invis-Server Upgrade

In dieser Rubrik wird es Anleitungen, Tipps & Tricks sowie Hintergrundwissen zum Thema „Upgrade“ geben. Anders als bei einem Update geht es beim Upgrade immer um den Umstieg auf neuere Versionen einer Software oder eines Gesamt-Produktes.

Für den invis-Server sind dabei folgende Themen von Interesse:

1. Upgrade von invis-Classic auf invis-AD
2. Upgrade von invis-AD 10.3 auf Basis von openSUSE 13.1 auf eine neuere invis-AD Version unter openSUSE Leap
3. Weiterhin von Bedeutung sind Upgrades relevanter Software-Komponenten wie etwa Zarafa oder Sernet-Samba

An dieser Stelle sollte von vorne herein klar sein, dass es kein direktes Upgrade von einem invis-Classic auf einen invis-AD gibt und geben kann. Es gibt aber eine Reihe von Möglichkeiten Daten aus einer alten in eine neue Installation zu migrieren. Die nachfolgenden Beschreibungen beschreiben die ersten beiden Punkte der vorangegangenen Liste. Punkt 3 der Liste ist dabei Teil des nachfolgenden Kapitels.

invis-AD 10.3 unter openSUSE 13.1 -> invis-AD ab 10.4 unter openSUSE Leap

Der Weg von einem frühen invis-AD hin zu einer aktuellen Version ist vor allem deshalb von Interesse, da sich openSUSE 13.1 bereits in der „Evergreen“ Phase befindet, also den regulären Maintenance Zyklus bereits hinter sich hat. Aber auch die Evergreen-Phase dauert nicht ewig, angedacht ist als deren Ende bereits der November 2016.

Die folgenden Anleitungen sind natürlich eine Momentaufnahme zum Zeitpunkt des Tests. Dinge ändern sich und nicht alle Systeme sind gleich. D.h. es kann beim Nachvollziehen dieser Anleitung zu Problemen kommen, die bei mir nicht aufgetaucht sind. In solchen Fällen ist natürlich nicht alles verloren, sondern es gilt sich Logfiles genau anzuschauen und die Situation zu analysieren. Es sollte selbstverständlich sein, dass ein solches Unterfangen nicht ohne vorherige **vollständige** Datensicherung gestartet werden sollte. Ist Ihr System auf Basis von **RAID1** aufgebaut ist es einfach sich durch auftrennen der RAID-Verbünde einen doppelten Boden zu erhalten. Einfach eine Platte aus dem System nehmen, auf die Seite legen und das Upgrade nur auf der verbliebenen Platte durchführen.

Trotz bestehender Risiken sollten die nachfolgenden Beschreibungen eine gute Basis für ein umfangreiches Upgrade liefern.

Dieser Weg schließt reguläre Updates das Upgrade auf eine aktuelle Zarafa-Version sowie eine aktuelle Sernet-Samba-Version ein. In ersten Tests hat sich gezeigt, dass es unbedingt notwendig ist zunächst Zarafa auf einen aktuellen Stand zu heben. Dieser Schritt **muss** aufgrund eines Problems in unseren invis-Server Software-Repositories auch vor einem einfachen Update erfolgen.

Zarafa Upgrade von Version 7.2.0 auf 7.2.2

Dauer: ca 1 bis 1,5 Stunden

Hinweis: *Die nachfolgende Anleitung gilt auch für ein Upgrade auf neueren invis-AD unter openSUSE Leap.*

Mit Einführung von Zarafa 7.2.2 hat sich die Struktur der Software-Pakete deutlich geändert, es haben sich Pfade geändert, die Zarafa Dienste werden nicht mehr mit „root“ Rechten ausgeführt und Zarafa beherrscht jetzt STARTTLS sowie „Perfect Forward Secrecy“. Kurz es wird ein einfaches Upgrade der Pakete.

Die Schritte im einzelnen:

Sichern Sie zunächst Ihre Zarafa-Datenbank:

```
linux:~ # zdbdump
```

Wenn Sie die invis-Server Zarafa Erweiterungen installiert haben können Sie zusätzlich noch alle Stores manuell sichern:

```
linux:~ # zdbackup
```

Stoppen Sie jetzt alle Zarafa-Dienste:

```
linux:~ # runzarafa stop
```

Deinstallieren Sie den alten Webaccess:

```
linux:~ # zypper rm zarafa-webaccess
```

Dabei werden alle Webacces-Plugin-Pakete ebenfalls deinstalliert.

Installieren Sie jetzt das neue Zarafa-Metapaket:

```
linux:~ # zypper ref
linux:~ # zypper in zarafa-server-packages
```

Dabei stößt ***zypper*** auf einen Paketkonflikt mit dem Paket „php5-mapi-7.2.0“. Wählen Sie hier Lösung 1, die Installation des erwähnten Paketes. Daraus resultiert kein weiteres Problem, da das Paket durch ein neues geringfügig anders benanntes Paket ersetzt wird.

Ist die Installation der neuen Pakete abgeschlossen, müssen manuell alle Reste der alten Paket-Struktur entfernt werden. Das lässt sich am besten per YaST erledigen. Suchen Sie hier in der Software-Verwaltung nach „zarafa“ und entfernen Sie alle Pakete die „7.2.0“ im Namen oder als Versionsnummer tragen.

Dabei wird sich YaST immer wieder wegen scheinbarer Abhängigkeitsprobleme beschweren. Brechen Sie das Fenster mit den Lösungsvorschlägen immer wieder ab und gehen Sie alle alten Pakete durch.

Hinweis: *Sie wählen in YaST ein Paket zur Deinstallation, Installation oder Aktualisierung immer mit der Leertaste aus. Dabei will YaST, wenn ein Paket installiert ist dieses beim ersten Klick auf die Leertaste „aktualisieren“ und meckert ggf. Abhängigkeitsprobleme an. Erst beim zweiten drücken der Leertaste wird ein Paket zur Deinstallation markiert. Erkennbar am „-“ Minuszeichen vor dem Paket. Alle zu deinstallierenden Pakete müssen also in der Liste mit dem „-“ gekennzeichnet sein. Ab dann gibt es auch keine Abhängigkeitsprobleme mehr.*

Passen Sie jetzt die Rechte auf die Konfigurationsdateien, das Attachement-Verzeichnis und die Logfiles an den neuen Benutzer bzw. die neue Gruppe an unter der Zarafa betrieben wird. Achten Sie darauf, dass der Pfad zum Zarafa-Attachements-Verzeichnis auf invis-Servern

```
/srv/zarafa/attachements
```

ist und somit von der Vorgabe abweicht.

```
linux:~ # chown -R zarafa.zarafa /srv/zarafa/
linux:~ # chown -R zarafa.zarafa /var/log/zarafa/
linux:~ # choen -R .zarafa /etc/zarafa/*
```

Jetzt müssen die Konfigurationen einzelner Dienste angepasst werden. Wichtig sind dabei vor allem „zarafa-ical“, „zarafa-gateway“ und der „zarafa-server“ selbst, da sich deren Konfigurationsdateien teils deutlich geändert haben und die neuen Dateien mit der Endung „.rpmnew“ in

```
/etc/zarafa
```

abgelegt wurden.

