Die invis Toolbox

Die invis-Server Toolbox besteht aus einer Reihe von Shell-Scripts die der Installation, Administration und Wartung des invis-Servers dienen.

Diese Seite widmet sich der Dokumentation der einzelnen Scripts in alphabetischer Reihenfolge.

Hinweis:*Tools, die wir im Laufe der Entwicklung des invis-Servers entfernt haben, finden Sie* ***hier***

invis-server - (Ab Version 13.0)

Vorab und ohne auf die alphabetische Reihenfolge zu achten möchten wir auf das kleine Helfer-Script ***invis-server*** hinweisen. Offiziell ist es seit Version 13 des invis-Servers enthalten. Es generiert eine Liste aller Scripts der invis-Toolbox, nebst kurzer Beschreibung dessen wozu das jeweilige Script gedacht ist.

Die Anwendung ist denkbar einfach. Rufen Sie es einfach mit einer der folgenden Option auf:

- **all** - Zeigt alle Script der Toolbox, unabhängig von deren Zugehörigkeit zu einer der folgenden Kategorien.
- **server** - Zeigt alle Scripts die zur Server-Administration gedacht sind.
- **portal** - Zeigt alle Scripts die zum invis-Portal gehören.
- **apps** - Zeigt Scripts die zu installierten Applikationen gehören.
- **setup** - Zeigt alle Scripts die zum Setup des Servers gehören.

```
invis:~ # invis-server server
```

actdehydrated (Ab Version invis 12.1)

Dieses Script kann den invis-Server auf die Nutzung von Sicherheitszertifikaten von „Let's Encrypt“ umschalten und Let's Encrypt Zertifikate erstmalig generieren.

Voraussetzung dafür ist, dass der Server via Internet auf Port 80 (HTTP) erreichbar ist und er über einen im Internet gültigen Namen verfügt.

Zu Hintergründen und Anwendung lesen Sie das Kapitel **Let's Encrypt** im Abschnitt invis Administration.

Hinweis: *Das Script muss nur ein einziges Mal ausgeführt werden.*

adbackup

Ein kleines Script zur Sicherung des gesamten Active Directories. Die Sicherung erfolgt nach:

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

Eine Sicherung des ADs ist immer dann Sinnvoll, wenn Sie planen manuelle Veränderungen daran durchzuführen.

Da das Script einen Sekunden-genauen Zeitstempel in den Namen des Sicherungsarchivs einpflegt können beliebig viele Sicherungen in kurzen Abständen aufeinanderfolgend angelegt werden.

Hinweis: *Idealerweise sollte das Script bei deaktiviertem Samba-AD Domain Controller ausgeführt werden, da ansonsten die Gefahr besteht, dass sich Teile des ADs während der Sicherung ändern und somit Inkonsistenzen entstehen können. Das Script warnt allerdings auch wenn dies geschieht und gibt Ihnen somit die Möglichkeit das Backup zu wiederholen.*

Seit invis-Version 13.5 wird dieses Script zeitgesteuert jede Nacht einmal ausgeführt.

Ausgeführt wird es ohne weitere Aufrupargumente:

```
invis:~ # adbackup
```

addposixattrs (Ab invis-Server 14.2)

Das Tool ist dazu gedacht ein bestehendes Benutzerkonto um POSIX-Attribute zu erweitern, damit sich der betroffene Benutzer auch direkt an der Konsole des Servers anmelden kann. Bestes Beispiel dafür ist das Konto des Domänenadministrators. Es wird bereits beim Domain-Provisioning angelegt. Es verfügt weder über ein Home-Verzeichnis noch wurde ein Logon-Script hinterlegt und er wird vom zugrunde liegenden Linux gar nicht gesehen. Das lässt sich mit ***addposixattrs*** ändern. Die Verwendung ist einfach:

```
invis:~ # addposixattrs benutzername
```

Wird es auf den Benutzer „administrator“ angewendet, wird diesem das Logon-Script „admin.cmd“ zugeordnet, andere Benutzer das Script „user.cmd“. Der Unterschied ist, dass dem Admin auch die Freigabe „Service“ als Laufwerk gemappt wird.

afterup

Mit Einführung des RPM-basierten Setups (invis server 9.2) wurde ***afterup*** auf ein Minimum an Funktionen reduziert.

Es ist immer wieder ärgerlich, wenn ein Online-Update eigene Konfigurationen überbügelt. Das Script ***afterup*** ist dazu gedacht solche Änderungen rückgängig zu machen. Das funktioniert natürlich nur bei Veränderungen, die immer wieder vorkommen. Derzeit behebt das Script lediglich ein Ärgernis:

1. Ein Update der Samba-Pakete verändert die Besitzrechte auf das Profil- sowie das Druckertreiberverzeichnis. In der Folge werden beim Anmelden am Client die Windows-Profile nicht mehr geladen und auch das Hochladen von Druckertreibern auf den Samba-Server ist nicht mehr möglich.

Ob weitere Funktionen notwendig sind wird sich wohl über kurz oder lang herausstellen.

afterup behebt diese Probleme, achtet aber dabei nicht auf Änderungen von Ihnen, die von der invis-Standard-Installation abweichen.

alldump

Das Tool ***alldump*** sichert alle Datenbanken des invis Servers. Dazu gehören alle in MySQL und PostgreSQL angelegten Datenbanken sowie das gesamte LDAP-Verzeichnis.

Die Dumps werden in /srv/shares/archiv/datenbanksicherungen abgelegt. Das Tool wird (ab Version 6.7 R1) üblicherweise über einen Cronjob (siehe /etc/cron.d/invis.cron) jeden Freitag gestartet. Es kann natürlich auch jederzeit von Hand auf der Kommandozeile genutzt werden.

aschenputtel

Der Name ist Programm „***aschenputtel***“ ist eines der Helper-Scripts für das invis-Portal. Es sorgt dafür, dass per Transfer-Seite im Portal auf den Server hochgeladene Dateien zunächst auf Virenbefall untersucht werden, bevor Sie auf dem Fileserver zur Verfügung stehen.

Von Viren befallene Dateien werden in das Quarantäne-Verzeichnis unter /var/spool/infected verschoben und auf .vir umbenannt. Alle sauberen Dateien landen in der Freigabe „Portal“ im Unterverzeichnis „uploads“.

Das Script wird während des Setups automatisch nach /etc/cron.hourly kopiert und entsprechend einmal pro Stunde ausgeführt.

avrun

Das Tool ***avrun*** jagt einen Virenscan über alle allgemeinen Fileserver-Freigaben, die Home-Verzeichnisse und die Samba-Profil-Verzeichnisse. Es ist seit invis Version 6.7 R1 an Bord und wird so gewünscht per Cronjob jede Nacht um 0:00Uhr gestartet. Virenverseuchte Dateien werden ins invis Quarantäneverzeichnis /var/spool/infected verschoben.

