Werkzeuge

0. Einleitung

Unter UNIX gibt es viele kleine Werkzeuge, das jedes für sich nur eine kleine Aufgabe erfüllt. Das sie durch UNIX–spezifische Methoden sehr effektiv verknüpft werden können, ist es möglich aus vielen kleinen Werkzeugen ein großes Werkzeug zu generieren, das in der Lage ist eine komplexe Aufgabe zu lösen. Der Anwender muß dafür kaum über Programmierkenntnisse verfügen.

Anstelle einer Einleitung – einige Beispiele

1. **Private CA erstellen**
2. **Einrichtung eines Nutzers am Instiut für Informatik**
3. **Ändern eines Nutzerpasswortes**
4. **Überwachung von Rechnern via Netzwerk**

```
1. Beispiel
-----------

   Scripte zur Verwaltung einer kleinen privaten CA. Dieses Script wird
   seit vielen Jahren mit OPENSSL ausgeliefert.

   Einsatzmöglichkeiten von Zertifkaten:
      1. TLS - sichere Datenübertragung im Firmennetz
      2. OPENVPN - virutelles privates Netz über Internet
      3. LDAP
      4. Mail (sendmail, imap, pop)

   Aufgaben:
      1. CA erzeugen
      2. Zertifikate für Hosts/Nutzer erzeugen

   Bestandteile:
      CA.sh     - Shell-Script, Schnittstelle zu OPENSSL
      CA.pl     - Perl-Script

   Quelle zum Anschauen:

          rabe:/opt/csw/ssl/misc/         - Solaris
          gruenau4:/usr/share/ssl/misc/   - Linux - SuSE 11.4
```



```
2.Beispiel
----------

   Einrichtung eines Nutzers am Instiut für Informatik

    1. Nutzer füllt Formular auf dem WWW-Server aus.
         Mittel:    - HTML-Seite
                    - C-Programm
         Ergebnis:  Ein geprüfter Datensatz auf dem WWW-Server

    2. Der Account-Verwaltungsrechner holt sich diese Datensätze
         Mittel:    - Shell-Script (get-new)
         Ergebnis:  Datensätze auf dem Account-Verwaltungsrechner
                    Datensätze auf dem WWW-Server gelöscht

    3. Der Account wird erzeugt.
         Mittel:    - Shell-Script (make, ldapadd, ...)
         Ergebnis:  Account erzeugt (passwd-Eintrag, Netzgruppeneintrag,
                                     Mail-Eintrag, Homedirectory angelegt)
                    Antrag wird gedruckt.
```

Datensatz, der von account.html (Formular) und account (C-Programm) auf dem WWW-Server erzeugt wird.

```
Name="Musterfrau"
Vorname="Gerlinde"
Staat="deutsch"
ImmNr="123456"
ImmJahr="2004"
Hauptf="Religionswissenschaften"
Nebenf="Informatik"
PLZ="12345"
Ort="Motzen"
Strasse="Nirgendwostr. 1"
Telefon="030/123456"
Dozent="Mustermann"
IPAdresse="195.93.64.177"
```

Dieser Datensatz wird von get-new auf auf den Account-Rechner kopiert. Anschließend wird dort "make-accounts" abgearbeitet. "make-accounts" überprüft noch einmal die eingegangenen Daten und generiert daraus alle notwendigen Daten für den Account. Anschließend wird ein Stammdatensatz für den Nutzer erzeugt.
Dazu werden aus diesem Script weitere Scripte aufgerufen:

```
useradd-script     - Shell-Script, das den Nutzer in LDAP, NIS und
                     Active-Dirctory anlegt
netguseradd-script - Shell-Script, Zuordnen des Nutzers zu einer Netzgruppe
mkhome             - Shell-Script, Anlegen und Initialisieren des
                     Homedirekctories (fob)
addmail            - Shell-Script, Erzeugen der Mailinformationen (mailbox)
```