Sichern Sie jeweils die alte Datei unter einem neuen Namen und aktivieren die jeweils neue Datei:

```
linux:/etc/zarafa # cp ical.cfg ical.cfg.old
linux:/etc/zarafa # cp ical.cfg.rpmnew ical.cfg
```

Wiederholen Sie dies für die Dateien „gateway.cfg“ und „server.cfg“.

Mit Hilfe des Tools ***diff*** können Sie sich jetzt einen Überblick über die Veränderungen schaffen:

```
linux:/etc/zarafa # diff -u ical.cfg.old ical.cfg
```

Sie müssen jetzt Ihre individuellen Konfigurationen in die neue Datei übernehmen. Die wichtigen Anpassungen im Einzelnen:

ical.cfg

```
...
# wether ssl connections can be made to the ical server
icals_enable = yes
...
# File with RSA key for SSL
ssl_private_key_file = /etc/ssl/private/ldap-key.pem

# File with certificate for SSL
```

```
ssl_certificate_file = /etc/ssl/certs/ldap-cert.pem
...
# SSL protocols to use, set to '!SSLv2' for 'ssl_enable_v2 = no'
ssl_protocols = !SSLv2 !SSLv3

# SSL ciphers to use, set to 'ALL' for backward compatibility
ssl_ciphers = ALL:!LOW:!SSLv2:!EXP:!aNULL

# Prefer the server's order of SSL ciphers over client's
ssl_prefer_server_ciphers = yes

############################################################
# OTHER ICAL SETTINGS

# The timezone of the system clock
server_timezone = Europe/Berlin
```

Wiedergegeben sind nur die Teile der Datei, in der Anpassungen vorgenommen wurden.

gateway.cfg

```
...
# enable/disable POP3, and POP3 listen port
pop3_enable     =       no
pop3_port       =       110
...
# enable/disable Secure IMAP, and Secure IMAP listen port
imaps_enable    =       yes
imaps_port      =       993
...
# File with RSA key for SSL
ssl_private_key_file    =       /etc/ssl/private/mail-key.pem

#File with certificate for SSL
ssl_certificate_file    =       /etc/ssl/certs/mail-cert.pem
...
# SSL protocols to use, set to '!SSLv2' for 'ssl_enable_v2 = no'
ssl_protocols = !SSLv2 !SSLv3

# SSL ciphers to use, set to 'ALL' for backward compatibility
ssl_ciphers = ALL:!LOW:!SSLv2:!EXP:!aNULL

# Prefer the server's order of SSL ciphers over client's
ssl_prefer_server_ciphers = yes
```

server.cfg

```
...
# e-mail address of the Zarafa System user
system_email_address    = administrator@yourdomain.tld
```

```
...
# The user under which we connect with MySQL
mysql_user              = zarafa

# The password for the user (leave empty for no password)
mysql_password          = your-secret
...
# When attachment_storage is 'files', use this path to store the files
attachment_path         = /srv/zarafa/attachments
...
# enable SSL support in server
server_ssl_enabled      = yes

# Listen for SSL connections on this port
server_ssl_port         = 237

# Required Server certificate, contains the certificate and the private key
parts
server_ssl_key_file     = /etc/ssl/private/zarafa.pem

# Password of Server certificate
server_ssl_key_pass     = replace-with-server-cert-password

# Required Certificate Authority of server
server_ssl_ca_file      = /etc/ssl/CA/cacert.pem

# Path with CA certificates, e.g. /etc/ssl/certs
server_ssl_ca_path      =

# SSL protocols to use, set to '!SSLv2' for 'server_ssl_enable_v2 = no'
server_ssl_protocols = !SSLv2 !SSLv3

# SSL ciphers to use, set to 'ALL' for backward compatibility
server_ssl_ciphers = ALL:!LOW:!SSLv2:!EXP:!aNULL

# Prefer the server's order of SSL ciphers over client's
server_ssl_prefer_server_ciphers = yes
...
# Name of the plugin that handles users
# Required, default = db
# Values: ldap, unix, db, ldapms (available in enterprise license)
user_plugin             = ldap
...
# location of the zarafa plugins
# if you have a 64bit distribution, this probably should be changed to
/usr/lib64/zarafa
plugin_path             = /usr/lib64/zarafa
...
# Disable features for users. Default all features are disabled. This
# list is space separated. Currently valid values: imap
disabled_features = pop3
```

Passen Sie hier auch die Einstellungen des Abschnitts

```
# CACHE SETTINGS
```

gemäß Ihrer alten Konfiguration an.

In den obigen Dateien wurde das SSL-Setup immer gemäß aktuellem Standard angepasst. Sie können dementsprechend auch alle weiteren Konfigurationsdateien der Zarafa-Dienste kontrollieren und anpassen.

Bleibt abschließend noch das Aktualisieren der Webapp:

```
linux:~ # zypper up zarafa-webapp
```

Auch hier sind an der Konfiguration geringfügige Anpassungen notwendig. Zu finden ist die Konfigurationsdatei unter:

```
/etc/zarafa/webapp/config.php
```

config.php

```
...
        // Default Zarafa server to connect to.
        #define("DEFAULT_SERVER","file://\\\\.\\pipe\\zarafa");
        #define("DEFAULT_SERVER","http://localhost:236/zarafa");
        define("DEFAULT_SERVER","file:///var/run/zarafad/server.sock");
...
        // Defines the default interface language. This can be overriden by
the user.
        // This language is also used on the login page
        if (isset($_ENV['LANG']) && $_ENV['LANG']!="C"){
                define('LANG', $_ENV["LANG"]); // This means the server
environment language determines the web client language.
        }else{
                define('LANG', 'de_DE.UTF-8'); // default fallback language
        }
...
        // Defines the default time zone, change e.g. to "Europe/London"
when needed
        if(function_exists("date_default_timezone_set")) {
                if(!ini_get('date.timezone')) {
                        date_default_timezone_set('Europe/Berlin');
                }
        }
```

Auch hier sind nur die Anpassungen aufgeführt. Die wichtigste Änderung ist die Tatsache, dass sich der Pfad zum Zarafa-Socket geändert hat.

Damit sind alle Änderungen vorgenommen und Zarafa ist auf Version 7.2.2 angehoben.

Abschließend noch der Hinweis für all diejenigen, die unsere invis-Server Zarafa-Erweiterungen

nutzen. Diese Erweiterung enthält die Komponenten Zarafa-Backup und den Zarafa Lizenz Dienst. Wir werden dieses Erweiterungspack nicht mehr aktualisieren. Wer ohnehin eine Subskription erworben hat, bekommt auch die von Zarafa unterstützten Pakete. Aus diesen Paketen in der Version für SUSE Linux Enterprise 12 heraus lassen sich die RPMs für Zarafa-Backup und Lizenz-Dienst auch unter openSUSE betreiben. Wer keinen Zugriff auf „supportete“ Pakete hat kann den Lizenz-Dienst auch aus den Zarafa Community-Pakten heraus installieren. Zu finden sind die Pakete hier:

Zarafa-Pakete ohne Support

Achtung: *Nicht enthalten ist der Lizenz-Dienst in den Open-Source Paketen.*

"kleines Update" mit dem invis-updater Script

Dauer: ca 0,5h

Das kleine Update unter Verwendung des ***invis-updater*** Scripts zählt eigentlich zu den regelmäßigen Wartungsarbeiten am System. Von uns wirde das genannte Script einzig zu dem Zweck entwickelt unkritische Sicherheitsaktualisierungen automatisiert und unbeaufsichtigt durchführen zu können.

