ISEC Übung 1

Shell Usage

Exercise 1

The given text is saved in original.txt. It is then read and piped to tr which exchanges all M to a and all F to j. The output of tr is saved to aufg1.txt.

```
cat original.txt | tr "MF" "aj" > aufg1.txt
```

Exercise 2

Analog zu 1 wird nun der Text aus aufg1.txt in tr gepiped. Die Buchstaben A-Z werden dabei zu L-Z bzw. A-L umgewandelt – tr erkennt dass es beide Ausgabe-Ranges benutzen muss um die erste Eingaberange (A-Z) komplett umwandeln zu können. Analog wird dies für die Kleinbuchstaben gemacht, die bei dieser Gelegenheit in Großbuchstaben umgewandelt werden.

The result of Ex.1 is piped to tr again, this time exchanging the whole range from A to Z to the letters L to Z and A to K, thus applying ROT11. Tr automatically realizes it needs to use two short ranges to cover one large input range. The second range does so likewise with lowercase letters, while transforming them to uppercase.

```
cat aufg1.txt | tr "[A-Za-z]" "[L-ZA-KL-ZA-K]" > aufg2.txt
```

Scripting

Exercise 1

```
#!/bin/bash

# check if command line argument given
if [ -z "$1" ]
then
        echo "usage: $0 url"
        exit 1
fi

echo "Fetching $1 ..."
# -o to find actual matches rather than lines with a match
# -e to use a regular expression
# -P to use Perl regex syntax for  using non-greedy modifiers to make sure
#          every a tag is matched on its own

# save list of links to a variable
links=$(curl -Ls $1 | grep -o -P -e '<a.*?href=\".*?\".*?>(.*?)</a>')

# count those links using grep by counting all lines
# kind of a workaround, but as wc -l is not allowed and using
# the -c flag on grep in the first place doesn't work as it counts
# lines containing on or more matches rather than the total number of
# matches, even with -o set
no=$(echo "$links" | grep -c 'a')
echo "Found $no links!"

# a list of all link tags are is still saved and can be used for further purposes
```

Exercise 2

Reads the User ID via command line so each user can fetch their personal page. A default user can also be saved in DEFAULTGET, but is anonymized for this exercise.

```
#!/bin/bash

STUB="https://hpi-vdb.de/vulndb/isec_task/?uid="
DEFAULTGET="x.x"

if [ -n "$1" ]
# test whether command-line argument is present
then
  GET=$1
else
  GET=$DEFAULTGET # Default, if not specified on command-line.
fi


# fetch all text from personal website
# -s suppresses statistics, so only the website's content is saved
# then filter out everything that looks like a uid: firstname.lastname
# assumption: first names are at least 2, last names at least 3 chars long
# and uids are only composed of lowercase latin letters
curl -s "$STUB$GET" | grep -o -P -e '[a-z]{2,}\.[a-z]{3,}'
```

Virtual Machines

Exercise 1

NAT

When using NAT = Network Address Translation, VirtualBox serves as a router between one or more VMs and the host PC, just like in a common setup where a router is set up between multiple computers in a home network and the internet. In the default configuration, all incoming traffic to the host is left as-is, and only replies to outgoing packets of a VM are forwarded back to the VM.

In the process, the router “translates” the packet IPs between the real network and the virtual one, e.g. all packets send by VMs have the host IP as sender.

VirtualBox allows to forward ports, in that case forwarding packets arriving on a specified port of the host to a specified port on the VM, thus allowing VMs to listen to incoming traffic as server applications need to.

NAT Network

Similar to simple NAT, though additionally, so-called NAT networks can be created that enable VMs to communicate with each other. Communication with the host and the outside world stay as with normal NAT.

Bridged Networking

With Bridged Networking, a virtual network interface is created on the host, allowing the VM to be part of the actual network the host is connected to. Each VM therefore gets their own IP in the host’s network.

Internal Networking

Sets up a network only between VMs, a VM cannot communicate with the host, let alone the outside network. Multiple internal networks can be set up to only allow specific VMs to communicate with each other.

Host-Only Networking

Works similarly to Bridged Networking, but doesn’t allow VMs to connect to the outside network – only to each other and the host.

UDP Tunnel Networking

Allows to set up a “tunnel” between to VMs located on different hosts by encapsulating all packages to a specific destination in UDP packages, which are then send to the other host on a specific port, unwrapped, and forwarded to a specific VM.

Generic Driver

Allows the user to choose a network driver, functionality depends on the selected driver. Is used as kind of an API for plugins, such as VDE (see below)

VDE Networking

Only available with the VDE plugin, VDE allows setting up a complex virtual network between multiple VMs and hosts, using virtual network components such as switches and wires.