!!! ACHTUNG - evtl. veraltet - ACHTUNG !!!

Diese Seite wurde zuletzt am 9. Juli 2014 um 10:53 Uhr geändert.

In der **Forensik** sind *die einen* der Meinung, man sollt immer ein paar „static binaries“ dabei haben…

die anderen meinen, den Aufwand könne man sich sparen, da man vorab eh nicht alle Eventualitäten (32/64-Bit, alter/neuer/embedded-CPU, alter/neuer/spezieller Kernel,…) abdecken kann und im Fall der Fälle eben diese eine dann fehlt.

Ich vermische beides einfach mal noch ein wenig mit der Sichtweise eines **Administrators** und bin der Meinung, man sollte…

- für seine eigenen Systeme immer welche zur Hand haben
- sich per Skript auch mal schnell aktuelle bauen können
- und im Fall der Forensik versuchen, vorher ein wenig (mehr) Aufklärung zu betreiben und passende bauen
- außerdem: was man hat, das hat man!

Hier also die Skripte (work in progress), die ich benutze:

busybox

```
 #!/bin/bash
 RELEASE="1.20.2"
 # dependencies (static libs):
 if [[|! -f /usr/lib/libc.a ]] ; then sudo aptitude install libc6-dev -y ;
fi
 # downloading...
 wget http://busybox.net/downloads/busybox-${RELEASE}.tar.bz2
 # extractin...
 tar xjf busybox-${RELEASE}.tar.bz2
 # compiling (and stripping)...
 cd busybox-${RELEASE}
 make defconfig
 sed -e 's/.*FEATURE_PREFER_APPLETS.*/CONFIG_FEATURE_PREFER_APPLETS=y/' -i
.config
 sed -e 's/.*FEATURE_SH_STANDALONE.*/CONFIG_FEATURE_SH_STANDALONE=y/' -i
.config
 sed -e 's/.*CONFIG_STATIC.*/CONFIG_STATIC=y/' -i .config
 make
 # final instructions:
 echo
 echo "Fertig:"
 echo "  PATH= $PWD/busybox sh"
 echo
 exit 0
```

Zuletzt getestet unter Debian GNU/Linux 6.0.7 (Squeeze) i386 & amd64.

libewf

```
 #!/bin/bash
 RELEASE="20130303"
 # dependencies (static libs):
 if [[|! -f /usr/lib/libc.a ]] ; then sudo aptitude install libc6-dev -y ;
fi
 if [[|! -f /usr/lib/libz.a ]] ; then sudo aptitude install zlib1g-dev -y ;
fi
 if [[|! -f /usr/lib/libssl.a ]] ; then sudo aptitude install libssl-dev -y
; fi
 if [[|! -f /usr/lib/libuuid.a ]] ; then sudo aptitude install uuid-dev -y ;
fi
 if [[|! -f /usr/lib/libfuse.a ]] ; then sudo aptitude install libfuse-dev -
y ; fi
 # downloading...
 wget http://libewf.googlecode.com/files/libewf-${RELEASE}.tar.gz
 # extracting...
 tar xzf libewf-${RELEASE}.tar.gz
 # compiling and stripping...
 cd libewf-${RELEASE}
 ./configure --enable-shared=no --enable-static-executables=yes
 make
 strip ewftools/ewfacquire
 # final instructions:
 echo
 echo "Das statisch kompilierte ewfacquire (und ein wenig mehr) findest Du
hier:"
 echo "  $PWD/ewftools/ewfacquire"
 echo
 echo "(Nicht vergessen: you have to be root!)"
 echo
 exit 0
```

Zuletzt getestet unter Debian GNU/Linux 6.0.7 (Squeeze) i386 & amd64.

dropbear

```
 #!/bin/bash
 RELEASE="2012.55"
 # dependencies (static libs):
 if [[|! -f /usr/lib/libc.a ]] ; then sudo aptitude install libc6-dev -y ;
fi
 # downloading...
 wget https://matt.ucc.asn.au/dropbear/releases/dropbear-${RELEASE}.tar.bz2
 # extracting...
 tar xjf dropbear-${RELEASE}.tar.bz2
 # compiling and stripping...
```

```
cd dropbear-${RELEASE}
./configure
make STATIC=1
make strip
# final instructions:
echo
echo "Fertig:"
echo "  $PWD/dcclient"
echo "  $PWD/dropbear"
echo "  (und ein bisschen mehr...)
echo
exit 0
```

Zuletzt getestet unter Debian GNU/Linux 6.0.7 (Squeeze) i386 & amd64.