Achtung: *im hier geschilderten Testfall hat die Anwendung des* **invis-updater** *Scripts vor dem oben beschriebenen Zarafa-Upgrade zu einem nicht mehr funktionierenden Zarafa geführt. Die Ursache hierfür lag an der Art und Weise wie wir Zarafa in unseren Repositories pflegen. Wir sind gerade dabei alles so umzustellen, dass dies zukünftig nicht mehr vorkommen kann. Aber auch in diesem Fall wäre es möglich gewesen das Zarafa-Upgrade nachträglich durchzuführen.*

Die Verwendung des Scripts ist denkbar einfach:

```
linux:~ # invis-updater
```

Das Script sucht alle anstehenden Updates, streicht solche von denen ein Risiko für die Funktionalität des Systems ausgehen können und solche die einen Reboot erfordern heraus, installiert die verbleibenden, startet (soweit möglich) alle betroffenen Dienste neu und wendet das invis-eigene ***afterup*** Script an.

Ein anschließender Reboot ist nicht notwendig, kann aber helfen evtl. auftretende Probleme zu lokalisieren.

Nach einem solchen Update ist zu über prüfen, ob ein evtl. enthaltenes invisAD-setup Update neue invis-Konfigurationsdateien geliefert hat. Diese liegen in

```
/etc/invis
```

und Unterverzeichnissen. Sie sind an der Endung „.dist“ zu erkennen. Ist das der Fall, sind die neuen Dateien mit den vorhandenen zu vergleichen und Neuerungen mit Copy & Paste in die eigenen Dateien zu übernehmen.

Wurde auch das Paket z-push erneuert, so ist in der Regel in dessen Konfigurationsdatei:

```
/srv/www/htdocs/z-push2/config.php
```

wieder die korrekte Zeitzone einzutragen:

```
...
    // Defines the default time zone, change e.g. to "Europe/London" if
necessary
    define('TIMEZONE', 'Europe/Berlin');
...
```

Hinweis: *Verwenden Sie für die Zeitzone keines der bekannten Kürzel wie „MET“, „CEST“ usw. Dies kann zu Problemen führen.*

Damit ist auch dieser Schritt erledigt.

reguläres Update mit zypper

Dauer: ca. 0,5h

Bei dieser Methode des Updates werden auch neue Software-Versionen kritischer Programme installiert. Genaugenommen sind es also teilweise „Upgrades“. Dies betrifft Software, die wir aus unseren invis-Server Repositories ausrollen, aber auch die Pakete aus dem Sernet-Samba-Respositories und des VirtualBox Repos.

Ist Virtualbox installiert sollten zunächst alle virtuellen Maschinen heruntergefahren und der vboxweb-Service beendet werden. Letzteres ist je nach Version nur mittels ***kill*** möglich.

```
linux:~ # /etc/init.d/vboxweb-service stop
linux:~ # ps ax|grep vbox
```

Sollte die ***ps*** Kommandozeile einen laufenden Prozess finden ist dieser mit:

```
linux:~ # kill -9 pid-number
```

abzuschießen.

Je nachdem wann Sie Ihren invis-Server installiert haben, kann es sein, dass Ihre Virtualbox-Repository veraltet ist. Ändern Sie die Datei

```
/etc/zypp/repos.d/virtualbox.repo
```

wie folgt ab:

```
[virtualbox]
name=VirtualBox for openSUSE 13.1
baseurl=http://download.virtualbox.org/virtualbox/rpm/opensuse/13.1
type=yum
enabled=1
priority=120
autorefresh=1
gpgcheck=1
```

```
gpgkey=https://www.virtualbox.org/download/oracle_vbox.asc
keeppackages=0
```

In der falschen Datei wurde auf das Repository der Vorgängerversion openSUSE 12.3 verwiesen.

Jetzt kann mit dem regulären Update begonnen werden.

```
linux:~ # zypper ref
linux:~ # zypper up
```

Im Verlauf des Updates wird ein neuer Kernel installiert. Dabei handelt es sich um Version 3.12 aus SLES 12, welche über das openSUSE Evergreen Projekt bereit gestellt wird. Hier gibt es gelegentlich ein Problem mit UDEV-Regeln bei der Erkennung von USB-Festplatten. Das kann also ein „invis-backup“ beeinflussen. Eine Lösung dafür gibt es schon, muss ich nur wie finden 😉

Nach dem Update müssen noch einmal die VirtualBox Kernel-Module erneuert werden. Mit dem Update der VirtuaBox Version wurde auch hier der Umstieg auf „systemd“ Volzogen, d.h. es existieren keine „init-Scripts“ mehr. Zum Erneuern der Kernel-Module bringt VirtualBox allerdings ein Shell-Script mit:

```
linux:~ # /usr/lib/virtualbox/vboxdrv.sh setup
```

Ist dies erfolgreich durchgelaufen können ggf. vorhandene virtuelle Maschinen wieder gestartet werden.

Damit ist auch dieser Schritt beendet.

Upgrade auf aktuelle Versionen von Sernet-Samba, SSSD und Clamav

Dauer: ca. 45 Minuten

Im vorangegangenen Schritt hat ***zypper*** einige Pakete ausgelassen, wie die folgende Ausgabe zeigt:

```
The following 25 package updates will NOT be installed:
  clamav grub2 grub2-i386-pc grub2-x86_64-efi libudev-mini1 ntop php5-pear-
Mail python-sssd-config sernet-samba sernet-samba-ad sernet-samba-client
sernet-samba-common
  sernet-samba-libs sernet-samba-libs-32bit sernet-samba-libsmbclient0
sernet-samba-winbind sssd sssd-ad sssd-krb5 sssd-krb5-common sssd-ldap sssd-
tools systemd systemd-sysvinit
  udev-mini
```

Darunter befinden sich alle Sernet-Samba-Pakete, die aktuellen Pakete des SSSD-Dienstes sowie ein aktualisiertes Clamav Paket. Diese Pakete werden nachfolgend manuell aktualisiert.

Sernet Samba

Legen Sie zunächst ein Backup des gesamten Active-Direcoties an:

```
linux:~ # adbackup
```

Sie finden die Sicherung unter

```
/srv/shares/archiv/sicherungen/vollsicherungen/ad/
```

Die Ausgangssituation für meinen Test war ein Sernet-Samba 4.2.4, zur Verfügung stand Version 4.2.9. Mit Version 4.2.8 haben dei Sernet-Paketbauer einiges an den Paketabhängigkeiten optimiert. Probleme, gerade im Zusammenspiel mit dem SSSD können dennoch auftreten.