Weitere benutzte UNIX-Werkzeuge:

grep, find, awk, sed, make, dd, ls, ssh, ldapadd, latex, dvips, pwgen

Erzeugter Stammdatensatz:

```
Name="Musterfrau"
Vorname="Gerlinde"
Staat="deutsch"
ImmNr="123456"
ImmJahr="2004"
Hauptf="Religionswissenschaften"
Nebenf="Informatik"
PLZ="12345
Ort="Motzen"
Strasse="Nirgendwostr. 1"
Telefon="030/123456"
Dozent="Mustermann"
IPAdresse="195.93.64.177"
UID="musterfr"
GRUPPE="nebenf04"
DATE="Mon Oct 20 18:07:51 MET DST 2004"
```

```
3. Beispiel
-----------

Ändern eines Nutzerpasswortes


Problem:

  Der Nutzer will in einer Institution nur eine Identität mit einem Passwort
  benutzen. Man hat aber mehrere Betriebssysteme mit unterschiedlichem
  Passwort-Managment, z.B. UNIX (NIS, Kerberos, LDAP) , Windows. Jedes
  System hat eine eigene Passwortdatenbank mit unterschiedlichen
  Crypt-Algorithmen.  Der Nutzer muß also bei jedem System sein Passwort
  einzeln neu setzen

             - sehr unfreundlich.


Wunsch:

  Eingabe des Passwortes an einer zentralen Stelle und automatisches
  Verteilen des Passwortes auf die verschiedenen Datenbanken.
```



```
Probleme:
---------

  Identifikation des Nutzers auf dem WWW-Server

  Sichere Ablage der Informationen auf dem WWW-Server

  Sichere Übertragung des neuen Passwortes zu den Datenbanken

  Zeitweises Merken der Passwortänderung, falls ein Account-System
  z.Z. nicht funktionsfähig ist.

Lösung:
-------

   WWW-Formular mit einigen Scripten ist dafür das geeignete Mittel.

   1. html-Formular zur Eingabe von
                Nutzername, Altes Passwort, 2 mal neues Passwort

   2. CGI-Script zur sicheren Ablage der Informationen  auf
            dem WWW-Server

   3. Übertragung der Informationen auf den Accountserver.
      Berechnung der verschiedenen gekrypteten Passworte.
      Einspeichern der Informationen in die Datenbasen
```

```
auf dem WWW-Server
------------------

Auf dem WWW-Server muß ein Formular bereitgestellt werden,
das dem Anwender die Eingabe der notwendigen Daten ermöglicht.

Notwendige Angaben sind:

   Nutzername
   altes Passwort
   neues Passwort (aus Sicherheitsgründen zweimal)


Anschließend muß von dem Formular aus ein Programm aufgerufen
werden, daß die Eingabedaten verarbeitet.

Der WWW-Server sollte eine gewisse Sicherheit aufweisen!!
```



```
Das HTML-Formular:      pass.shtml

<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
<!--#include virtual="/head.html" -->
<title>Ändern eines WWW-Passwortes</title>

<h1>Ändern des Domain-Passwortes </h1>
<h1>(rür UNIX und zentrale Windows-Domain)</h1>

<hr size=1>
<h4>
    Analog zum UNIX-Kommando "passwd" werden Nutzername und
    altes Passwort erfragt, um festzustellen, ob der Betreffende
    eine Änderung des Passwortes vornehmen darf.
    Um Tippfehler zu vermeiden, muss das neue Passwort zweimal
    angegeben werden.
<br><br>
    Alle Studenten und die meisten Mitarbeiter haben Zugriff zur
    Domain "all" und ändern bitte dort ihr Passwort.
<br><br>
    Mitarbeiter, die nur in einer Lehrstuhl-Domain bekannt sind,
    können ihr Domain-Passwort ändern, in dem sie die
    entsprechende Domain auswählen.
<br><br>
    Das Passwort eines Nutzers wird grundsätzlich in allen Domainen
    geändert, in denen er bekannt ist (Unix und Windows).
<br>
<hr size=1>
<br><br>
```

```
<form action=
     "https://base3.informatik.hu-berlin.de/cgi-bin/change-passwd"
           method="POST">
   Domain:                          <br>
   <select type="text" name="Subdomain" value="all" >
       <option> all
       <option> sar
       <option> alkox
   </select>   <br>
   Account:                     <br>
   <input type="text" name="User" >            <br>
   Altes Passwort (nicht sichtbar):            <br>
   <input type="password" name="oldPassword">   <br>
   Ein Passwort muss mindestens 7 Zeichen lang sein und mindestens einen
   Buchstaben, eine Ziffern und ein Sonderzeichen enthalten. Das neue
   Passwort sollte nicht mit dem alten Passwort identisch sein.
   <p>
   Neues Passwort (zur Sicherheit bitte zweimal eingeben):      <br>
   <input type="password" name="Password1" >
   <input type="password" name="Password2" >    <br>
   <input type="submit" value="Neues Passwort eintragen">
   </p>
</form>
</h4>
<!--#include virtual="/end.html" -->
```


