Lab 3: Shell scripting and systemd -

Parth Kalkar

Q1. Create a simple script to print odd numbers from 1 to 10.

- This is the following proposed solution

```
echo "enter n value as range to calculate odd"
read n
i=1
while [ $i -le $n ]
do
if [ `expr $i % 2`-ne 0 ]
then
echo odd=$i
fi
i=`expr $i + 1`
done
```

```
parth@GL62M-7RDX:~/Desktop/SNA$ sh num.sh
enter n value as range to calculate odd
10
odd=1
odd=3
odd=5
odd=7
odd=9
```

Q2. Create a shell script which makes new users and their corresponding passwords for 10 accounts in the system. Script should read the data from the file users.txt(you can write data to this file).

- We can use the following script to add new users directly from the command line
- No repeated user is allowed else it will show that the user already exists

```
i=1
while [ $i -le 10 ]
do
if [ $(id -u) -eq 0 ]; then
  read -p "Enter username : " username
  read -p "Enter password : " password
  egrep "^$username" /etc/passwd >/dev/null
  if [ $? -eq 0 ]; then
     echo "$username exists!"
     exit 1
  else
     pass=$(perl -e 'print crypt($ARGV[0], "password")' $password)
     useradd -m -p $pass $username
     [ $? -eq 0 ] && echo "User has been added to system!" || echo "Failed to add a user!"
  fi
else
  echo "Only root may add a user to the system"
  exit 2
fi
i=èxpr $i + 1`
done
```

```
parth@GL62M-7RDX:~/Desktop/SNA$ sudo ./pass.sh
[sudo] password for parth:
Enter username : q1
Enter password : 11
User has been added to system!
Enter username : q2
Enter password : 12
User has been added to system!
Enter username : q3
Enter password : 13
User has been added to system!
Enter username : q4
Enter password : 14
User has been added to system!
Enter username : q5
Enter password : 15
User has been added to system!
Enter username : q6
Enter password : 17
User has been added to system!
Enter username : q7
Enter password : 18
User has been added to system!
Enter username : q8
Enter password : 19
User has been added to system!
Enter username : q9
Enter password : 111
User has been added to system!
Enter username : q10
Enter password : 112
User has been added to system!
```

- To add user from a txt file with username as and data as input, we can do the following

```
filename='user.txt'
if [ $(id -u) -eq 0 ]; then
  n=1
  while read line; do
     echo $line
     username=$(echo $line | awk '{print $1;}')
     password=$(echo $line | awk '{print $2;}')
     egrep "^$username" /etc/passwd >/dev/null
```

```
    if [ $? -eq 0 ]; then
        echo "$username exists!"
        exit 1
    else
        pass=$(perl -e 'print crypt($ARGV[0], "password")' $password)
        useradd -m -p "$pass" "$username"
        [ $? -eq 0 ] && echo "User has been added to system!" || echo "Failed to add a
user!"
    fi
    done < $filename
else
  echo "Only root may add a user to the system."
  exit 2
fi
```

```
parth@GL62M-7RDX:~/Desktop/SNA$ sudo ./read_user.sh
p1 p[1]
User has been added to system!
p2 p[2]
User has been added to system!
p3 p[3]
User has been added to system!
p4 p[4]
User has been added to system!
p5 p[5]
User has been added to system!
p6 p[6]
User has been added to system!
p7 p[7]
User has been added to system!
p8 p[8]
User has been added to system!
p9 p[9]
User has been added to system!
p10 p[10]
User has been added to system!
```

User.txt has the following entries

p1 p[1]
p2 p[2]
p3 p[3]

p4 p[4]
p5 p[5]
p6 p[6]
p7 p[7]
p8 p[8]
p9 p[9]
p10 p[10]

Q3. Provide an example of the shell script where you can pass result (not an exit code) of the executed function in a subshell to the parent's shell

- sub-shell

```
#!/bin/bash
function retfunc(){
echo "Hello Parent"
}
```

- parent-shell

```
#!/bin/bash
set -e
source sub-shell.sh
output=$(retfunc)
echo $output
```

- Output

```
parth@GL62M-7RDX:~/Desktop/SNA$ sudo ./sub-shell.sh
parth@GL62M-7RDX:~/Desktop/SNA$ sudo ./parent-shell.sh
Hello Parent
```

Q4. Create a script that redirects system date and disk utilization in a file with systemd.

Steps:

- Create a script or executable that the service will manage. In our case it is test_service.sh
- Copy the script to /usr/bin and make it executable:

```
sudo cp test_service.sh /usr/bin/test_service.sh
```

```
sudo chmod +x /usr/bin/test_service.sh
```

- Create a Unit file to define a systemd service. In our case it is myservice.service
- Copy the unit file to `/etc/systemd/system` and give it permissions:

```
sudo cp myservice.service /etc/systemd/system/myservice.service
```

```
sudo chmod 644 /etc/systemd/system/myservice.service
```

- Once you have a unit file, you are ready to test the service:

```
sudo systemctl start myservice
```

- Check the status of the service:

```
sudo systemctl status myservice
```

- The service can be stopped or restarted using standard systemd commands:

```
sudo systemctl stop myservice
```

```
sudo systemctl restart myservice
```

- Finally, use the `enable` command to ensure that the service starts whenever the system boots:

```
sudo systemctl enable myservice
```

- Myservice.service has the following

```
[Unit]
Description=My disk monitoring service

[Service]
Type=simple
User=root
Group=root
TimeoutStartSec=0
Restart=on-failure
RestartSec=30s
ExecStart=/bin/bash /usr/bin/test_service.sh/
SyslogIdentifier=Diskutilization

[Install]
WantedBy=multi-user.target
```

- test-service.sh has the following

```
date >> /var/storage-monitor.txt
sudo du -sch / >> /var/storage-monitor.txt
```

- Output