Entsprechend wird die Sache ab hier etwas kniffliger. Der Versuch alle Sernet-Pakete zu aktualisieren lieferte bei meinem Test eine nicht aufzulösende Abhängigkeit zu einer Bibliothek „libarchive.so.2“ die definitiv nicht zur Verfügung steht. Da sie auch unter openSUSE Leap nicht zur Verfügung steht und Sernet-Samba 4.2.9 dort ohne ein solches Problem installiert wird, habe ich beschlossen es zu ignorieren.

```
linux:~ # zypper up sernet-samba sernet-samba-ad sernet-samba-client sernet-
samba-common sernet-samba-libs sernet-samba-libs-32bit sernet-samba-
libsmbclient0 sernet-samba-winbind
```

Zur Installation aller Pakete habe ich die bemängelten Abhängigkeitsprobleme immer durch „Lösung 2“:

```
Lösung 2: sernet-samba-client-99:4.2.9-19.suse121.x86_64 beschädigen durch
Ignorieren einiger Abhängigkeiten
```

„gelöst“.

Führen Sie jetzt das Script ***afterup*** aus:

```
linux:~ # afterup
```

Sind alle Sernet-Paket aktualisiert, ist ein Neustart des Systems sehr hilfreich. Bei meinem Test war es vorher nicht möglich Auf den Fileserver zuzugreifen oder der Domäne beizutreten. Nach dem Neustart klappte alles.

SSSD

Ebenfalls Probleme aufgrund von Abhängigkeiten gab es beim SSSD. Das Update wurde beim regulären Update zurück gehalten, da es mit den Paketen „libudev-mini1-210.40“ kollidierte.

```
linux:~ # zypper up sssd
Daten des Repositories laden ...
Installierte Pakete lesen ...
Paketabhängigkeiten auflösen ...

Problem: this-is-only-for-build-envs, benötigt von libudev-
mini1-210-40.1.x86_64, wird von keinem Repository angeboten
```

```
Lösung 1: Folgende Aktionen werden ausgeführt:
  Deinstallation von libudev-mini1-208-35.1.x86_64
  Deinstallation von udev-mini-208-35.1.x86_64
Lösung 2: sssd-1.13.3-181.7.x86_64 nicht installieren
Lösung 3: sssd-1.13.3-181.7.x86_64 nicht installieren
Lösung 4: libudev-mini1-210-40.1.x86_64 beschädigen durch Ignorieren einiger
Abhängigkeiten
```

Durch Auswahl von „Lösung 1“ lies sich das Problem beheben. Da sie auch die Blockade eines „systemd“ Updates verursachte wird auch diese Update gleich mit erledigt.

Sie können sich jetzt mit

```
linux:~ # zypper ps
```

alle vom Update betroffenen Dienste anzeigen lassen oder gleich das System neu starten. Letzteres ist keine schlechte Idee, da der systemd ja bekanntlich für den Start eines System verantwortlich ist.

Nach erfolgreichem Neustart sollten Sie sich mit

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

alle Benutzer Ihres Systems anzeigen lassen. Werden dabei die Benutzerkonten aus dem Active Directory angezeigt hat alles funktioniert.

Clamav

Das Clamav beim regulären Update nicht vollständig aktualisiert wurde liegt daran, dass für openSUSE 13.1 lediglich Version 0.99 zur Verfügung steht. Das Bugfix-Release 0.99.1 stellen wir über unser invis-common Repository zur Verfügung. D.h. Das Upgrade erfordert also einen Anbieter-Wechsel.

Der Versuch Clamav zu aktualisieren liefert folgenden Hinweis:

```
linux:~ # zypper up clamav
Daten des Repositories laden ...
Installierte Pakete lesen ...
Es gibt einen Aktualisierungskandidaten für 'clamav', aber er ist von einem
anderen Anbieter. Verwenden Sie 'zypper install clamav-0.99.1-139.4.x86_64'
um diesen Kandidaten zu installieren.
Paketabhängigkeiten auflösen ...

Keine auszuführenden Aktionen.
```

Entsprechend dem Hinweis sieht dann das Upgrade aus:

```
linux:~ # zypper in clamav-0.99.1-139.4.x86_64
```

Danach kann der der Clam-Daemon neu gestartet werden:

```
linux:~ # systemctl restart clamd.service
linux:~ # systemctl restart freshclam.service
```

Damit ist auch dieser Schritt abgeschlossen und Ihr System auf Basis von openSUSE 13.1 auf dem neuesten Stand.

Distribution Upgrade openSUSE 13.1 auf openSUSE leap 42.1

Dauer: ca. 4h (wenn keine Katastrophen eintreten)

An dieser Stelle möchte ich noch einmal an die eingangs erwähnte Datensicherung oder das beiseite Legen einer Festplatte aus einem RAID Verbund erinnern. Ab hier wird es riskant! Erhöht wird das Risiko noch einmal dadurch, dass wir mit openSUSE 13.2 eine Release überspringen, die wir für invis-Server nie unterstützt haben. openSUSE selbst erwähnt in den Anleitungen des Projekts immer, dass nur das Upgrade von einer Release zur nächsten unterstützt wird. Hoffen wir also das Beste!

Voraussetzung für alles Weitere ist auf jeden Fall, dass alle zuvor beschriebenen Schritte erfolgreich beendet wurden und Ihr System in einem funktionierenden Zustand ist.

Vorbereitung

Im ersten Schritt sind die genutzten Repositories auf den Stand der Ziel-Release umzustellen. Dazu lassen wir uns zunächst einmal alle vorhandenen Repositories anzeigen:

```
linux:~ # zypper lr
#  | Alias                              | Name
| Aktiviert | Aktualisieren
---+------------------------------------+-----------------------------------
-----------------------------------+-----------+-------------
 1 | download.opensuse.org-13.1-non-oss | Aktualisierungs-Repository (Nicht-
Open-Source-Software)               | Ja        | Ja
 2 | download.opensuse.org-non-oss      | Haupt-Repository (NON-OSS)
| Ja        | Ja
 3 | download.opensuse.org-oss          | Haupt-Repository (OSS)
| Ja        | Ja
 4 | download.opensuse.org-update       | Hauptaktualisierungs-Repository
| Ja        | Ja
 5 | openSUSE-13.1-1.10                 | openSUSE-13.1-1.10
| Ja        | Ja
 6 | repo-debug                         | openSUSE-13.1-Debug
| Nein      | Ja
 7 | repo-debug-update                  | openSUSE-13.1-Update-Debug
| Nein      | Ja
 8 | repo-debug-update-non-oss          | openSUSE-13.1-Update-Debug-Non-Oss
| Nein      | Ja
 9 | repo-source                        | openSUSE-13.1-Source
| Nein      | Ja
10 | sernet-samba-4.2                   | SerNet Samba 4.2 Packages
```

```
(suse-13.1)                                  | Ja        | Nein
11 | spins_invis_common                 | Common packages for invis-Server
stable & unstable (openSUSE_13.1)    | Ja        | Ja
12 | spins_invis_stable                 | Production Project for the
openSUSE invis-Server Spin (openSUSE_13.1) | Ja        | Nein
13 | virtualbox                         | VirtualBox for openSUSE 13.1
| Ja        | Ja
```

Darunter befinden sich mit den Repositories 10 (Sernet Samba) und 13 (VirtualBox) zwei Fremd-Repositories und mit 11 & 12 die Repositories des invis-Server Projektes. Alle Anderen sind Standard-Repos von openSUSE.