Ob das Tool regelmäßig laufen soll, kann in der Datei /etc/invis/invis.conf eingestellt werden.

clean_recycle (Ab 7.1-R1)

Auf invis-Servern erzeugt Samba automatisch Papierkorb-Ordner (.recycle) in allen Freigaben außer „Transfer“. Hier landen alle von Benutzern gelöschte Dateien. Das Script ***clean_recycle*** bereinigt diese Verzeichnisse um ein Volllaufen der Partitionen durch eigentlich gelöschte Dateien zu verhindern. Es wird per Cron-Dienst täglich um 4:00Uhr ausgeführt. Die Funktion kann grundsätzlich in der invis-Konfigurationsdatei (/etc/invis/invis.conf) aktiviert bzw. deaktiviert werden, weiterhin kann das Maximalalter der der Dateien und Verzeichnisse in den Papierkörben festgelegt werden.

clean_transfer

Auf invis-Servern gibt es eine Fileserver Freigabe namens „transfer“ gedacht für alle möglichen Dateien, die mal eben anderen zur Verfügung gestellt werden sollen. Meist verwandelt sich dieses Verzeichnis schon nach kurzer Zeit in so etwas wie die Betriebsmülhalde, niemand fühlt sich dafür verantwortlich mal wieder Ordnung zu machen.

clean_transfer per Cronjob einmal täglich aufgerufen erledigt dies. Im Script können Sie das zu reinigende Verzeichnis und das maximale Alter von Dateien festlegen. Voreingestellt sind /srv/shares/transfer und 21 Tage. Alles was älter ist wird gelöscht.

Wenn Sie es verwenden möchten kopieren Sie es einfach nach /etc/cron.daily und vergessen Sie nicht Ihre Mitarbeiter darüber zu informieren. Zusätzlich wir im zu säubernden Verzeichnis eine Liesmich-Datei erzeugt, die auf das Löschen alter Dateien hinweist - aber wer liest sowas schon?

creategroupshare, createhome & deletehome

Dies sind weitere Helper-Scripts für das invis-Portal, die der Benutzer- und Gruppenverwaltung dienen.

Sie sind dazu gedacht beim Anlegen von Gruppen und Benutzern Gruppenarbeits- bzw. Home-Verzeichnisse anzulegen, bzw. beim Löschen von Gruppen und Benutzern deren Verzeichnisse für eine automatische Archivierung zu markieren.

Seit Version 6.8-R1 erzeugt ***createhome*** ein kleines Windows-Batchscript, welches bei der Domänenintegration von Windows7 Clients hilft. Das Script jeweils benutzerbezogene Script wird in der Freigabe „Service“ im Verzeichnis „winscripts“ abgelegt und **muss** direkt nach der ersten Anmeldung eines neuen Benutzers mit Administratorenrechten ausgeführt werden. Da beim Öffnen einer Windows-Kommando-Shell (cmd.exe) mit Administratorenrechten, darin die verbundenen Netzlaufwerke nicht zur Verfügung stehen, muss das Script unter Verwendung des UNC-Pfades aufgerufen werden:

```
\\server\service\winscripts\mvlibbenutzername.cmd
```

delssscache (ab Version 10.0)

Änderungen an der Konfiguration sss Daemons zur Linux-Client Anbindung erfordern meist das vollständige Löschen der SSS-Cache Datenbank. Die Ausführung des Kommandos:

```
linux:~ # sss_cache -UG
```

genügt in den meisten Fällen nicht. Anzuwenden ist das Script, nach Änderungen an der Datei

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

, wenn im Anschluss daran auch nach einiger Wartezeit das Kommando

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

falsche benutzer und Gruppen anzeigt. Dies gilt sowohl auf dem Server als auch auf angeschlossenen Linux-Clients. D.h. kopieren sie das Script ggf. auf alle Liinux-Clients.

Aufgerufen wird es ohne weitere Argumente:

```
linux:~ # delssscache
```

diskchecker

Das ***diskchecker*** Script kontrolliert sowohl Software-RAID-Verbünde wie auch Festplatten (***smartctl***) auf Fehler. Die Ergebnisse werden für das invis Portal aufbereitet nach /var/spool/results/diskchecker kopiert. Neben Fehlern werden im Portal auch die aktuellen Festplattentemperaturen angezeigt.

Tritt ein Fehler auf, wird unmittelbar eine Warn-Email an den zuständigen Administrator gesendet. Die Empfängeradresse ist im Script einzustellen.

Das Tool wird ab Version 6.7 R1 alle zwei Stunden per Cronjob ausgeführt.

Achtung: *Ab Version 6.9-R1 prüft* ***diskchecker*** *ob ein gefundenes Laufwerk SMART unterstützt. Dies dient dazu beispielsweise Cardreader von der Prüfung auszuschließen. Die Informationen dazu holt sich* ***diskchecker*** *aus dem Sys-Filesystem. Leider sind die Einträge, die SMART-Unterstützung anzeigen nicht einheitlich. Woher die Unterschiede kommen ist mir nicht bekannt. Der Eintrag der SMART-Unterstützung kennzeichnet muss derzeit manuell ermittelt und in /etv/invis/invis.conf eingetragen werden.*

Ermitteln des Strings:

```
linux:~ # udevadm info --query=all --path=/sys/block/sdX | grep SMART
```

Bisher bekannte Srings sind:

- ID_ATA_FEATURE_SET_SMART_ENABLED=1
- UDISKS_ATA_SMART_IS_AVAILABLE=1

Diese müssen in der invis Konfigurationsdatei hinter „SMARTString:“ eingetragen werden. Dabei ist auf genaue Schreibweise zu achten!

In Einzelfällen haben wir festgestellt, dass die Information, ob SMART unterstützt wird oder nicht, im Sys-Dateisystem fehlen. In diesem Fall kann eine Unterscheidung nach Anschlusstyp als Workaround herhalten. Hintergrund ist einfach, dass SMART-Abfragen am USB-Bus häufig nicht funktioniert, während sie an anderen Bus-Systemen kein Problem darstellen. In diesem Fall kann der folgende String in die invis-Konfiguration eingetragen werden:

- ID_BUS=ata

Bis einschließlich invis 6.9-R1-alpha9 muss der String noch direkt in das ***diskchecker*** Script eingetragen werden.

dwdatasnapshot (Ab 6.9-R1-alpha7)

… ist ein kleines Kommandozeilenwerkzeug zur Erstellung von Snapshots des Dokuwiki-Datenverzeichnisses. Es kann händisch, ohne Angabe von Optionen auf der Kommandozeile ausgeführt werden, wird aber auch einmal wöchentlich Samstags um 1:30Uhr per Cron-Dienst aufgerufen.

