GnupgCard und LTSP

LTSP (=Linux Terminal Server Project) ist ein Terminal-Server System für plattenlose Thinclients. Diese werden über PXE gebootet. Die grafische Oberfläche, sowie die meisten Programme laufen remote auf dem Server.

Prinzipiell ist die Möglichkeit der Smartcard-Nutzung bei Remote Anwendungen eingeschränkt, da das Kartenlesegerät mit der lokalen Hardware verbunden ist.

Da LTSP aber auch lokal laufende Applikationen unterstützt, kann die GnupgCard auch mit LTSP eingesetzt werden.

Installation der nötigen Software im Chroot

Die nötige Software wie gnupg2 und pcscd, muss lokal im LTSP Chroot installiert sein.

```
ltsp-chroot -m
apt-get install gnupg2 scdaemon gpgsm pcscd
exit
```

Ob die Karte erkannt wird, lässt sich in einem lokalen xterm testen:

```
ltsp-localapps xterm
```

Im neuen xterm kann dann mit

```
gpg --card-status
```

getestet werden, ob alles richtig konfiguriert ist.

Im Home-Direcotry der jeweiligen User sollte gnupg genauso eingerichtet werden, wie bei einer normalen lokalen Verwendung.

Terminal

Um z.B. den ssh-support des gnupg-agent nutzen zu können, müssen wir auch die Terminal-Anwendung lokal laufen lassen und in der .bashrc konfigurieren, dass der gnupg-agent gestartet wird.

Als Standard gibt es ein lokales xterm innerhalb des LTSP-Chroot. Dieses kann entweder aus dem Menü oder per `ltsp-localapps xterm` gestartet werden. Alternativ kann natürlich ein komfortableres Terminal als lokale Applikation installiert werden (z.B. mate-terminal).

Die folgende .bashrc Konfiguration stellt sicher, dass der gpg-agent http://unix.stackexchange.com/questions/46960/how-to-configure-gpg-to-enter-passphrase-only-once-per-session - und nur dann, wenn es sich um eine lokale Session handelt.

In die .bashrc muss folgendes eingefügt werden:

.bashrc

```
# Providing gpg-agent in a local-apps-environment
# start only, if we are local
if   ip addr|grep -q $LTSP_CLIENT/24 ; then
    # Invoke GnuPG-Agent the first time we login.
    # Does `~/.gpg-agent-info' exist and points to gpg-agent process
accepting signals?
    if test -f $HOME/.gpg-agent-info && \
        kill -0 `cut -d: -f 2 $HOME/.gpg-agent-info` 2>/dev/null; then
        GPG_AGENT_INFO=`cat $HOME/.gpg-agent-info | cut -c 16-`
        echo "gpg-agent already running, not starting a second one"
    else
      # No, gpg-agent not available; start gpg-agent
      echo "no running gpg-agent found. starting"
      gpg-agent --daemon --enable-ssh-support \
      --write-env-file "${HOME}/.gpg-agent-info"
    fi

    # the next script looks for a running gpg-agent and exports the
    # environment variables
    if [ -f "${HOME}/.gpg-agent-info" ]; then
       . "${HOME}/.gpg-agent-info"
       export GPG_AGENT_INFO
       export SSH_AUTH_SOCK
       echo "connect to running gpg-agent"
    fi
fi
```

Weitere Anwendungen

Generell gilt: alle weiteren Anwendungen, die auf gnupg zugreifen sollen, müssen als lokale Applikationen installiert werden.

Weiterhin muss sichergestellt werden, dass diese aus einem Terminal heraus gestartet werden, in dem die gpg-agent Umgebungsvariablen bekannt sind. Das Starten aus dem Mate-Menü heraus leistet das nicht.

Icedove und Enigmail

Icedove mit Enigmail funktioniert problemlos, wenn beide als lokale Applikationen installiert sind. Allerdings greifen sie nicht auf einen ggf. laufenden gpg-agent zurück. Soll also eine Passphrase für Schlüssel, die nicht auf der Karte sind, zwischen gespeichert werden oder wird aus anderen Gründen (s.u.) ein gpg-agent benötigt, muss Icedove aus einem Terminal mit entsprechend laufendem gpg-

agent heraus gestartet werden.

Alternativ kann unter /usr/share/applications/ eine .desktop Datei angelegt werden, die mit einem Skript verknüpft ist, das wiederum die Umgebungsvariablen für den gpg-agent setzt und dann die Anwendung startet:

/usr/share/applications/icedove-local-wrapper.desktop

```
[Desktop Entry]
Name=Icedove-Lokal
Comment=Read/Write Mail/News with Icedove
GenericName=Mail Client
Exec=ltsp-localapps icedove-local-wrapper
Terminal=false
X-MultipleArgs=false
Type=Application
Icon=icedove
Categories=Network;Email;News;GTK;
MimeType=message/rfc822;x-scheme-handler/mailto;
StartupWMClass=Icedove-bin
StartupNotify=true
Name[ca]=Client de correu Icedove
Name[cs]=Poštovní klient Icedove
Name[fi]=Icedove-sähköposti
Name[fr]=Messagerie Icedove
Name[ja]=Icedove メールクライアント
Name[pl]=Klient poczty Icedove
Name[pt_BR]=Cliente de E-mail Icedove
Name[sv]=E-postklienten Icedove
Comment[ca]=Llegiu i escriviu correu
Comment[cs]=Čtení a psaní pošty
Comment[de]=Icedove (lokal) Emails lesen und verfassen
Comment[fi]=Lue ja kirjoita sähköposteja
Comment[fr]=Lire et écrire des courriels
Comment[it]=Leggere e scrivere email
Comment[ja]=メールの読み書き
Comment[pl]=Czytanie i wysyłanie e-maili
Comment[pt_BR]=Ler e escrever suas mensagens
Comment[sv]=Läs och skriv e-post
GenericName[ja]=メールクライアント
Keywords=Mail;Contact;Addressbook;News;
```

und das Wrapper-Skript:

/opt/ltsp/i386/usr/local/bin/icedove-local-wrapper

```
#!/bin/bash
# We need this wrapper for starting local apps which need a running
gpg-agent

```

```
# Providing gpg-agent in a local-apps-environment

# Providing gpg-agent in a local-apps-environment
# start only, if we are local
if   [ "$LTSP_CLIENT" != "" ] && ip addr|grep -q $LTSP_CLIENT/24 ; then
        echo "We are in ltsp-environment. Starting gpg-agent."

        # Invoke GnuPG-Agent the first time we login.
        # Does `~/.gpg-agent-info' exist and points to gpg-agent
process accepting signals?
        if test -f $HOME/.gpg-agent-info && \
            kill -0 `cut -d: -f 2 $HOME/.gpg-agent-info` 2>/dev/null;
then
            GPG_AGENT_INFO=`cat $HOME/.gpg-agent-info | cut -c 16-`
#            echo "gpg-agent already running, not starting a second
one"
        else
          # No, gpg-agent not available; start gpg-agent
#          echo "no running gpg-agent found. starting"
          gpg-agent --daemon --enable-ssh-support \
          --write-env-file "${HOME}/.gpg-agent-info"
        fi

        # the next script looks for a running gpg-agent and exports the
        # environment variables
        if [ -f "${HOME}/.gpg-agent-info" ]; then
           . "${HOME}/.gpg-agent-info"
           export GPG_AGENT_INFO
           export SSH_AUTH_SOCK
#           echo "connect to running gpg-agent"
        fi
fi

# start icedove
which icedove && icedove
```