wie die vorbereitete OpenVPN-Konfigurationsdatei in der Service-Freigabe unter:

```
/srv/shares/service/VPN-Clients/Linux
```

Hinweis: *invisconnect benötigt uum Einhängen von NFS-Freigaben das Paket nfs-common benötigt, welches nicht immer zum Standardinstallationsumfang einer Linux-Installation gehört. Installieren Sie es ggf. nach.*

```
linux:~ # sudo apt-get install nfs-common
```

Legen Sie es auf Ihrem PC oder Notebook am besten unter

```
/usr/local/bin
```

ab, damit es auch anderen Benutzern zur Verfügung steht.

Testen Sie jetzt den Aufbau der Verbindung:

```
linux:~ # invisconnect c
```

Sie werden vom Script zunächst vom sudoers-Mechanismus nach Ihrem lokalen Passwort gefragt. Nach dessen Eingabe erfolgt unmittelbar die Abfrage des OpenVPN-Schlüsselpasswortes.

Treten beim Verbindungsaufbau keine Fehler auf, werden folgende Verzeichnisse angelegt:

- /mnt/invis/shares – Alle Server-Freigaben
- /mnt/invis/home – Die freigegebenen Home-Verzeichnisse

Diese Verzeichnisse dienen als Mount-Points für das Einhängen der Server-Freigaben. Das Einhängen wird ebenfalls vom Script übernommen. Weiterhin erzeugt das Script temporär eine neue „resolv.conf“ Datei, was dazu führt, dass Ihr invis-Server für die Dauer der OpenVPN-Verbindung als DNS-Server genutzt wird. Dies erleichtert die Nutzung der Dienste des Servers.

Sie können jetzt sowohl auf Ihr Home-Verzeichnis, als auch, gemäß Ihrer Zugriffsrechte, auf die Freigaben des Servers zugreifen. Limitierender Faktor dabei sind die Upload-Bandbreiten der beiden Internet-Verbindungen. Sie müssen sich also ggf. beim Zugriff auf Dateien in Geduld üben.

Auch sollten Sie bei instabilen Internet-Verbindungen über „Online-Bearbeitung“ von Dateien gründlich nachdenken. Kopieren Sie sich Dateien zur Bearbeitung evtl. in Ihr lokales Home-Verzeichnis und anschließend wieder zurück auf den Server.

Nach erledigter Arbeit beenden Sie die Verbindung wie folgt:

```
linux:~ # invisconnect d
```

Dabei wird auch Ihre ursprüngliche „resolv.conf“ wieder hergestellt. Vergessen Sie das Trennen der Verbindung, bleibt die geänderte „resolv.conf“ erhalten, was dazu führt, dass sie keinen gültigen Nameserver ansprechen können. Unter Ubuntu hilft da leider auch kein Neustart oder Trennen und Wiederverbinden Ihrer lokalen Netzwerkverbindung.

Sie müssen die Datei von Hand editieren und folgenden Eintrag hinzfügen:

```
nameserver 127.0.0.1
```

Alle anderen Einträge können Sie löschen.

Achtung: *Das gilt nur für Ubuntu und nicht für openSUSE. Dort genügt es eine fehlerhafte „resolv.conf“ zu löschen und die Netzwerkverbindung kurz zu unterbrechen. Danach wird automatisch eine neue „resolv.conf“ erzeugt.*

Hier noch der Vollständigkeit halber das ***invisconnect*** Script:

```
#!/bin/bash
```

```
# (c) 2015 Stefan Schaefer - invis-server.org
# License: GPLv3

ovpnconf="/etc/openvpn/vpn-client.ovpn"
mntpathbase="/mnt/invis"
domain="invis-net.loc"
invisip="192.168.220.10"
delay=20

usage() {
    echo -e "Geben Sie an, ob Sie die Verbindung zum Server aufbauen oder
trennen möchten."
    echo -e "Verbinden:\tinvisconnect c"
    echo -e "Trennen:\tinvisconnect d"
}

case "$1" in
    "c")
            # OpenVPN Verbindung aufbauen
            sudo /usr/sbin/openvpn --config $ovpnconf --daemon -log
ovpnsession.log
            # invis Server Freigaben einhaengen
            echo "$delay Sekunden Wartezeit, bevor das Script fortgesetzt
wird."
            sleep $delay
            # invis DNS Server nutzen
            # resolv.conf sichern
            sudo mv /etc/resolv.conf /etc/resolv.conf.ori
            # temporaere resolv.conf erzeugen
            sudo echo -e "search $domain" > /tmp/resolv.conf
            sudo echo -e "nameserver $invisip" >> /tmp/resolv.conf
            sudo mv /tmp/resolv.conf /etc/resolv.conf
            # Testen, ob Zielverzeichnisse vorhanden, wenn nicht anlegen.
            if [[ ! -d $mntpathbase ]]; then
                sudo mkdir -p $mntpathbase/shares
                sudo mkdir -p $mntpathbase/home
            fi
            # Freigaben einhaengen
            sudo mount -t nfs $invisip:/srv/nfs4_base/shares
$mntpathbase/shares
            sudo mount -t nfs $invisip:/srv/nfs4_base/home $mntpathbase/home
            ;;
        "d")
            # Freigaben aushaengen
            sudo umount $mntpathbase/shares
            sudo umount $mntpathbase/home
            # OpenVPN beenden
            openvpnpid=$(pgrep openvpn)
            sudo kill $openvpnpid
            # Urspruengliche resolv.conf wiederherstellen
            sudo mv /etc/resolv.conf.ori /etc/resolv.conf
```

```
            ;;
        *)
            usage
            ;;
esac
```