Ziel der Sicherung ist das Verzeichnis:

```
/srv/shares/archiv/sicherungen/vollsicherungen/dokuwikisicherungen/
```

Das Sicherungsziel kann in der invis-Konfigurationsdatei geändert werden.

emergmailer (Ab invis-Server 13.x)

Dieses Script ist nur ein Helferscript, welches von anderen Scripts des Servers zum Versand von Warn-Emails an den zuständigen Administrator verwendet wird, bzw. verwendet werden kann.

extzu (Ab 7.0-R1)

Das Script erweitert ein im LDAP vorhandenes POSIX- bzw. Samba-Benutzerkonto um die nötigen Zarafa-Attribute.

```
linux:~ # extzu username [0|1]
```

Der zweite Aufrufparameter kann die Werte „0“ bzw. „1“ annehmen und entscheidet darüber, ob es sich um einen reinen „Sharedstore“ (1) oder ein vollständiges Benutzerkonto (0) handelt.

Unterstrichener Text

Scripts rund um fetchmail

Alle nachfolgenden Scripts gibt es ab invis-Server Version 14.3.

addmailaccount

Um dem invis-Server externe Mailkonten bekannt zu machen beispielsweise um den Mail-Abruf per fetchmail zu organisieren wurde ursprünglich das invis-Portal benutzt. Das war und ist so aufgebaut, dass es durch die Benutzer selbst bedient werden kann. Praktisch für Benutzer, eher ungelenk für den Administrator. Letzterer muss sich mit den Zugangsdaten des Zielbenutzers am Portal anmelden um diesen Job zu erledigen. Leider hat sich in der Vergangenheit gezeigt, das „normale Benutzer“ entweder nicht in der Lage oder schlicht nicht willens sind Ihre Mail-Konten-Zugangsinformationen am

invis-Portal einzugeben. Schade eigentlich…

Um es dem Administrator einfacher zu machen, gibt es jetzt mit ***addmailaccount*** ein einfaches Script um dies auf der root-Konsole des Servers zu erledigen. Es ist ein interaktives Script und fragt die Informationen über ein Eingabeformular ab. Aufgerufen wird es wie folgt:

```
invis:~ # addmailaccount benutzername
```

Der Rest ist selbsterklärend.

changemacstate

Mit ***changemacstate*** wurde ein ergänzendes Script erstellt um den Email-Abruf für einzelne Benutzer zu aktivieren oder zu deaktivieren. D.h. Es werden Zeilen in die fetchmailrc-Datei eingefügt oder entfernt.

```
invis:~ # changemacstate benutzername [a|d]
```

Dabei stehen die Optionen **a** für aktivieren und **d** für deaktivieren.

refreshfrc

Sollten sich im LDAP des Servers Änderungen an den Zugangsdaten der externen Mailkonten ergeben haben, können diese im Block in die fetchmailrc-Datei übernommen werden. Das Script wird einfach ohne weitere Optionen aufgerufen und schreibt die fetchmailrc-Datei neu.

foldernames.php

Dieses Script gehört zu den Kopano-Administrationswerkzeugen. Es dient dazu die Namen der Standard-Ordner in verschiedene Sprachen zu übersetzen.

```
invis:~ # foldernames.php username sprache
```

Beispiel

```
invs:~ # foldernames.php heinz de_DE.UTF-8
```

Alternativ kann auch das Kopano-eigene Script ***kopano-localize-folders*** verwendet werden:

```
invis:~ # kopano-localize-folders -u heinzb --lang de_DE.UTF-8
```

freeports

Das Script ermittelt zufällig einen freien Port aus dem Bereich von 50000 bis 65535 zur beliebigen

Verwendung aus. Dieses Script wird verwendet um beispielsweise den SSH oder den HTTPs Port eines invis-Servers auszuwählen.

fixcacerts (Ab invis Version 13.1)

Bei verschiedenen openSUSE Leap Distributionsupgrades, etwa von 42.1 auf 42.2 oder von 42.2 auf 42.3 ist uns aufgefallen, dass das System danach keinen Zugriff mehr auf die mitgelieferten Stammzertifikate mehr hatte. Das bringt eine Reihe von Problemen mit sich, so ist es nicht merh möglich Software per ***zypper*** oder YaST nachzuinstallieren. Wie sich das Problem manuell beheben lässt haben wir an anderer Stelle hier im Wiki bereits beschrieben. Das Script ***fixcacerts*** führt alle Reparaturschritte automatisch aus:

```
invis:~ # fixcacerts
```

fixgsacls (Ab invis Version 14.1)

Das Script setzt „verkorkste“ Zugriffs-ACLs für die Gruppen-Arbeitsverzeichnisse in der Gruppen-Freigabe auf die Anfangswerte zurück. Gleichzeitig werden Verzeichnisse, die manuell auf der obersten Ebene der Gruppen-Freigabe angelegt wurden umbenannt indem das Script die Endung „-bitte_Support_anrufen“ an die Verzeichnisnamen anhängt.

Das Script kann sowohl auf der Kommandozeile des Servers, als auch aus dem invis-Portal heraus aufgerufen werden.

```
invis:~ # fixgsacls
```

getcertinfo (ab invisAD 10.4)

Das Script ist ein Helfer des invis-Portals. Es ermittelt ob die verschiedenen Server-Zertifikate noch gültig sind. Die Ergebnisse des Scriptlaufs werden auf der Status-Seite des invis-Portals angezeigt.

getusvvalues

Dieses Script kann Zustandsdaten einer APC-USV auslesen. Es wird auf invisAD Servern regelmäßig per Cron-Job ausgeführt. Es legt die Informationen nach

```
/var/spool/results/usv
```

ab. Dort werden sie von der Status-Seite des invis-Portals abgerufen. Die entsprechende Funktion muss in der Konfiguration des invis-Portals aktiviert werden. In Datei

```
/etc/invis/portal/config.php
```

muss die nachfolgende Zeile den Wert „true“ erhalten.

```
// Aktivieren der APCUPS Daemon Abfrage
$STATUS_APCUPSD = true;
```

getusvvalues funktioniert nur mit APC-USVs, die das „modbus“ Protokoll unterstützen und dieses aktiviert ist.

groupadd2ad (Ab invisAD 10.5)

Im invis-Portal können Benutzergruppen angelegt und mit Mitgliedern gefüllt werden. Dies ist, wenn viele Gruppen angelegt werden sollen allerdings mühsame Arbeit. Das Script ***groupadd2ad*** erledigt diese Aufgabe effizienter. Damit können auf der Kommandozeile anhand einer Liste viele Gruppen in einem Arbeitsschritt angelegt werden.