Aufgerufenens CGI-Script

Das Programm, das das die Eingabedaten prüft und dann ablegt, ist ein sogenanntes "CGI-Script". Es kann ein binäres Programm, ein Perl-Script, ein Shell-Script oder ... sein.
Je nach Anwendungshäufigkeit sollte das Programm geschrieben sein - Scripte benötigen höhre Prozessorleistungen als binäre Programme!!
In diesem Beispiel wird ein Shellscript benutzt, das folgende Funktionen realisiert:

Prüfen der Aufrufparameter des Programmes.

Prüfen des Nutzernamens und des alten Passwortes mittels LDAP-Abfrage.

Prüfen der Identität der neu eingegeben Passworte.

Prüfen der Komplexität des neu eingegebenen Passwortes.

Verschlüsseln der Informationen und Abspeichern, damit sie später weiterverarbeitet werden können.

Kommunikation mit dem Nutzer (Erfolgsmeldung oder Fehlermeldung)

Das CGI-Script(Shell-Script):   change-passwd

```
#!/bin/sh
set -f
TEMP=/tmp/XXXXX$$
TEMP1=/tmp/YYYYY$$
SUBMIT=/var/submit
SLEEP=0
/usr/bin/echo $REMOTE_ADDR >>/tmp/logpass
sendmsg() {
 if [ $SLEEP -ne 0 ] ; then  sleep $SLEEP; fi
 /usr/bin/echo "Content-type: text/html\n"
 /usr/bin/echo '<!DOCTYPE html PUBLIC' \
                    '"-//W3C//DTD HTML 4.01 Transitional//EN">'
 /usr/bin/cat /var/apache2/htdocs/head.html
 /usr/bin/cat <<ERROR
     <TITLE>Ändern des Domain-Passwortes</TITLE>
     <H1> Ändern des Domain-Passwortes</H1>
     <body>
     <hr size=1> <br>
     <H3>$MESSAGE </H3> <br>
     <hr size=1>
     <font size=4>
          <A href="https://base3.informatik.hu-berlin.de/pass.shtml">
             zurück zum Ändern des Domain-Passwortes</A> </font>
ERROR
 /usr/bin/cat /var/apache2/htdocs/end.html
}
```



```
if [ "X$REQUEST_METHOD" != "XPOST" ] ; then
  /usr/bin/cat <<EOF
Content-type: text/html
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
<html>
  <head>
    <meta http-equiv="refresh"
           content="0; URL=http://www2.informatik.hu-berlin.de/">
  </head>
    <body></body>
    </html>
EOF
    exit 1
fi

/usr/bin/cat > $TEMP

SUBDOMAIN=`/usr/bin/awk '-F&' '{ split($1,F,"="); print F[2]; }' $TEMP`
W1=`/usr/bin/awk '-F&' '{ split($2,F,"="); print F[2]; }' $TEMP`
/usr/bin/awk '-F&' '{split($3,F,"=");printf("%s",F[2]);}' $TEMP > $TEMP1
chmod og-rwx $TEMP1
W2=`/usr/bin/cat $TEMP1`