Erstellen wir zunächst eine Sicherheitskopie des Repo-Verzeichnisses:

```
linux:~ # cp -R /etc/zypp/repos.d /etc/zypp/repos.d.bak
```

VirtualBox stellt ein fertiges Repository für openSUSE Leap (und 13.2 in Kombination) zur Verfügung, d.h. es ist am einfachsten das vorhandene zu löschen und das neue hinzuzufügen:

```
linux:~ # zypper rr 13
Repository 'VirtualBox for openSUSE 13.1' wird entfernt
.....................................................................
.....................................[fertig]
Repository 'VirtualBox for openSUSE 13.1' wurde entfernt.
linux:~ # zypper ar
http://download.virtualbox.org/virtualbox/rpm/opensuse/13.2/virtualbox.repo
Repository 'VirtualBox for openSUSE 13.2 / Leap 42.1' wird hinzugefügt
.....................................................................
.........................[fertig]
Repository 'VirtualBox for openSUSE 13.2 / Leap 42.1' erfolgreich
hinzugefügt
Aktiviert: Ja
Autoaktualisierung: Nein
GPG-Überprüfung: Ja
URI: http://download.virtualbox.org/virtualbox/rpm/opensuse/13.2
```

In Bezug auf Sernet-Samba haben wir die Qual der Wahl. Es wird kein Repository explizit für openSUSE Leap 42.1 zur Verfügung gestellt, allerdings je eines für SLES12 und eines für openSUSE 13.2. openSUSE Leap 42.1 hegt verwandtschaftliche Beziehungen zu beiden.

Bei Neuinstallationen verwenden wir das Repository für SLES12, also sollte das auch hier keine falsche Entscheidung sein. Da hier in der Repository-Datei Zugangsdaten für das Repository enthalten sind ist es einfacher die Datei selbst zu editieren, statt sie zu ersetzen. Öffnen Sie dazu

```
/etc/zypp/repos.d/sernet-samba4.repo
```

und passen Sie die Pfade und Bezeichnungen wie folgt an:

```
[sernet-samba-4.2]
name=SerNet Samba 4.2 Packages (SLES 12)
type=rpm-md
```

```
baseurl=https://sernet-samba-public:Noo1oxe4zo@download.sernet.de/packages/samba/4.2/sles/12/
gpgcheck=1
gpgkey=https://sernet-samba-public:Noo1oxe4zo@download.sernet.de/packages/samba/4.2/sles/12/repodata/repomd.xml.key
enabled=1
```

Alle weiteren Repositories bearbeiten wir bezüglich der Versionsnummer in einem Rutsch unter Verwendung von ***sed***

```
linux:~ # sed -i 's/13\.1/42\.1/g' /etc/zypp/repos.d/*
```

Leider haben sich mit openSUSE leap auch die Pfade geändert. In allen openSUSE Repositories ist in der Zeile „baseurl“ nach „distribution“ oder „update“ und vor der Versionsnummer das Unterverzeichnis „leap“ einzufügen. Auch hier hilft ***sed***:

```
linux:~ # sed -i 's%distribution\/%distribution\/leap\/%g'
/etc/zypp/repos.d/*
linux:~ # sed -i 's%update\/%update\/leap\/%g' /etc/zypp/repos.d/*
```

In den beiden Update-Repositories ergeben sich noch ein paar händisch zu beseitigende Ungereimtheiten. Ändern Sie:

```
download.opensuse.org-13.1-non-oss.repo
```

```
[download.opensuse.org-42.1-non-oss]
name=Aktualisierungs-Repository (Nicht-Open-Source-Software)
enabled=1
autorefresh=1
baseurl=http://download.opensuse.org/update/leap/42.1/non-oss/
path=/
type=rpm-md
keeppackages=0
```

Hierin muss in der „baseurl“ Zeile „42.1-non-oss“ in „42.1/non-oss“ geändert werden.

```
download.opensuse.org-update.repo
```

```
[download.opensuse.org-update]
name=Hauptaktualisierungs-Repository
enabled=1
autorefresh=1
baseurl=http://download.opensuse.org/update/leap/42.1/oss
path=/
type=rpm-md
keeppackages=0
```

Hierin musste am Ende der URL das Unterverzeichnis „/oss“ angefügt werden.

Damit sind alle Repositories angepasst.

Upgrade

Damit sind wir am „point of no return“ angelangt. Bevor es los geht noch eine wichtige Warnung:

Achtung: *Sollten Sie Ihren invis-Server unter Verwendung einer grafischen Oberfläche betreiben, sollten Sie das nachfolgende Upgrade auf KEINEN Fall in Runlevel 5 durchführen, sondern auf RL3 in einer Konsole! …auch von einem Upgrade via SSH würde ich absehen.*

Die Durchführung des Upgrades ist relativ simpel. Bereinigen Sie zunächst die Repository-Caches und frischen Sie sie auf:

```
linux:~ # zypper clean
linux:~ # zypper ref
```

Sollten dabei Fehler auftreten, kontrollieren Sie noch einmal die Pfadangaben in den Repository-Dateien.

Das eigentliche Upgrade erfolgt dann mit:

```
linux:~ # zypper dup
```

Das Ganze beginnt mit Abhängigkeitsproblemen, was nicht weiter verwunderlich ist. Verschiedene Sernet-Samba-Pakete sind abhängig von einem Paket namens „insserv-compat“:

Konflikte

```
Problem: sernet-samba-99:4.2.9-19.suse132.x86_64 benötigt /lib/lsb/init-
functions, was aber nicht angeboten werden kann
  Gelöschte Anbieter: insserv-compat-0.1-8.1.2.x86_64
Lösung 1: veraltetes insserv-compat-0.1-8.1.2.x86_64 behalten
Lösung 2: Deinstallation von sernet-samba-99:4.2.9-19.suse121.x86_64
Lösung 3: sernet-samba-99:4.2.9-19.suse132.x86_64 beschädigen durch
Ignorieren einiger Abhängigkeiten

Wählen Sie aus den obigen Lösungen mittels Nummer oder brechen Sie a(b).
[1/2/3/b] (b): 3
```

Das passiert genau drei mal. Ich habe mich jeweils für Lösung 3 entschieden und scheine damit richtig gelegen zu haben, da im Laufe der Installation ein neueres „insserv-compat“ Paket installiert wurde.

Aufmerksam haben mich auch einige der angezeigten Vendor-Changes gemacht:

Vendor changes

```
The following 10 packages are going to change vendor:
  grub2             openSUSE -> obs://build.opensuse.org/spins:invis
  grub2-i386-pc     openSUSE -> obs://build.opensuse.org/spins:invis
  grub2-x86_64-efi  openSUSE -> obs://build.opensuse.org/spins:invis
  ipcalc            obs://build.opensuse.org/spins:invis -> openSUSE
  libgsoap-2_8      obs://build.opensuse.org/spins:invis -> openSUSE
```

```
  liblzma5-32bit    obs://build.opensuse.org/spins:invis -> openSUSE
  ntop              openSUSE -> obs://build.opensuse.org/spins:invis
  php5-pear-Mail    obs://build.opensuse.org/spins:invis -> openSUSE
  postfix           obs://build.opensuse.org/spins:invis -> openSUSE
  postfix-mysql     obs://build.opensuse.org/spins:invis -> openSUSE
```

Mir war vorher gar nicht bewusst, dass wir Grub im eigenen Repository pflegen. Ich muss wohl mal im Team fragen warum das so ist. Im Moment kann man das nur so hinnehmen und alles mit „y“ akzeptieren.