Die notwendige Liste hat das Format:

```
gruppenname,gruppentyp,beschreibung,mitglied1 mitglied2 ...
```

Dabei werden die einzelnen Felder mit einem Komma getrennt, die Liste der Mitglieder stellt ein einzelnes Feld dar, in dem die einzelnen Werte durch Leerzeichen getrennt werden. Die Angabe von Gruppen als Mitglieder einer Gruppe ist erlaubt, setzt aber voraus, dass die als Mitglied genannte Gruppe bereits exisisiert. Hier ist ggf. auf die richtige Reihenfolge in der Liste zu achten.

Alle neuen Gruppen erhalten automatisch die notwendigen Attribute für UNIX-Clients (SFU/RFC2307) sowie ggf. notwendige Attribute zur Nutzung der Gruppen in einer Groupware. Unterstützt wird hier bisher nur Zarafa.

Beim Gruppentyp wird zwischen „Security“ und „Distribution“ unterschieden. Es ist darauf zu achten, dass diese Angabe zwischen Groß- und Kleinschreibung unterscheidet. Der erste Buchstabe muss groß geschrieben werden.

Dabei sind Distribution-Gruppen lediglich Verteilergruppen für das Mailsystem und Security-Gruppen solche für die Besitzrechte bzw. ACLs im Dateisystem gesetzt werden können.

Anwendung

```
linux:~ # groupadd2ad /pfad/zur/liste.txt
```

hostadd2ad (Ab invisAD 11.0)

Normalerweise können neue IP-Geräte über das invis-Portal in den DHCP- und DNS-Datenbestand im Active Directory aufgenommen. Das ist solange OK, wie es sich um einzelne Geräte bzw. Hosts handelt. Sollen aber eine Reihe von Geräten aufgenommen werden, ist dieser Weg eher mühsam.

Genau hier setzt ***hostadd2ad*** an. Es kann eine Liste von Hosts verarbeiten und alle Geräte auf einmal ins AD integrieren.

Die Liste muss im Prinzip wie eine CSV Datei aufgebaut sein, allerdings ohne Überschriften. Trennzeichen ist das Komma.

Beispiel

```
a8:10:16:56:0c:21,172.16.1.65,tk-systel26,TK Systemtelefon 26
a8:10:16:56:0c:35,172.16.1.66,tk-systel27,TK Systemtelefon 27
a8:10:16:56:0c:34,172.16.1.67,tk-systel28,TK Systemtelefon 28
a8:10:16:56:0e:56,172.16.1.68,tk-systel29,TK Systemtelefon 29
a8:10:16:56:0d:bd,172.16.1.69,tk-systel30,TK Systemtelefon 30
a8:10:16:56:0d:bc,172.16.1.70,tk-systel31,TK Systemtelefon 31
```

Also:

MAC-Adresse,IP-Adresse,Hostname ohne Domain,Standort oder Bemerkung

Dabei darf die Standort-Spalte keine Umlaute enthalten und sie sollten bez. der IP-Adressen aufpassen, dass Sie in das Schema Ihres invis-Servers passen. Die Aufteilung der IP-Adressbereiche finden Sie in:

```
/etc/invis/portal/config.php
```

Dort können Sie sie auch nach eigenen Anforderungen anpassen.

Die zuletzt vergebene IP-Adresse eines Bereiches können Sie über das invi-Portal heraus finden. invis-Server unterscheiden die folgenden IP-Adressbereiche:

- Server
- Client-PCs
- Drucker
- IP-Geräte

Anwendung

```
linux:~ # hostadd2ad /pfad/zur/liste.txt
```

importics (Ab Version 12.1)

Tool zum automatischen Import von ICS-Kalenderdateien in die persönlichen Kalender der Groupware Kopano.

Die Funktion wird nach dem invis-Setup nicht automatisch aktiviert. Dies kann in der invis-Konfiguration vorgenommen werden. Alternativ kann ***importics*** auch manuell aufgerufen werden. Es müssen dazu zuvor Zugangsdaten eines Kopano-Admins in die invis-Server Konfigurationsdateien eingetragen werden.

invis.conf

```
...
```

```
# Kopano Admin Konto
kAdmin:kadmin

# iCAL Importer aktivieren [j/n]
iCalImport:n

# iCAL URL
iCalUrl:http://localhost:8080/ical/
...
```

invis-pws.cponf

```
# Kopano Admin PW
kAdminPass:kadmin-pass
```

In den Home-Verzeichnissen der invis-Server Benutzer existiert ein Unterverzeichnis namens „ics“ hier können die Benutzer ICS-Dateien ablegen. Das Script ***importics*** führt diese Dateien mit einer Sicherung des Zielkalenders zusammen und importiert Sie anschließend in den Kalender.

Achtung: *Der vom Script genutzte Weg ICS-Dateien mittels des Tools* **curl** *zu importieren birgt Risiken, da darüber importierte Daten den vorhandenen Zielkalender vollständig überschreiben. Daher wird zunächst der Kalender des Benutzers als ICS-Datei gesichert, dann mit den neuen Dateien kombiniert und wieder eingepflegt. Sollte es beim Reimport zu Problemen kommen, kann dies Datenverlust im Kalender zur Folge haben. Regelmäßige Datensicherungen der Kopano Stores sind also unabdingbar.*

Um das Script manuell zu nutzen muss es lediglich aufgerufen werden. Es benötigt keinerlei Aufrufargumente.

```
linux:~ # importics
```

Wird die Funktion über die invis-Server Konfiguration aktiviert wird das Script jede Stunde aufgerufen. Ob das die ideale Zyklus ist, kann ohne Tests mit sehr großen Kalendern nicht gesagt werden.

inetcheck

Zweck des Scripts ist die Überprüfung der Internetverbindung. Per Cronjob regelmäßig aufgerufen testet es, ob eine Verbindung generell steht, DNS funktioniert oder es zu Ping-Verlusten kommt - die Verbindung also schlecht ist.

Die Ergebnisse des Tests werden sowohl in Form einer HTML-Datei unter dem Namen inetcheck.html im DocumentRoot-Verzeichnis des Webservers (/srv/www/htdocs) als auch aufbereitet für das invis-Portal unter /var/spool/results/inetcheck bereit gestellt. (Seit es dieses Script gibt, haben sich Kundenanrufe nacht dem Muster: „der Server geht nicht“ deutlich verringert). Für das Script spielt es keine Rolle, ob die Internetvrbindung per DSL-Modem oder durch einen Router hergestellt wird.

Eine zweite Funktion ist die dynamische Aktualisierung eines DNS-Eintrages, ähnlich wie dynDNS. Voraussetzung zur Nutzung dieser Funktion ist der administrative Zugriff auf einen DNS-Server im Internet - etwa auf einen eigenen Root-Server.