W3=`/usr/bin/awk '-F&' '{ split($4,F,"="); print F[2]; }' $TEMP`
W4=`/usr/bin/awk '-F&' '{ split($5,F,"="); print F[2]; }' $TEMP`
```

```
#
# Pruefen ob alle Felder ausgefuellt sind
#
if [ "x$W1" = "x" -o "x$W3" = "x" -o "x$W4" = "x" -o ! -s $TEMP1 ] ; then
    MESSAGE="Fehler: Ein Feld wurde nicht ausgefüllt."
    /usr/bin/rm -f $TEMP1
    /usr/bin/rm -f $TEMP
    sendmsg
    exit
fi
#
#  Zugriffsberechtigung pruefen
#
/opt/csw/bin/ldapsearch -x -D \
       uid=$W1,ou=People,ou=$SUBDOMAIN,dc=informatik,dc=hu-berlin,dc=de \
       -y $TEMP1 -b ou=People,ou=$SUBDOMAIN,dc=informatik,dc=hu-berlin,dc=de
       uid=$W1  uid userPassword 2>/dev/null | \
       /usr/bin/awk '/^uid:/{ print $2; } \
               /^userPassword::/{ print $2; }' >$TEMP 2>/dev/null
R1=`/usr/bin/head -1 $TEMP`
R2=`/usr/bin/tail -1 $TEMP`
/usr/bin/rm -f $TEMP1
/usr/bin/rm -f $TEMP
if [ "x$R1" != "x$W1" -o "x$R2" = "x" ] ; then
   SLEEP=5
   MESSAGE="Fehler: Nutzeridentifikation fehlgeschlagen - falscher Nutzer"
   sendmsg
   exit
fi
```



```
if [ "x$W3" != "x$W4" ] ; then
    MESSAGE="Fehler: Eingaben neues Passwort nicht identisch)."
    sendmsg
    exit
fi
if [ "x$W2" = "x$W3" ] ; then
    MESSAGE="Fehler: Neues Passwort ist identisch mit dem alten Passwort."
    sendmsg
    exit
fi
#
# Komplexitaet des Passwortes pruefen
#
/opt/csw/bin/gawk '{ if ( $1 ~ "[a-zA-Z]" )  RET=1;  \
                     if ( $1 ~ "[0-9]"    )  RET=RET+1; \
                     if  ($1 ~ "[^0-9a-zA-Z\n]" ) RET=RET+1;   \
                     if  ( length($1) >= 7 )      RET=RET+1; } \
                   END { exit (RET) } ' <<EOF
$W3
EOF
RES=$?
if [ $RES != 4 ] ; then
    MESSAGE="Fehler: Eingegebenes neues Passwort zu einfach!! \
           <br>        Mindestens 7 Zeichen, mit Buchstaben, \
                       Ziffern und Sonderzeichen"
    sendmsg
    exit
fi
```

```
#
#  Nutzer beruhigen - Erfolgsmeldung ausgeben
#
MESSAGE="Passwort für Nutzer '$W1' zur Änderung vorgemerkt. \
         <br> Änderung erfolgt spätestens in einer Stunde."
sendmsg
#
# Informationen verschlüsseln und zur Abholung bereitstellen
#
/usr/bin/cat <<EOF  | \
     /opt/csw/bin/gpg -r ldaps -e >$SUBMIT/$W1.gpg 2>/dev/null
$W1 $SUBDOMAIN $W1 $W3 $R2
EOF
chmod og-rwx $SUBMIT/$W1.gpg
exit
```