bash

```
#!/bin/bash
if [[|! -x /usr/bin/yacc ]] ; then sudo aptitude install bison -y ; fi
if [[|! -x /usr/bin/flex ]] ; then sudo aptitude install flex -y ; fi
RELEASE="4.2"
LASTPATCH="42"
# downloaden
wget ftp://ftp.gnu.org/gnu/bash/bash-${RELEASE}.tar.gz
PATCH="1"
while [[|$PATCH -le $LASTPATCH ]] ; do
  if [[|$PATCH -ge 1 -a $PATCH -le 9 ]] ; then PNIFN="00$PATCH" ;
  elif [[|$PATCH -ge 10 -a $PATCH -le 99 ]] ; then PNIFN="0$PATCH" ;
  else PNIFN="$PATCH" ; fi
  wget ftp://ftp.gnu.org/gnu/bash/bash-${RELEASE}-patches/bash$( echo
$RELEASE | tr -d "." )-$PNIFN
  (( PATCH++ ))
done
# entpacken
tar xvzf bash-${RELEASE}.tar.gz
# patchen
cd bash-${RELEASE}
PATCH="1"
while [[|$PATCH -le $LASTPATCH ]] ; do
  if [[|$PATCH -ge 1 -a $PATCH -le 9 ]] ; then PNIFN="00$PATCH" ;
  elif [[|$PATCH -ge 10 -a $PATCH -le 99 ]] ; then PNIFN="0$PATCH" ;
  else PNIFN="$PATCH" ; fi
  patch -p0 < ../bash$( echo $RELEASE | tr -d "." )-$PNIFN
  (( PATCH++ ))
done
# kompilieren
export CFLAGS="-static"
export CXXFLAGS="-static -static-libgcc -static-libstdc++"
export LDFLAGS="-static"
```

```
./configure --without-bash-malloc --enable-static-link
make
# kleinkram
for BINARY in bash; do
  strip "$BINARY"
  chmod +x "$BINARY"
done
# vorerst per hand an die richtige stelle kopieren...
exit 0
```

Zuletzt getestet unter Debian GNU/Linux 6.0.7 (Squeeze) amd64.

coreutils

```
#!/bin/bash
RELEASE="8.9"
# downloaden
wget ftp://ftp.gnu.org/gnu/coreutils/coreutils-${RELEASE}.tar.gz
# entpacken
tar xvzf coreutils-${RELEASE}.tar.gz
# kompilieren
cd coreutils-${RELEASE}
sed -i '/elf_sys=yes/s:yes:no:' configure
# das hostname aus den veralteten net-utils ist maechtiger, also ohne!
export CFLAGS="-static"
export CXXFLAGS="-static -static-libgcc -static-libstdc++"
export LDFLAGS="-static"
./configure --without-selinux
make
# kleinkram
for BINARY in base64 cat cp date dd df du echo id head ls md5sum printenv
pwd sha1sum sha256sum tail test tee tr true false uname uptime who basename
chgrp chmod chown cut env kill  mkfifo mknod mkdir mv ln rm rmdir sort stty
su sum sync touch tty uniq wc; do
  strip src/"$BINARY"
  chmod +x src/"$BINARY"
done
# vorerst per hand an die richtige stelle kopieren...
exit 0
```

Zuletzt getestet unter Debian GNU/Linux 6.0.7 (Squeeze) amd64.

grep

```
#!/bin/bash
# getestet mit:
#RELEASE="2.7"
```

```
RELEASE="2.7"
# downloaden
wget ftp://ftp.gnu.org/gnu/grep/grep-${RELEASE}.tar.gz
# entpacken
tar xvzf grep-${RELEASE}.tar.gz
# kompilieren
cd grep-${RELEASE}
env LDFLAGS=-static ./configure
make
# kleinkram
for BINARY in grep; do
  strip src/"$BINARY"
  chmod +x src/"$BINARY"
done
# vorerst per hand an die richtige stelle kopieren...
exit 0
```

iproute2

```
#!/bin/bash
# getestet mit:
#RELEASE="2.6.35"
RELEASE="2.6.35"
# abhaengigkeiten pruefen und ggf. installieren
if [[|! -x /usr/bin/flex ]] ; then sudo aptitude install flex -y ; fi
if [[|! -r /usr/include/db_185.h ]] ; then sudo aptitude install libdb-dev
-y ; fi
# downloaden
wget
http://devresources.linuxfoundation.org/dev/iproute2/download/iproute2-${REL
EASE}.tar.bz2
# entpacken
tar xvjf iproute2-${RELEASE}.tar.bz2
# kompilieren
cd iproute2-${RELEASE}
env LDFLAGS=-static make
# kleinkram
for BINARY in ip/ip misc/ss; do
  strip "$BINARY"
  chmod +x "$BINARY"
done
# vorerst per hand an die richtige stelle kopieren...
exit 0
```

lsof

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

```
# getestet mit:
#RELEASE="4.84"
RELEASE="4.84"
# downloaden
wget ftp://ftp.fu-berlin.de/pub/unix/tools/lsof/lsof_${RELEASE}.tar.bz2
# entpacken
tar xvjf lsof_${RELEASE}.tar.bz2
cd lsof_${RELEASE}
tar xvf lsof_${RELEASE}_src.tar
# kompilieren
cd lsof_${RELEASE}_src
./Configure -n linux
make "$( grep ^CFGL= Makefile ) -static"
# kleinkram
for BINARY in lsof; do
  strip "$BINARY"
  chmod +x "$BINARY"
done
# vorerst per hand an die richtige stelle kopieren...
exit 0
```