Um diese Funktion aktivieren zu können müssen Sie Ihren Nameserver (/etc/named.conf) zunächst so konfigurieren, dass er DDNS-Updates erlaubt. Die Authorisierung sollte per dnssec-Schlüssel erfolgen.

```
zone "ihredomain.de" in {
        type master;
        file "ihredomain.zone";
            update-policy {
                grant invis.ihredomain.de. name invis.ihredomain.de. A;
                grant invis-zuhause.ihredomain.de. name invis-
zuhause.ihredomain.de. A
                ........
                };
};

        key invis.ihredomain.de. {
            algorithm HMAC-MD5;
            secret
"rqmdtOgn4+DHo023hGT++Ih0nAxfwph+UNtOXcVcLBNcckE+JvoiP+l5wVJNi4hHyCJZLVNyhau
4ENoioTZr2Q==";
        };

        key invis-zuhause.ihredomain.de {
            .......
```

Zunächst wird innerhalb einer Zonendefinition per „update-policy“ festgelegt, wer welche DNS-Einträge verändern darf. Das Schlüsselwort „grant“ erwartet hier etwa die Angabe eines Schlüsselnamens. Hinter „name“ folgt der zu aktualisierende Hostname und der Typ des DNS-Records - hier „A“.

Wie gezeigt können beliebig viele Update-Regeln pro Zone erstellt werden.

Außerhalb der Zonendefinition werden die Schlüssel eingetragen. Es empfiehlt sich für jeden zu aktualisierenden Host einen eigenen Schlüssel zu generieren, die erleichtert das Sperren einzelner Einträge. Ich habe es mir zur Gewohnheit gemacht den Schlüssel nach dem zu aktualisierenden DNS-Record zu benennen.

Damit alles möglichst reibungslos funktioniert sollten Sie die Zeitangaben in der Zonendatei möglichst auf Werte im Minutenbereich reduzieren.

Die Schlüssel werden wie folgt erzeugt:

```
dnssec-keygen -a HMAC-MD5 -b 512 -n HOST invis.ihredomain.de
```

Dabei werden zwei Schlüssel-Dateien (Private- & Public-Key) erzeugt. Das „secret“ für die Datei named.conf entnehmen Sie aus der auf „.private“ endenden Datei.

Kopieren Sie beide Dateien auf den DDNS-Client (invis Server) ins Verzeichnis /etc/ssl/ddns und passen Sie abschließend die folgenden Zeilen im inetcheck-Script an.

```
ddnsup="y"
nameserver="ihr.dnsserver.de"
```

```
fqdn="invis.ihredomain.de"
keynumber="12345"
```

Die fünfstellige Schlüssel-Nummer entnehmen sie den Dateinamen der Schlüsseldateien. Beispiel: „Kinvis.ihredomain.de.+157+***12345***.private“. Testen Sie es einfach von Hand aus. Wenn etwas nicht klappt, könnte das invis-Forum weiterhelfen.

Sie können das Script einfach alle paar Minuten per cron ausführen lassen, ein DDNS-Update erfolgt nur, wenn sich Ihre IP-Adresse seit dem letzten Update geändert hat.

Das Tool wir ab invis Version 6.7 R1 per Cronjob automatisch alle 10 Minuten ausgeführt.

inhume (Ab invis-Server Version 14.0)

Dieses Script ist dazu gedacht verwaiste Kopano-Stores und ownCloud-Konten zuvor gelöschter Benutzer ebenfalls zu löschen.

Hinweis: *Das Script legt keine Datensicherung der zu löschenden Daten an. Ziel ist es ja eben die verwaisten Daten zu löschen.*

ipservicestatus & ipservicecontrol (ab invisAD 10.1)

Beide Scripts sind nicht zur direkten Nutzung gedacht. Sie sind Helferscripts zur Verwaltung von Diensten über das invis-Portal.

invisarchiver (ab invisAD 10.4)

Mit Version 10.4 wurde die Archivierung von Benutzer- und Gruppendaten gelöschter Benutzer und Gruppen neu organisiert. ***invisarchiver*** wird zyklisch ausgeführt, es sucht nach zu archivierenden Daten und legt diese komprimiert unter

```
/srv/shares/archiv
```

ab.

Die Archiv-Freigabe ist seither nur noch für Mitglieder der Gruppe „Archiv“ möglich.

inviscerts (ab invisAD 10.5)

Mit Einführung von invis-Version 10.5 wurde (bzw. wird) die Erstellung von Server-Zertifikaten auf die Verwendung von „easy-rsa“ Version 3 umgestellt. Ab dieser Version verfügt der invis-Server nur noch über eine Zertifizierungsstelle (CA) und nicht wie zuvor je eine für das System und eine für OpenVPN. Die CA wird während des Server-Setups von ***sine*** angelegt. Mit ***inviscerts*** werden die verschiedenen Server- und Client-Zertifikate verwaltet. Es ist in der Lage Zertifikate zu erneuern, wenn diese

abgelaufen sind, neue zu erstellen oder VPN-Client-Zertifikate zurückzuziehen.

Anwendung:

```
linux:~ # inviscerts [intern|extern|ms|vpn|crl]
```

Sie benötigen für alle Operationen das Passwort Ihrer CA.

Die Optionen im Einzelnen:

- **intern** - Legt ein Zertifikat für interne Dienste an.
- **extern** - Legt ein Zertifikat für alle via Internet erreichbaren Dienste wie an.
- **ms** - Legt ein Zertifikat für den Mailserver an.
- **vpn** - Legt VPN Client-Zertifikate an.
- **crl** - Aktualisiert die Certificate Revocation List.

Detaillierte Erläuterungen zur Verwendung und Funktionsweise des Scripts sind **hier** zu finden.

invis-updater (ab invisAD 10.1 / Deprecated)

Hinweis: *Dieses Script wird ggf. wieder entfernt. Das Installieren von reinen Sicherheits-Update mittels openSUSEs* ***you*** *ist vorzuziehen.*

„fire and forget“ Script zur Installation von Updates auf invis-Servern. Dem Script kann über eine Negativliste gesagt werden, welche Updates als kritisch angesehen werden, In der Voreinstellung sind dies Kernel-Updates, Aktualisierungen der installierten SQL-Dienste oder auch VirtualBox, wenn installiert. Alle anderen Updates werden installiert und davon betroffene Dienste automatisch neu gestartet. Integriert wird weiterhin die Ausführung des Scripts „afterup“, letzteres muss also bei Verwendung des invis-updaters nicht mehr händisch ausgeführt werden.