Folgende kleine Programme/Werkzeuge wurden benutzt:

- **sh – Bourne-Shell – eine Scriptsprache**
- **rm – File löschen**
- **cat – Kopieren eines Textes von der Standardeingabe in ein File**
- **echo – Ausgabe eines Textes auf die Standarausgabe**
- **head – Anzeigen des Anfanges eines Files**
- **tail – Anzeigen des Ende eines Files**
- **awk, gawk – Kleiner Interpreter zur Verarbeitung von Textfiles**
- **gpg – Verschlüsseln eines Textes**

Damit ist auf dem WWW-Server alles fertig. Die Daten liegen gut verschlüsselt in einem Directory zur Abholung bereit. Sollte der WWW-Server gehackt werden, kann der Hacker mit den Daten nichts anfangen.

Auf dem Account-Server müssen jetzt zur gegebenen Zeit die Daten von dem WWW-Server abgeholt werden und den einzelnen Passwort-Datenbanken zugeführt werden.

Dies geschieht durch das Script "get-passwd", das alle Stunde durch den "CRON" aufgerufen wird. Das Script realisiert dann folgende Dienste:

Transport der verschlüsselten Informationen vom WWW-Server zum Account-Server

Löschen der Informationen auf dem WWW-Server

Auspacken der Informationen

Das Shell-Script: get-passwd

```
#!/bin/sh
BASE=/etc/YP/Passwd
cd $BASE/New
# holen von Nutzeraenderungen
LS=`/usr/bin/ssh  root@base3  /usr/bin/ls '/var/submit/*.gpg' 2>/dev/null`
ERROR=0
if [ "x$LS" != "x" ] ; then
     /usr/bin/scp 'root@base3:/var/submit/*' $BASE/New
     for i in $LS
       do
         NAME=`basename $i`
         if [ ! -s $BASE/New/$NAME ] ; then
            echo "get-pass: Transportfehler bei $i"
            ERROR=1
         fi
     done
     /usr/bin/chmod og-rwx *
     if [ $ERROR -eq 0 ] ; then
        /usr/bin/ssh root@base3 /usr/bin/rm '/var/submit/*.gpg'
     else
        echo "get-pass: kein rm auf base3, da Transportfehler"
     fi
fi
# Aenderung ausfuehren
$BASE/change-passwd
```

```
Das Shell-Script:       change-passwd

#!/bin/sh
# Uebernahme der Passwoerter in LDAP, NIS und Windows
#
# Programm-Basis
BASE=/etc/YP/Passwd
#
# Quelle
SUBMIT=$BASE/New
#
# NIS-Datenbasis
NISBASE=/etc/YP
#
#   nur benutzt wenn Windows nicht reagiert
WINDOWSPASSWDSAVE=$BASE/Windows
#
# LDAP-Datenbasis-Account
ACCESS="-x -D cn=Manager,dc=informatik,dc=hu-berlin,dc=de -w "
ALLSUBDOMAINS="all alkox sar"
#
```



```
#
# Passwort an LDAP uebergeben
#
modify_ldap()
{
  echo "Try modify Password in LDAP for user '$USER' in Domain '$SUBDOMAIN'"
  RES=`/opt/csw/bin/ldapsearch -x \
         -b ou=People,ou=$SUBDOMAIN,dc=informatik,dc=hu-berlin,dc=de \
          uid=$USER uid | /usr/bin/grep "uid:"`
    if [ "x$RES" = "x" ] ; then
          echo "LDAP: User '$USER' not in Domain '$SUBDOMAIN'\n"
          return
    fi

    /opt/csw/bin/ldapmodify $ACCESS  `cat $BASE/Secure/ldap` <<EOF
dn: uid=$USER,ou=People,ou=$SUBDOMAIN,dc=informatik,dc=hu-berlin,dc=de
changetype: modify
replace: shadowLastChange
shadowLastChange: $AKTDATE
-
replace: userPassword
userPassword: {crypt}$NEWCRYPT
EOF
    return
}
```

```
#
# Passwort an NIS uebergeben
# kann spaeter gestrichen werden
#
modify_nis()
{
  echo "Try modify Password in NIS for user '$USER'"
  ONISPAS=`awk -F: "/^$USER:/{ print \\$2; }" $NISBASE/passwd`
  if [ "x$ONISPAS" != "x" ] ; then