Gegen Ende des Upgrades traten die bekannten Fehler mit VirtualBox-Upgrades auf. Das aktuelle Paket brach bei der Installation ab. Gelöst habe ich es in dem ich nach ein paar Minuten auf „r“ für Retry gedrückt habe. Danach wurde das Paket ohne Probleme installiert.

Das Upgrade lief etwa eine Stunde. Obwohl ich vermutet hatte, dass es mit Grub Probleme geben würde, habe ich mich zu einem direkten Reboot entschieden und mein System startete nicht mehr.

Um das zu beheben habe ich zur „Super-Grub-Disk“ gegriffen **(und das ist an einem 1. April ein ECHTER Spaß)** und mein System gestartet, was ebenfalls problemlos funktionierte.

Zum Beheben der Probleme genügte es Grub neu in die MBRs aller Festplatten zu installieren:

```
linux:~ # grub2-install /dev/sda
linux:~ # grub2-install /dev/sdb
```

Der nächste Reboot funktioniert dann ohne Super-Grub-Disk (hätte man auch vorher erledigen können), allerdings ohne unser invis-Grub-Theme. Das wiederherzustellen ist aber allenfalls Kosmetik und fällt in die Rubrik Nacharbeit.

…und Nacharbeit gibt es, wäre ja auch ein bisschen zuviel des Guten, wenn ein System nach Neuinstallation von mehr als 1300 Paketen Problemlos funktioniert.

Nacharbeit

Nach dem Neustart laufen einige (von einander mehr oder minder abhängige) Dienste nicht. Bevor wie dies beheben werfen wir einen Blick auf die Konfigurationsdateien des invis-Servers selbst. Darin haben sich auch ein paar Dinge geändert.

invis-Server Konfiguration

Im Verzeichnis

```
/etc/invis
```

ist nach dem Upgrade eine auf „.rpmnew“ endende Datei zu finden. Sie enthält einige Konfigurationen der jetzt aktualisierten invis-Version die in der alten Datei noch nicht vorhanden waren. Diese sind händisch in die genutzte Datei „invis.conf“ zu übertragen.

Hier die entsprechenden Einträge, die Sie beim Übernehmen natürlich an Ihre Installation anpassen müssen:

```
...
# DNS Infos
domain:invis-net.loc
revDomain:220.168.192.in-addr.arpa
...
# Wenn einzelne Zarafa-Dienste nicht benötigt werden, koennen diese aus dem
nachfolgenden Array entfernt werden.
zServices:licensed server search spooler dagent gateway ical monitor
```

Hinweis: *Der Eintrag hinter zServices ist in Klammern gesetzt. Das ist ein Bug, die Klammern müssen entfernt werden. Statt dessen müssen Sie im Script „runzarafa“ hinzugefügt werden:*

Verändern Sie Zeile 31 im Script

```
/usr/bin/runzarafa
```

wie folgt:

```
services=(`getconfdata $conffile "zServices" "2"`)
```

Samba

Allem voran Sernet-Samba. Problem Nr. 1 wird dadurch verursacht, dass Samba nicht auf sein Sicherheitszertifikat zugreifen kann. Ursache hierfür ist, dass während des Upgrades ein neues Verzeichnis

```
/etc/ssl/certs
```

angelegt wurde. Darin befanden sich die alten Zertifikate. Da während des Upgrades allerdings eine Kopie des alten Verzeichnisses gesichert wurde ist das schon mal kein großes Problem. Einfach die Zertifikate aus der Sicherung wieder an den richtigen Ort kopieren:

```
linux:~ # cp /etc/ssl/certs.rpmsave/ldap-cert.pem /etc/ssl/certs
linux:~ # cp /etc/ssl/certs.rpmsave/mail-cert.pem /etc/ssl/certs
```

Aber auch danach funktionierte der Samba-Neustart zunächst nicht. Ich konnte dafür zwei Ursachen ausmachen:

1. Ein Problem bei Nutzung auf die Libraries, die für den Zugriff auf LDB-Dateien notwendig sind. Hier bringt Sernet-Samba eigene Libraries mit.
2. Unter openSUSE Leap läuft üblicherweise Apparmor. Dieses Problem ist hier im Wiki bereits als Bug beschrieben.

Zur Lösung von Problem ein habe ich ohne viel herumzuprobieren einfach alle Sernet-Pakete noch einmal installiert. Problem Nummer zwei wird einfach mit der Deaktivierung von Apparmor behoben.

```
linux:~ # chkconfig -d boot.apparmor
linux:~ # rcapparmor stop
```

Hinweis: *ich habe hier bewusst auf die klassischen Befehle zurückgegriffen, da in diesem Fall das klassische init-Script des Apparmor Dienstes erhalten geblieben ist und „systemctl“ das scheinbar nicht bemerkt.*

Danach Samba starten:

```
linux:~ # systemctl start sernet-samba-ad.service
```

..und überprüfen:

```
linux:~ # systemctl status sernet-samba-ad.service
```

DNS

Zweites Problem, der DNS-Server startete nicht. Auch das ist ein alt bekanntes Problem, **bind** hat einfach keinen Zugriff auf auf das Verzeichnis:

```
/var/lib/samba/private
```

Das erledigt unser ***afterup*** Script:

```
linux:~ # afterup
```

Dann den Nameserver starten:

```
linux:~ # systemctl start named.service
```

und natürlich überprüfen.

DHCP

Da Samba nicht lief, konnte auch der DHCP-Server nicht starten. Das lässt sich jetzt nachholen:

```
linux:~ # systemctl start dhcpd.service
```

und natürlich überprüfen.

Webserver

Auch Apache verweigert nach dem Upgrade den Start. Eine Status-Abfrage des Dienstes liefert das Problem. Der folgende Fehler:

```
Invalid command 'Order', perhaps misspelled or defined by a module not
included in the server configuration
```

wurde gleich mehreren Konfigurationsdateien festgestellt:

```
...
AH00526: Syntax error on line 8 of /etc/apache2/conf.d/zarafa-webaccess-
mobile.conf
...
AH00526: Syntax error on line 8 of /etc/apache2/conf.d/zarafa-webaccess.conf
...
AH00526: Syntax error on line 29 of /etc/apache2/vhosts.d/invis-sslvh.conf
...
AH00526: Syntax error on line 22 of /etc/apache2/vhosts.d/invis-vh.conf
...
AH00526: Syntax error on line 22 of /etc/apache2/vhosts.d/z-push_vh.conf
```

Es haben sich eine Konfigurationsdateien eingeschlichen, die noch nicht mit Apache 2.4 kompatibel ist. Übeltäter ist zum einen der Zarafa-Webaccess. Ich neige dazu auf das etwas antiquierte Stück Software zugunsten der modernen Zarafa-Webapp zu verzichten und zum anderen drei Konfigurationsdateien des invis-Servers selbst.

Beheben wir zunächst das Problem der Webacces-Dateien händisch. Da beide Dateien im Prinzip identisch sind hier nur ein Beispiel.

Aus:

```
Order Allow,Deny
Allow from all
```

wird:

```
Require all granted
```

Für die Dateien des invis-Servers empfiehlt es sich diese aus den mitgelieferten Beispieldateien neu zu erstellen. Die Beispiele sind unter

```
/usr/share/doc/packages/invisAD-setup/examples/webserver
```

zu finden.