Als kritisch eingestufte Updates müssen händisch installiert werden.

kbackup & kdbdump (Ab 7.0 R4 bis 11.0: zbackup & zdbdump)

Beide Scripts dienen der Sicherung der Daten einer Kopano-Installation. Während ***kbackup*** auf das Kopano-eigene Bricklevel-Backup-System zugreift erstellt ***kdbdump*** einen vollständigen Dump der zugehörigen Kopano MariaDB-Datenbank.

Aus Sicherungen, die mittels ***kbackup*** angelegt werden lassen sich gesamte Stores oder auch nur einzelne versehentlich gelöschte MAPI-Objekte wie etwa eine Email wiederherstellen. Verloren gehen dabei aber vollständig die auf Objekte und Verzeichnisse gesetzten Zugriffsrechte.

Der SQL-Dump hingegen dient dem Disaster-Recovery. Es ist immer sinnvoll beide Datensicherungen parallel zu betreiben.

Mit hinzufügen des Scripts ***kdbdump*** wurde die Sicherung der Kopano-Datenbank aus dem Script

alldump herausgenommen. Die Sicherung der Kopano-Datenbank stellt aufgrund der objektorientierten Datenhaltung andere Anforderungen an den Dump als etwa bei Group-e.

Die Sicherungen sind unter

```
/srv/shares/sicherungen
```

zu finden.

kmemcalc (Vor V. 12.0 zmemcalc)

Das Script berechnet auf der Grundlage des im Server zur Verfügung stehenden Arbeitsspeichers und Empfehlungen seitens Zarafa vernünftige Speichereinstellungen für MariaDB (innodb_buffer_pool_size, innodb_log_file_size) und Zarafa (cache_cell_size). Während der Installation nutzt ***sine*** das Script bereits um MariaDB und Zarafa zu konfigurieren.

Anwendung

```
linux:~ # kmemcalc
```

Ohne Aufrufargumente, ist das Script geeignet um von anderen Scripts genutzt zu werden. Es liefert die drei genannten Werte durch Leerzeichen getrennt, ohne erläuternde Texte.

Eine Menschen-lesbare Ausgabe erzeugt:

```
linux:~ # kmemcalc v
```

Achtung: *Wenn Sie zu irgend einem Zeitpunkt der Wert innodb_log_file_size in /etv/my.cnf ändern, müssen Sie MariaDB stoppen, dann die beiden Logfiles ib_logfile0 und ib_logfile1 in /var/lib/mysql löschen (oder verschieben) und dann MariaDB wieder starten.*

krbtest (ab invisAD 10.3)

Dieses Script testet, ob Kerberos funktioniert. Zur Anwendung benötigen Sie das Passwort des Domänen-Administrators.

Getestet werden ob die DNS-Servicelocation-Records korrekt sind und, ob eine Kerberos-Authentifizierung funktioniert.

membermod / mmall (letzteres seit invis-Server 14.1)

Das Scripts sind dazu gedacht ein Maschinen-Konto mit UNIX-Attributen zu erweitern. Notwendig ist das um beispielsweise Maschinen-Konten Zugriff auf Fileserver-Freigaben zu gewähren, etwa wenn Software via GPOs ausgerollt wird.

Dabei kann ***membermod*** gezielt auf einzelne Maschinenkonten angewendet werden:

```
invis:~ # membermod hostname
```

Als Aufrufargument wird der Computername des Maschinenkontos ohne Domäne erwartet.

mmall hingegen sucht im LDAP-Verzeichnis nach allen Maschinenkonten und erweitert diese, wenn nicht bereits geschehen, alle um UNIX-Attribute. Das Script kann sowohl auf der Kommandozeile:

```
invis:~ # mmall
```

(ohne Aufrufargumente) als auch ab invis-Server 14.1 aus dem invis-Portal heraus aufgerufen werden.

mkdbsilent

Ein kleines Script zum schnellen Anlegen von MySQL-Datenbanken nebst zugehörigem Benutzer. Das Script gibt lediglich das Passwort des Benutzers zurück.

```
linux:~ # mkdbsilent databasename username [a|r]
```

Der dritte Aufrufparameter kann die Werte „a“ und „r“ annehmen. Er gibt an, ob der User volle Rechte („a“) oder über einen reduzierten Rechtesatz („r“) bestehend aus „SELECT, INSERT, UPDATE, DELETE, CREATE, DROP, INDEX und ALTER“ verfügen soll.

mkkopanokey (Vor invis-Version 12.0: mkzarafakey)

Wenn die Kopano-Dienste verschlüsselt angeboten werden sollen müssen für Kopano Server-Schlüssel und Server-Zertifikat in einer Datei zusammen gefasst werden. Genau dass macht dieses Script.

Die resultierende Datei wird unter

```
/etc/invis/kopano/kopano.pem
```

abgelegt.

Hinweis: *Müssen Zertifikate erneuert werden, ruft das Script* ***inviscerts mkkopanokey*** *automatisch auf.*

mkkopres (Ab invis-Version 14.0)

Kleines Script zur Erstellung von Kopano-Ressourcen auf Basis vorhandener Kopano Shared-Stores.

```
invis:~ # mkkopres accountname resourcetype
```

Das Script erwartet zwei Aufrufargumente, zunächst den Kontonamen des zu erweiternden Shared-Stores und anschließend den Typ der anzulegenden Ressource. Beim Ressourcen-Typ wird zwischen

„equipment“ und „room“ unterschieden. Wird als Typ „equipment“ angegeben, fragt das Script noch danach, wie viele Exemplare der Ressource zur Verfügung stehen. Also beispielsweise „3 Beamer“. Kopano lehnt in diesem Fall eine Buchung erst ab, wenn die vorhandene Anzahl der Ressource überbucht wird.

moddnsrecords (Ab invisAD 10.0)

Das Script ist als „vereinfachendes“ Frontend für ***samba-tool*** zum Anlegen, Entfernen oder Ändern von DNS A- und PTR-Records gedacht. ***moddnsrecords*** wird auch vom invis-Portal zum Eintragen, Löschen oder Anpassen von DNS-Einträgen genutzt.

Anwendung

```
linux:~ # moddnsrecords {a|r|u} {A|PTR} dnsobjekt wert
```

Darin bedeutet:

- a: add / hinzufügen
- r: remove / entfernen
- u: update / aktualisieren

Beipiele

A-Record hinzufügen:

```
linux:~ # moddnsrecords a A pc-buchhaltung 192.168.222.124
```

Bei A-Records wird als Objektname nur der Host-Anteil des DNS-Namens angegeben.

PTR-Record entfernen:

```
linux:~ # moddnsrecords r PTR 192.168.222.124 pc-buchhaltun.ad-net.loc
```

Bei PTR-Records muss sowohl die vollständige IP-Adresse, als auch der volle FQDN eines Hosts angegeben werden.