    # Uebernahme in NIS-Datenbaseis
    /opt/csw/bin/gsed -iold -e "/^$USER:/s;${ONISPAS};${NEWCRYPT};" \
                      $NISBASE/passwd
    chmod og-rwx $NISBASE/passwd $NISBASE/passwd.old
    echo "NIS: For User '$USER' password changed."
  else
    echo "NIS: User  '$USER'  nicht in NIS-Datenbasis"
  fi
  return
}
```



```
#
# Passwort an Windows uebergeben
# Passwort wird gemerkt, wenn Windows mal wieder nicht will!!!!!
#
modify_windows()
{
    echo "Try modify Password in windows for user '$USER'"
    /usr/bin/ssh -l root spree cscript "//NoLogo" pw.vbs  <<EOF
$USER
$UNCPASS
EOF
    RES=$?
    if [ $RES -eq 0 ] ; then
        echo "Windows: For User '$USER' password changed."
        return
    fi
    cp $SUBMIT/$USERGPG $WINDOWSPASSWDSAVE/$USERPG
    chmod og-rwx $WINDOWSPASSWDSAVE/$USERPG
    if [ $RES -eq 6 ] ; then
        echo "Warnung!! User '$USER' Passwort fuer Windows falsch"
        return
    fi
    echo "Windows!! User '$USER'  nicht in Windows bekannt."
    return
}
```

```
cd $SUBMIT
for USERGPG in *
  do
    if [ "x$USERGPG" = 'x*' ] ; then
       continue
    fi
    USER=`echo $USERGPG | /usr/bin/sed "s/\.gpg//"`
    # Klartextpassword Crypten, dazu mit gpg entschluesseln
    UNC=`cat $USERGPG | \
         /opt/csw/bin/gpg -q --batch --passphrase-file $BASE/Secure/gpg -d`
    U=`/usr/bin/cat <<EOF | awk '{ print $3 }'
$UNC
EOF`
    if [ "x$U" != "x$USER" ] ; then
       echo "Warnung!! Aenderungssatz $USER.gpg nicht fuer User '$USER' \
                                           - Passwort nicht veraendert"
       unset UNC
       continue
    fi
    SUBDOMAIN=`/usr/bin/cat <<EOF | awk '{ print $2 }'
$UNC
EOF`
```



```
    RES=`/opt/csw/bin/ldapsearch -x \
           -b ou=People,ou=$SUBDOMAIN,dc=informatik,dc=hu-berlin,dc=de \
           uid=$USER uid | /usr/bin/grep "uid:"`
    if [ "x$RES" = "x" ] ; then
       echo "Warnung!! User '$USER' nicht in LDAP-Datenbasis -\
               Passwort nicht veraendert"
       unset UNC
       continue
    fi
    # pruefen des alten Passwortes mit dem uebergebenen alten Passwort
    /opt/csw/bin/ldapsearch $ACCESS \
          `cat $BASE/Secure/ldap` \
          -b ou=People,ou=$SUBDOMAIN,dc=informatik,dc=hu-berlin,dc=de \
           uid=$USER userPassword | \
           awk '/^userPassword::/{ print $2; }' >/tmp/asdf$$
    LNPA=`cat /tmp/asdf$$`
    rm -f /tmp/asdf$$
    NPA=`/usr/bin/cat <<EOF | awk '{ print $5 }'
$UNC
EOF`
    if [ "x$NPA" != "x$LNPA" ] ; then
       echo "Veralteter Aenderungssatz fuer User '$USER'  \
                    - Passwort nicht veraendert."
       unset UNC
       continue
    fi
```

```
    # Aenderungsdatum
    AKTDATE=`/opt/csw/bin/gdate +%s`
    UNCPASS=`/usr/bin/cat <<EOF | awk '{ print $4 }'
$UNC
EOF`
    unset UNC
    export UNCPASS
    NEWCRYPT=`$BASE/Bin/passcrypt`

    # Uebernahme in LDAP in alle SUBDOMAIN
    for SUBDOMAIN in $ALLSUBDOMAINS
      do
       modify_ldap
    done
    # Uebernahme in nis
    modify_nis
    echo
    # Uebernahme in windows
    modify_windows
    #
    unset UNCPASS
    unset NEWCRYPT
    # löschen der übermittelten Daten
    rm $SUBMIT/$USERGPG
done
unset ACCESS


---------------------------------------
```