Erstellen Sie von der alten Datei eine Sicherheitskopie und entnehmen Sie ihr bitte folgende Informationen:

- Den Port des SSL-vHosts
- Den Server-Namen (DDNS-Name)
- Den lokalen Server-Namen

und tragen Sie die Informationen in die neue Datei aus dem Beispielverzeichnis ein:

```
....
```

```
<VirtualHost *:httpsport>
    DocumentRoot "/srv/www/htdocs/portal"
    ServerName your.ddns-domain.net
...

# Deeplinks verhindern
    SetEnvIfNoCase Referer "^http://invis.invis-net.loc" dontblock
    SetEnvIfNoCase Referer "^https://your.ddns-domain.net" dontblock
...
```

Das Ergebnis könnte so aussehen:

```
....
<VirtualHost *:53532>
    DocumentRoot "/srv/www/htdocs/portal"
    ServerName meinefirma.dyndns.net
...

# Deeplinks verhindern
    SetEnvIfNoCase Referer "^http://invis.firma-net.loc" dontblock
    SetEnvIfNoCase Referer "^https://meinefirma.dyndns.net" dontblock
...
```

Kopieren Sie die neue Datei nach:

```
/etc/apache2/vHosts.d
```

Verfahren Sie mit den weiteren Dateien auf die gleiche Weise. In der vHost Konfiguration des z-Push Dienstes hat sich noch ein Bug eingeschlichen, den Sie jetzt gleich mit beheben können. Es wurden darin für Access- und Erro-Log die gleiche Datei angegeben. Benennen Sie die Datei für das Zugriffsprotokoll einfach um:

```
    ServerName meinefirma.dyndns.net
    SSLEngine On
    ErrorLog /var/log/apache2/z-push-error.log
    CustomLog /var/log/apache2/z-push-access.log common
```

In dieser und der Datei des Standard-vHosts ist jeweils nur der „ServerName“ ist lediglich anzupassen. Achten Sie darauf, dass bei Standard-vHost (invis-vh.conf) als ServerName der interne Hostname Ihres invis-Servers angegeben werden muss.

Ist alles erledigt, steht auch dem Start des Apache-Webservers nichts mehr im Wege.

Zarafa

Für Zarafa ergibt sich ebenfalls ein kleines Problem. Während des Upgrades wurden neue Konfigurationsdateien in

```
/etc/zarafa
```

abgelegt, auf die Zarafa unter der Benutzerkennung „zarafa“ bzw. der Gruppe „zarafa“ nicht zugreifen kann. Dies lässt sich beheben mit:

```
linux:~ # chown -R .zarafa /etc/zarafa
linux:~ # runzarafa stop
linux:~ # runzarafa start
```

Kosmetik

Färben wir zum Schluss noch Grub wieder in invis-Orange ein. Dazu ist zunächst das invis-Theme zu installieren:

```
linux:~ # tar -xf /usr/share/doc/packages/invisAD-
setup/examples/grub/invis8.tar.gz -C /boot/grub2/themes/
```

Jetzt noch die Konfigurationsdatei platzieren:

```
linux:~ # cp /usr/share/doc/packages/invisAD-setup/examples/grub/grub
/etc/default/
```

Abschließend muss eine Grub-Bootloader Konfiguration erzeugt werden:

```
linux:~ # grub2-mkconfig -o /boot/grub2/grub.cfg
```

Ein Reboot sollte jetzt die Korrekte invis-Version vor orangem Hintergrund zeigen.

Wer möchte kann jetzt noch die Konfigurationsdateien wichtiger Dienste wie z.B. Postfix mit den neuen Vorlagen in

```
/usr/share/doc/packages/invisAD-setup/examples/
```

abgleichen und ein letztes reguläres Update laufen lassen.

Ansonsten gilt: **Wilkommen zu Ihrem aktuellen invis-Server**

invisAD 10.4 -> invisAD 10.5

Mit Version 10.5 des invis-Servers ergeben sich ein paar strukturelle Unterschiede. Allem voran geht dabei die Umstellung der „Public Key Infrastructure“, also die Organisation von Schlüsseln und Zertifikaten für die verschiedenen Verschlüsselungsaufgaben.

Bisher wurden zwei sogenannte Zertifizierungsstellen (CA) erstellt, davon diente eine zur Verwaltung der Schlüssel und Zertifikate für die verschiedenen Serverdienste (LDAP, Mail, Web) und eine zweite ausschließlich für openVPN. Zur Pflege der Schlüssel- und Zertifikate wurde das Toolkit „easy-rsa“ in Version 2.0 verwendet, der Rest direkt mit ***openssl*** und eigenen Scripten.

Das Toolkit „easy-rsa“ liegt inzwischen in Version 3.0 vor und wurde im Zuge der Weiterentwicklung deutlich verbessert. Nach ersten Tests damit war klar, dass easy-rsa 3.0 ideal zur Verwaltung aller Schlüssel- und Zertifikaten eines invis-Servers ist. Kurzum ab Version 10.5 verfügen invis-Server nur noch über eine mit easy-rsa erstellte Zertifizierungsstelle.

Aus dem bisherigen Script ***serverkeys*** zur Generierung von Serverzertifikaten wurde ***inviscerts***. Die Handhabung des Scripts wird im Abschnitt „invis Administration“ erläutert.

Beim Upgrade von 10.4 auf 10.5 müssen die Schritte zum Aufbau einer PKI die ***sine*** während des Setups erledigt manuell vorgenommen werden.

Aufbau einer Public Key Infrastruktur mit easy-rsa

Unter anderem zur Nutzung im invis-Server wurde ein easy-rsa 3.x RPM im Repository „spins:invis:common“ gebaut. Die nachfolgende Anleitung beschreibt, wie mit easy-rsa eine PKI manuell aufgebaut wird. Die spätere Verwaltung von Zertifikaten übernimmt auf invis-Servern das Script ***inviscerts***.

Ab Version 3.0 stellt easy-rsa nur noch ein einziges Script ***easyrsa*** für alle Aufgaben zur Verfügung. Sämtliche Konfigurationsdateien liegen in:

```
/etc/easy-rsa
```

Vor dem Aufbau einer PKI muss im genannten Verzeichnis die Datei „vars“ an die eigenen Bedürfnisse angepasst werden. Es ist im Unterschied zu älteren easy-rsa Versionen nicht mehr notwendig die Variablen in dieser Datei manuell mittels ***source*** in die aktuelle Shell Umgebung zu laden. Dies erledigt ***easyrsa*** selbsttätig.

Vorbereitung

Angepasst werden müssen zumindest folgende Zeilen:

Name der PKI:
(Zeile 65)

```
...
set_var EASYRSA_PKI             "$EASYRSA/fsp-net.loc"
...
```

PKI individualisieren:
(Ab Zeile 84)

```
...
set_var EASYRSA_REQ_COUNTRY     "DE"
set_var EASYRSA_REQ_PROVINCE    "Hessen"
set_var EASYRSA_REQ_CITY        "Schotten"
set_var EASYRSA_REQ_ORG         "FSP Computer und Netzwerke"
set_var EASYRSA_REQ_EMAIL       "stefan@invis-server.org"
```

```
set_var EASYRSA_REQ_OU          "invis-Server.org"
...
```

Die vorgegebene Länge der DH-Parameter ist auf 2048 Bit eingestellt, dies reicht nach gegenwärtigem Stand der Technik aus. Eine Erhöhung auf 4096 Bits verlängert den Bau der DH-Datei immens (Die Rede ist hier von Stunden).