netcat

```
#!/bin/bash
# es wird davon ausgegangen, dass build-essential bereits installiert
wurde!
if [[|! -x /usr/bin/svn ]] ; then sudo aptitude install subversion -y ; fi
# checkout
svn co https://nc110.svn.sourceforge.net/svnroot/nc110 nc110
cd nc110/nc110
# "patchen", damit es fehlerfrei kompiliert werden kann:
sed -i "s/LD = \$(CC) -s/LD = \$(CC)/" Makefile
sed -i '/#include <fcntl.h>/ a\#include "resolv.h"' netcat.c
# kompilieren
export CFLAGS="-static"
export CXXFLAGS="-static -static-libgcc -static-libstdc++"
export LDLAGS="-static"
export DFLAGS="-DGAPING_SECURITY_HOLE"
make linux
# kleinkram
for BINARY in nc ; do
  strip "$BINARY"
  chmod +x "$BINARY"
done
# vorerst per hand an die richtige stelle kopieren...
exit 0
```

Zuletzt getestet unter Debian GNU/Linux 6.0.7 (Squeeze) amd64.

Besonderer Augenmerk liegt auf der Option „-e“, welche zu Schulungszwecken als „Backdoor für Arme“ genutzt werden kann!

net-tools

```
[[...]]
```

pciutils

```
#!/bin/bash
# getestet mit:
#RELEASE="3.1.7"
RELEASE="3.1.7"
# downloaden
wget
ftp://atrey.karlin.mff.cuni.cz/pub/linux/pci/pciutils-${RELEASE}.tar.gz
# entpacken
tar xvzf pciutils-${RELEASE}.tar.gz
# kompilieren
cd pciutils-${RELEASE}
env LDFLAGS=-static make ZLIB=no
# kleinkram
for BINARY in lspci; do
  strip "$BINARY"
  chmod +x "$BINARY"
done
# vorerst per hand an die richtige stelle kopieren...
exit 0
```

procps

```
#!/bin/bash
# getestet mit:
#RELEASE="3.2.8"
RELEASE="3.2.8"
# downloaden
wget http://procps.sourceforge.net/procps-${RELEASE}.tar.gz
# entpacken
tar xvzf procps-${RELEASE}.tar.gz
# kompilieren
cd procps-${RELEASE}
make LDFLAGS=-static SHARED=0
# kleinkram
for BINARY in ps/ps sysctl w free top uptime watch vmstat; do
  strip "$BINARY"
```

```
  chmod +x "$BINARY"
done
# vorerst per hand an die richtige stelle kopieren...
exit 0
```

sharutils

```
#!/bin/bash
# getestet mit:
#RELEASE="4.10"
RELEASE="4.10"
# downloaden
wget ftp://ftp.gnu.org/gnu/sharutils/sharutils-${RELEASE}.tar.bz2
# entpacken
tar xvjf sharutils-${RELEASE}.tar.bz2
# kompilieren
cd sharutils-${RELEASE}
env LDFLAGS=-static ./configure
make
# kleinkram
for BINARY in uuencode uudecode; do
  strip src/"$BINARY"
  chmod +x src/"$BINARY"
done
# vorerst per hand an die richtige stelle kopieren...
exit 0
```

sysvinit

```
#!/bin/bash
# getestet mit:
#RELEASE="2.88dsf"
RELEASE="2.88dsf"
# downloaden
wget
http://download.savannah.gnu.org/releases/sysvinit/sysvinit-${RELEASE}.tar.b
z2
# entpacken
tar xvjf sysvinit-${RELEASE}.tar.bz2
# kompilieren
cd sysvinit-${RELEASE}/src
### nur last, wer mehr braucht, kompiliert alles
###  ... strippen und chmodden danach aber fuer den
###  rest nicht vergessen!
env LDFLAGS=-static make last
# kleinkram
for BINARY in last; do
```

```
  strip "$BINARY"
  chmod +x "$BINARY"
done
# vorerst per hand an die richtige stelle kopieren...
exit 0
```

usbutils

```
#!/bin/bash
# getestet mit:
#RELEASE="0.86"
RELEASE="0.86"
# abhaengigkeiten
if ! $( dpkg -l | grep libusb-dev > /dev/null ) ; then aptitude install
libusb-dev -y ; fi
# downloaden
wget
http://netcologne.dl.sourceforge.net/project/linux-usb/usbutils/usbutils-${R
ELEASE}.tar.gz
# entpacken
tar xvzf usbutils-${RELEASE}.tar.gz
# kompilieren
cd usbutils-${RELEASE}
env LDFLAGS="-static" ./configure
sed -i 's/${prefix}\/share/./' Makefile
make
# kleinkram
for BINARY in lsusb ; do
  strip "$BINARY"
  chmod +x "$BINARY"
done
# vorerst per hand an die richtige stelle kopieren...
# usb.ids nicht vergessen (selbe verzeichnis wie lsusb)!!!
exit 0
```

util-linux

```
[[...]]
```

From:
http://wiki.neumannsland.de/ - **Patricks DokuWiki**

Permanent link:
http://wiki.neumannsland.de/mw2dw:ds3000-static-binaries

Last update: **2019/09/23 14:44**