Kleines C–Programm, da es unter SOLARIS nicht das passende Werkzeug zum Erzeugen eines gecrypteten Passwortes gibt.

passcrypt.c

```
#include <stdio.h>
#include <unistd.h>
#include <stdlib.h>

char chars[]="abcdefghijklmnopqrstuvwxyz0123456789ABCDEFGHIJKLMNOPQRSTUVWXYZ"

main(int argc,char *argv[])
{
  char *pass,ch[2];

  pass=NULL;
  if ( argc == 2 ) {
     pass=argv[1];
  }
  if ( NULL != getenv("UNCPASS"))
      if ( strcmp("UNCPASS",getenv("UNCPASS"))) pass=getenv("UNCPASS");
  if ( pass == NULL ) exit(1);
  srand(time(NULL)%getpid());
  ch[0]=chars[rand()%62];
  ch[1]=chars[rand()%62];
  printf("%s",crypt(pass,ch));
  exit(0);
}
```

```
4. Beispiel
-----------

Überwachung von Rechnern via Netzwerk

Ziel: Prüfen ob ein Rechner nocht im Netz ist (ping), Prüfen ob sich die
      MAC-Adresse eines existierenden Rechners geändert hat.
      Dadurch wird der Tausch von Rechnern erschwert.

Notwendige Daten:
      Nutzer, die sich den Rechner vor Ort anschauen können
      Liste von zu überwachenden Rechner
      Liste der MAC-Adressen der zu überwachenden Rechner

Arbeitsweise:
      Rechner anpingen
      MAC-Adresse im ARP-Stack auslesen
      MAC-Adresse prüfen
      eventuell Nachricht an anwesende Person versenden

Probleme:
      Script möglichst nur einmal starten
      Fehlerhafte Rechner nur einmal melden
```



```
Liste der ständigen Administratoren: etc/users
   baerwolf
   kaempfer
   wozobule
   bell


Liste der MAC-Adressen:  etc/ethers
   adler      8:0:20:22:41:bc
   alex       0:1:2:1e:d5:45
   ampere     0:0:c0:99:13:29
   amsel      0:3:ba:a:e2:38
   aqua       0:3:93:3:60:1a
   at286      0:a0:24:c1:74:59
   beat       8:0:20:93:b7:98
   spock      0:3:ba:9:5d:b9
   staaken    0:1:2:1e:9f:f4
   star       0:3:ba:1d:7f:2
   zentusv    0:80:c8:aa:17:a8

Liste der zu überwachenden Rechner:  etc/pools
   #
   # Linux-Pool     III.2.12    19 Maschinen
   #
   marzahn  III.2.12-ML4
   tegel    III.2.12-ML4
   rudow    III.2.12-ML3
   dahlem   III.2.12-ML2
```

```
Script, das die MAC-Adressen überwacht: bin/ethers-check
 in crontab:
   0,10,20,30,40,50 * * * * /usr/sbin/check/bin/ethers-check

#!/bin/sh
USERS=/usr/sbin/check/etc/users
HOSTLIST=/usr/sbin/check/etc/pools
ETHERS=/usr/sbin/check/etc/ethers
LOCK=/usr/sbin/check/lock
MYLOCK=/usr/sbin/check/lock/ethers-check
LOG=/usr/sbin/check/log/ethers
if [ -f ${MYLOCK} ] ; then
      exit 0
fi
/bin/touch ${MYLOCK}
for HOST in `/bin/awk '{ print $1 }' $HOSTLIST`
do
  if [ \! -f ${LOCK}/$HOST  -a "$HOST" != "#" ] ; then
    /bin/touch ${LOCK}/$HOST
    /usr/sbin/ping $HOST 10 >/dev/null
    RET=$?
    if [ $RET -eq 0 ] ; then
      IARP=`/usr/sbin/arp $HOST | /bin/awk '{ print $4 ; }'`
      SARP=`/bin/grep "^$HOST\>" $ETHERS | /bin/awk '{ print $2 }' `
      if [ "x$IARP" = "xno" ] ; then
         /bin/rm -f ${LOCK}/$HOST
         continue
      fi
```



```
      if [ "x$IARP" != "x$SARP"  ] ; then
        echo Fehler IARP: $IARP   SARP $SARP
        RAUM=`/bin/grep "^$HOST\>" $HOSTLIST | /bin/awk '{ print $2 }' `
        for u in `/bin/cat $USERS`
          do
            /usr/ucb/mail -s "Achtung: Rechner $HOST in Raum $RAUM \
                                       manipuliert" $u  <<EOF
            Achtung!!!
            Die MAC-Adresse des Rechners $HOST hat sich geaendert.
                   gefundene MAC-Adresse: $IARP
                   soll MAC-Adresse:      $SARP
            Standorte der Rechners:
                      $HOST       Raum: ${RAUM}
            Danke, Der elektronische Diensthabende
            PS: Systemadministrator knecht:
                Datei knecht:${LOCK}/$HOST streichen, wenn Schaden behoben!!!
EOF
        done
        /bin/echo "`date`: \c" >>$LOG
        /bin/echo  $HOST in Raum $RAUM manipuiliert >>$LOG
      else
        /bin/rm -f ${LOCK}/$HOST
      fi
    else
        /bin/echo "`date`: \c" >>$LOG
        /bin/echo  $HOST in Raum $RAUM reagiert nicht auf ping >>$LOG
    fi
  fi
done
/bin/rm -f ${MYLOCK}
```