PTR-Record ändern:

```
linux:~ # moddnsrecords u PTR 192.168.222.124 pc-buchhaltung.ad-net.loc
```

Hinweis: *Geändert werden kann lediglich der Wert und nicht der Name eines Objekts. Die Umbenennung eines Objektes kann nur durch löschen und erneutes anlegen realisiert werden.*

netsetup

Das Script ***netsetup*** benötigen Sie nur ein einziges Mal **vor** der Installation des invis-Servers. Es legt UDEV-Regeln für die Benennung der Netzwerkschnittstellen in „extern“, „intern“ und ggf. „dmz“ an.

ocsubsync (Ab invis-Version 13.2)

Das Tool ***ocsubsync*** ist dazu gedacht, lokale ownCloud-Konten in eine ebenso lokale Gruppen-Freigabe zu synchronisieren. Genutzt wird es Daten von Freelancern oder Filialen die mit der lokalen ownCloud Installation sychronisiert werden, auch für lokale Benutzer verfügbar und veränderbar zu machen.

Gedacht ist es so, dass eine lokale Gruppe mit Gruppenverzeichnis angelegt wird und ein ownCloud-Konto in dieses Gruppenverzeichnis synchronisiert wird. Diese Dateien können jetzt lokal von Mitgliedern der Gruppe bearbeitet werden, diese Veränderungen werden genauso mit ownCloud synchronisiert wie Dateien die extern geändert werden.

ocsubsync kennt folgende Aufrufparameter:

- **(add|rem) username** - Fügt ein neues Konto der Synchronisation hinzu oder entfernt es.
- **sync [username]** - Synchronisiert entweder alle konfigurierten Konten oder nur das angegebene.
- **jobs** - zeigt an, ob gerade Synchronisationsjobs laufen.
- **accounts** - gibt alle zur Synchronisation konfigurierten Konten aus.

pwsettings (Ab invisAD 10.4)

Mit dem Script ***pwsettings*** können die Einstellungen zur Passwortsicherheit im Active Directory vorgenommen werden. Es können Passwort-Laufzeit, -Komplexität und -Länge eingestellt werden. Das Script wird einfach ohne weitere Optionen auf der Kommandozeile aufgerufen:

```
linux:~ # pwsettings
```

rpmkeyimporter

Dieses Script ist dazu gedacht öffentliche Schlüssel von Buildservice-Repositories in die RPM-Datenbank zu importieren. Notwendig war dies als das Setup-Script noch auf dem Paketmanager ***smart*** basierte. Mit dem Umstieg auf ***zypper*** war es vorübergehend überflüssig geworden.

Inzwischen ist es an zypper-Repositories angepasst und wird zukünftig wieder vom Setup-Script genutzt um die Integration der zusätzlichen Repositories zu automatisieren.

runkopano (Vor V. 12.0: runzarafa)

Script zum Starten und Stoppen aller Kopano-Dienste.

```
linux:~ # runkopano [start|stopp|status]
```

Die Option „status“ ist erst ab invisAD Version 11.0 enthalten. Ebenfalls ab Version 11.0 kann in der

invis-Server Konfigurationsdatei

```
/etc/invis/invis.conf
```

festgelegt werden welche Dienste von ***runkopano*** berücksichtigt werden sollen.

safebootpart

Das Script ist dazu gedacht ein Backup der Bootpartition in Form eines Images anzulegen. Ziel der Sicherung ist das Archiv-Verzeichnis des invis-Servers.

safebootpart wird beispielsweise von „invis-rdbu“ unserer eigenen Datensicherungslösung verwendet.

scanleases (Ab 6.9-R1-alpha7)

Nicht immer läuft die Pflege eines Netzwerkes wie gewünscht. Wildwuchs ist schnell passiert. Mal eben einen neuen Rechner oder Drucker an's Netz gestöpselt und vergessen. Klappt ja alles, wozu gibts den DHCP Server.

Geschieht dies zu oft, tummeln sich einige Geräte im Netz deren IP-Adresse niemand kennt, bzw. es lassen sich IPs anpingen, von denen niemand weiß, welches Gerät dahinter steckt.

Hier hilft ***scanleases***, es arbeitet sich durch die Leases-Datenbannk des DHCP-Servers und sammelt Informationen über die darin aufgeführten Geräte. Die Ergebnisse des Scans, schreibt es nach:

```
/var/spool/results/dhcpscan/leases-scan.txt
```

Möglicherweise lässt sich diese Datei auch mal im Portal verarbeiten….

serverkeys (Ab Version 6.8 R7 bis invisAD 10.4)

Hinweis: *Dieses Script wird ab invis-Server AD 10.5 durch das Script* ***inviscerts*** *ersetzt.*

Mit diesem Script lassen sich (einfacher als zuvor) Schlüssel für Web- und Mailserver erzeugen. Im Unterschied zu seinen Vorgängern entfällt die ganze Fragerei. Die notwendigen „Common Names“ werden über eine dynamisch generierte Konfigurationsdatei eingelesen und ***openssl*** im Batch-Betrieb ausgeführt. Es wird also zum Schlüsselbau nur noch das Passwort der lokalen CA benötigt.

Ausgeführt wird das Script wie folgt:

```
linux:~ # serverkeys ms|ws|ldap
```

Dabei stehen die Parameter *ms*, *ws* und *ldap* einfach für Mail-, Web-Server und LDAP-Server.

sudorule2ad (Ab Version 13.1)

Seit einiger Zeit arbeitet das invis-Team daran Sudoers-Regeln im ActiveDirectory zu speichern und via PAM zu nutzen. Leider ist unsere Arbeit noch nicht von Erfolg gekrönt. Zwar funktioniert das Speichern der Regeln im AD, allerdings die Nutzung durch ***sudo*** noch nicht.

swpestat

Mit ***swpestat*** lassen sich einzelne Links im invis-Portal aktivieren bzw. deaktivieren.