Voreingestellt sind Gültigkeitsdauern für CA und damit signierte Zertifikate von 10 Jahren. Kann man lassen, die Lebensdauer der Zertifikate im Vergleich zur CA zu verkürzen ist auch OK.

Jetzt kann die PKI vorbereitet werden:

```
linux:~ # easyrsa init-pki
```

Damit wird unter „/etc/easy-rsa“ eine Verzeichnisstruktur für die neue PKI angelegt und die Vorbereitungen sind abgeschlossen.

PKI erzeugen

Benötigt werden folgende Komponenten:

1. Eine CA zum signieren von Server und Client Zertifikaten
2. Eine Diffie-Hellman Parameter Datei für sicheren Schlüsselaustausch
3. Eine CRL Datei um Zertifikate zurückzuziehen

CA erstellen:

Auch hier genügt ein einziger Befehl:

```
linux:~ # easyrsa build-ca
```

Beim Bau der CA wird ein Passwort erfragt, dieses Passwort wird später beim signieren von Zertifikaten benötigt. Dieses Passwort darf nicht in falsche Hände gelangen.

Diffie-Hellman Parameter erstellen:

```
linux:~ easyrsa gen-dh
```

CRL erzeugen

```
linux:~ easyrsa gen-crl
```

Hierfür wird wieder das CA-Passwort benötigt.

invis-Classic -> invis-AD

Aufgrund der immensen Unterschiede zwischen beiden Systemen gibt es hier schlicht keinen

Migrationsweg. Vielmehr geht es darum Daten und Informationen aus einem klassischen invis-Server auf einen invisAD zu migrieren.

Im Idealfall wird ein neuer invisAD auf neuer Hardware aufgebaut. Möglich aber um Längen aufwändiger ist die Neuinstallation auf gleicher Hardware. In letzterem Fall müssen zuvor alle relevanten Daten der alten Installation gesichert werden.

Zu den zu sichernden Daten zählen:

- Das vollständige LDAP Verzeichnis in Form eines LDIF-Dumps
- Das gesamte „/etc“ Verzeichnis
- Alle Datenbanken inklusive der Dokuwiki Daten
- Alle Nutzdaten des Fileservers
- Den gesamten Mailbestand.

Alleine für die Sicherung müssen Sie je nach Datenmenge mit vielen Stunden Arbeit und vor allem Wartezeit rechnen. Da ist es eher ratsam sich für die Neuinstallation wenigstens neue Festplatten zu gönnen und die Platten des alten Servers für die Dauer die Migration in einen anderen PC einzubauen.

Wie auch immer, aus eigener Erfahrung empfehle ich **immer** komplett neue Hardware für die Neuinstallation.

Es gilt vorab eine weitere grundsätzliche Entscheidung zu fällen. Der Aufwand LDAP-Informationen aus einem OpenLDAP Dump zu extrahieren und für den Import in ein AD umzubauen ist mühsam und fehleranfällig. Wenn es sich beim zu migrierenden invis-Server um eine Installation mit nur wenigen Benutzern und überschaubarem Umfang an IP-Geräten im Netz handelt, ist es einfacher und schneller die Daten in der Neuinstallation über das invis-Portal neu einzugeben. Die Grenze würde ich bei ca. 20 Benutzern ziehen.

Nachfolgend Anleitungen und Tipps zur Bewältigung der Migration. Eine genaue Anleitung wie die oben beschriebene invisAD → invisAD Migration wird es hier nicht geben.

LDAP Informationen migrieren

Zu den LDAP-Informationen zählen Benutzerkonten, deren zugehörige Email-Konten Informationen und die Daten von DNS- und DHCP-Server.

Benutzerkonten

Beginnen wir mit einer ernüchternden Information. **Lassen Sie es sein!**

Samba bietet mit der Funktion **classic-upgrade** eine Möglichkeit eine Windows-Domäne auf Basis von Samba3 (NTLM Domäne) unabhängig vom verwendeten Backend (ldap, tdbsam usw.) in eine AD-Domäne zu migrieren. Diesen Weg habe ich bereits einmal erfolgreich beschritten. Ziel der damaligen Migration war allerdings kein invis-Server sondern ein alleinstehender Active-Directory Domain-Controller.

invis-Server nutzen OpenLDAP (Classic) bzw. den LDAP-Dienst eines Active-Directories für weit mehr als nur die Speicherung von Benutzerinformationen. Einerseits erweitern wir das AD mit zahlreichen

Schema-Erweiterungen, diese müssten nach einem Classic-Upgrade alle händisch eingepflegt werden. Andererseits muss man sich der Tatsache bewusst sein, dass die Benutzerverwaltung mit AD gänzlich anders aufgebaut ist als auf dem klassichen Server. Ein invis-Classic legt grundsätzlich POSIX-kompatible Benutzerkonten an, die um Windows-Attribute erweitert wurden. Ein invisAD hingegen legt Windows-Benutzerkonten an, die um POSIX-Attribute erweitert werden. Auf einem invisAD Server nutzen wir gänzlich andere UID- und GID-Bereiche als auf dem invis-Classic. Daraus können sich nach einer solchen Migration Probleme im laufenden Betrieb ergeben.

Wer dennoch diesen Weg beschreiten möchte findet im Samba Wiki eine Anleitung für den Upgrade-Prozess.

Vorbereitend müssen Sie dennoch eine volle invisAD-Installation durchführen. Stoppen Sie vor dem Classic-Upgrade den Samba-Dienst, sichern Sie die Samba-Konfiguration unter

```
/etc/samba
```

und erstellen Sie ein Backup des neuen leeren ADs:

```
linux:~ # adbackup
```

Ist die Migration per Classic-Upgrade gelungen, müssen Sie alle Daten- und Schema-Erweiterungen händisch ins neue AD integrieren. Auch das habe ich bereits einmal durchgeführt, die entsprechende Anleitung folgt.

Früher oder später wird der invisAD-Server auch ein Script zum massenhaften anlegen neuer Benutzerkonten mitbringen, was dann auch eine gewisse Erleichterung darstellt.

Der Vorteil dieser mühsamen Methode sollte allerdings nicht unerwähnt bleiben. Die Benutzerprofile der Domänenbenutzer funktionieren auch nach der Migration noch, da sich die Dmomain-SID hierbei nicht ändert. Eine vollständige Neuinstallation bedeutet neue Benutzer-Konten auch wenn sich Benutzernamen und Passwörter nicht ändern. Das bedeutet, dass alle Client-PCs der Domäne neu beitreten und alle Benutzerprofile neu aufgebaut werden müssen. Ein wenig googlen fördert zwar auch Tools zur Profilmigration zu Tage, allerdings kann ein wenig „Tabula Rasa“ manchmal nicht schaden.

… to be continued

Konfigurationen migrieren

Datenbanken migrieren

Email Bestand migrieren

Fileserver-Datenbestand migrieren