Um sich einen Überblick über den Status aller Portal-Links anzuzeigen führen Sie folgendes Kommando aus:

```
linux:~ # swpestat status
invisAdminManual: FALSE | invisUserManual: TRUE | Shell-in-a-Box: TRUE |
openSUSEManual: TRUE | OpenStreetMap: TRUE | phpvirtualbox: TRUE |
Mailmanadmin: FALSE | Routenplaner: TRUE | apacheManual: TRUE |
phpLDAPadmin: TRUE | GelbeSeiten: TRUE | Telefonbuch: TRUE | WebCDwriter:
FALSE | invisServer: TRUE | GoogleMaps: TRUE | Kachelmann: TRUE |
Tagesschau: TRUE | phpMyAdmin: TRUE | phpPgAdmin: TRUE | FGCManual: TRUE |
FaxClient: FALSE | Kivitendo: TRUE | Roundcube: FALSE | Telepolis: TRUE |
Wikipedia: TRUE | ZarafaAcc: FALSE | ZarafaApp: TRUE | invisBlog: TRUE |
invisWiki: TRUE | Dokuwiki: TRUE | Etherpad: FALSE | ownCloud: TRUE |
waWision: FALSE | DictCHi: FALSE | Group-e: FALSE | Mailman: FALSE |
MetaGer: TRUE | Spiegel: TRUE | Verkehr: TRUE | ixquick: TRUE | CorNAz: TRUE
| DictEN: TRUE | DictES: TRUE | DictFR: TRUE | DictIT: TRUE | FWTest: TRUE |
Google: TRUE | Heise: TRUE | Bahn: TRUE | CUPS: TRUE | SOGo: FALSE | ntop:
TRUE |
```

Aktivieren oder Deaktivieren funktioniert wie folgt:

```
linux:~ # swpestat entryname [TRUE|FALSE]
```

Dabei muss „entryname“ genau einem der in der obigen Liste angezeigten Namen entsprechen.

upgradead (Ab invis-Server Version 14.1)

Trotz der merkwürdigen Benennung des Scripts hat es nicht mit dem Tod zu tun. Der Name setzt sich aus „upgrade“ und „ad“ zusammen. Gedacht ist es um eine ActiveDirectory-Sicherung aus einer Samba-Version kleiner 4.8 in einer Samba 4.10 Umgebung wiederherzustellen. Kurz um ein bestehendes AD in einem aktuellen invis-Server ab Version 14.1 wiederzubeleben.

Als Aufrufargument erwartet das Script den Pfad zur wiederherzustellenden AD-Sicherungsdatei:

```
invis:~ # upgradead
```

```
/srv/shares/archiv/sicherungen/vollsicherungen/ad/Samba_20190729-172425.tar.gz
```

Beachten Sie dabei, dass bei diesem Vorgang das bestehende AD des Servers auf dem das Script ausgeführt wird natürlich ersetzt wird. Das Script erzeugt vorsichtshalber vorab eine Sicherung des Bestands.

Scripts rund um Virtualbox

addvbsubnet (ab invis Version 15.0)

Wird eine virtuelle Maschine per „Bridge“ (Netzwerkbrücke) mit dem lokalen Netzwerk verbunden kann dass dazu führen, dass der lokale Zugriff auf die VM (beispielsweise per Remotedesktopverbindung) dazu führen sehr langsam ist, bzw. immer wieder kurz hängt. Woran das genau liegt konnten wir nicht herausfinden, außer dass sowohl der aus dem lokalen Netz eingehende Datenverkehr, als auch der zur VM laufende Verkehr physisch eine Netzwerkkarte teilen.

Um dies zu umgehen wird es jetzt möglich virtuelle Maschinen an ein eigenes „internes“ Subnetz (Host-only-Network) zu binden und dieses wiederum der internen Firewall-Zone des invis-Servers zuzuordnen und vom lokalen DHCP-Server mit IP-Adressen zu versorgen.

Das Script fügt ein solches Host-only-Net zu Virtualbox hinzu, fügt es als Subnet in die LDAP-gestützte Konfiguration des DHCP-Servers ein und auch in die interne Firewall-Zone. Zwischen den NEtzwerkschnittstellen der internen Zone ist grundsätzlich ein Routing konfiguriert, d.h. die virtuellen Maschinen, die in diesem Netz laufen sind aus dem lokalen Netz direkt erreichbar und können auch der lokalen Domäne beitreten.

Anwendung:

```
invis:~ # addvbsubnet ip-adresse subnetz/maske
```

Dabei steht „ip-adresse“ für die Adresse die die zugehörige „vboxnet“ Schnittstelle am invis-Server bekommt, „subetz“ für die Netzwerkbasis-Adresse und „maske“ für die Subnetzmaske in Langschreibweise.

Im Anschluss an die Ausführung des Scripts muss der DHCP-Server am invis-Server neu gestartet werden:

```
invis:~ # systemctl restart dhcpd.service
```

addvm2subnet (ab invis Version 15.0)

Dieses Script ist dazu gedacht neue DHCP-Reservierungen in VBox-Subnetzen einzurichten und für diese DNS-Records (Forward & Reverse) anzulegen.

Hinweis: *das Script kann auch für das Standard-Netz verwendet werden. Anders als das invis-Portal ermittelt es aber nicht automatisch die zuletzt vergebene Adresse eines Netzes und teilt die Hosts auch nicht gemäß den Vorgaben „Server“, „PC“, „Drucker“ oder „IP-Gerät“ auf. Das ist damit*

Handarbeit. Um die nächste freie IP-Adresse zu ermitteln, kann es aber die bereits vergeben Adressen auflisten und auch die vorhandenen Subnetze anzeigen.

Das Script im Query-Modus:

```
invis:~ # addvm2subnet q IP-Part
```

Das „q“ steht für „query“ und IP-Part für den Teil einer IP-Adresse der im Subnetz liegt oder es kennzeichnet. Also etwa:

```
invis:~ # addvm2subnet q 172.18.4
CN=DHCP Config,CN=DHCP-Server,cn=invis-server,DC=invis-server,DC=lan
Ausgabe vergebener IP-Adressen:
172.18.4.2
172.18.4.5
172.18.4.1
172.18.4.3
172.18.4.4
Mögliche Subnetze sind:
172.18.0.0
172.19.0.0
```

Im Beispiel wäre dann die IP-Adresse 172.18.4.6 die nächste freie Adresse.

Das Sript im aktiven Modus:

Um einen Host mit der gewünschten Adresse ins gewünschte Subnetz einzutragen sieht der Scriptaufruf wie folgt aus:

```
invis:~ # addvm2subnet 08:00:27:1f:d2:9a 172.19.4.1 erp1-vm
```

Achten Sie bei Eingabe der Mac-Adresse auf die Schreibweise, alle Buchstaben klein geschrieben und der Doppelpunkt als Trennzeichen. Es folgt die gewünschte IP-Adresse im anvisierten Subnetz und der Hostname ohne Domain.

vboxstop

Wenn während eines Updates des Servers auch VirtualBox aktualisiert werden soll, scheitert das meist daran, dass noch Komponenten von VirtualBox aktiv sind. ***vboxstop*** beendet alle störenden Prozesse.

Hinweis: *Fahren Sie vorher alle VMs herunter.*

wawips (Ab Version 13.0)

Dieses Tool wird per Cronjob regelmäßig ausgeführt und dient dazu in der ERP-Lösung WaWision Wiedervorlagen auszulösen.

zarafa_schema_add.sh

Dieses Script wird während des Setups von ***sine*** genutzt um die notwendigen Zarafa Schemaerweiterungen ins AD einzupflegen. Danach wird es nicht mehr gebraucht.