Report for Project 1
COMP 7500 Advanced Operating systems

By using Oracle VM VirtualBox, I installed CentOS7 to work on project 1 but after finishing the project I couldn't transfer the files from virtual machine to host machine. Upon referring Piazza, the instructor mentioned we could use CentOS9, so I completed my project using CentOS9.

All the commands below work in root only. By using script command to create a typescript, I created a typescript (update.script) to log to check if the packages are updated. Here is the result: -

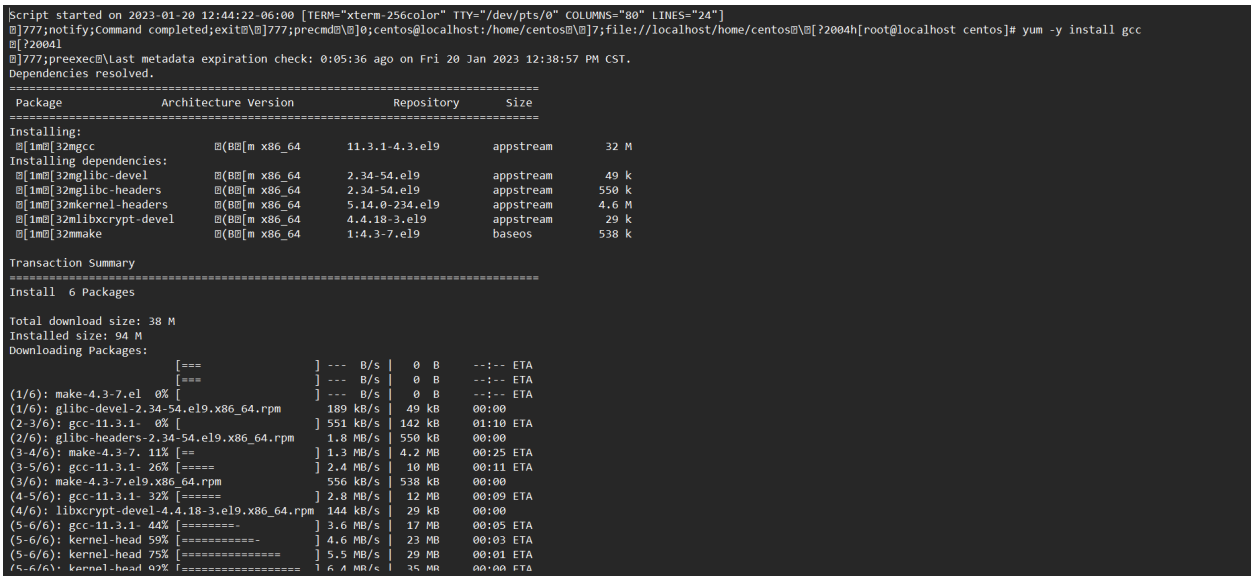
```
Script started on 2023-01-20 12:38:41-06:00 [TERM="xterm-256color" TTY="/dev/pts/0" COLUMNS="80" LINES="24"]
]777;notify;Command completed;\]777;precmd\]0;centos@localhost:/home/centos\]7;file://localhost/home/centos\[?2004h[root@localhost centos]# yum -y update
[?2004l
]777;preexec\CentOS Stream 9 - Ba     [===                ] ---  B/s |   0  B     --:-- ETA
CentOS Stream 9 - Ba     [===                ] ---  B/s |   0  B     --:-- ETA
CentOS Stream 9 - Ba     [   ===             ] ---  B/s |   0  B     --:-- ETA
CentOS Stream 9 - Ba     [      ===          ] ---  B/s |   0  B     --:-- ETA
CentOS Stream 9 - Ba 12% [==                 ] 2.4 MB/s | 759 kB     00:02 ETA
CentOS Stream 9 - Ba 88% [=================-  ] 3.2 MB/s | 5.4 MB     00:00 ETA
CentOS Stream 9 - BaseOS                       3.8 MB/s | 6.0 MB     00:01
CentOS Stream 9 - Ap     [          ===        ] ---  B/s |   0  B     --:-- ETA
CentOS Stream 9 - Ap 59% [==========-         ] 8.6 kB/s | 2.6 kB     00:00 ETA
CentOS Stream 9 - Ap  3% [-                  ] 120 kB/s | 565 kB     02:10 ETA
CentOS Stream 9 - Ap 34% [======-            ] 1.1 MB/s | 5.4 MB     00:09 ETA
CentOS Stream 9 - Ap 69% [=============-      ] 2.2 MB/s |  11 MB     00:02 ETA
CentOS Stream 9 - AppStream                     10 MB/s |  16 MB     00:01
CentOS Stream 9 - Ex     [            ===     ] ---  B/s |   0  B     --:-- ETA
CentOS Stream 9 - Ex100% [====================]  26 kB/s | 8.4 kB     00:00 ETA
CentOS Stream 9 - Ex  0% [                    ]  26 kB/s |   0  B     00:00 ETA
CentOS Stream 9 - Extras packages               14 kB/s |  10 kB     00:00
Dependencies resolved.
Nothing to do.
Complete!
]777;notify;Command completed;yum -y update\]777;precmd\]0;centos@localhost:/home/centos\]7;file://localhost/home/centos\[?2004h[root@localhost centos]# exit
[?2004l
]777;preexec\exit

Script done on 2023-01-20 12:39:14-06:00 [COMMAND_EXIT_CODE="0"]
```

Using script, I created a typescript (packages.script) to log all the packages that we need to install for the project. The installed packages are:-

yum -y install gcc
yum -y install gcc-c++
yum -y install vim-enhanced
yum -y install emacs
yum -y install gdb
yum -y install ethtool
yum -y install pciutils
yum -y install file

Here's the snippet of the log: -



Also scripted(system.script) information about CPU frequency, cache size, memory size, the list of PCI devices, hard drive, network MAC address and link speed, and the devices generating interrupts by using the commands: -

more /proc/cpuinfo
more /proc/meminfo
/sbin/lspci
/sbin/ethtool eth0
/sbin/ifconfig eth0
more /proc/interrupts

Here is the snap of the script: -

Using vim editor, created a c program (project1.c) which takes 10 numbers from the user and calculates the average of the square root of the array.

```c
#include<stdio.h>
#include<math.h>
int main()
{
    /* Function to take ten numbers from user and sum the square roots of the numbers and find the average of the square roots */
    int array[10];
    int max=10;
    double sum=0;
    double avg;
    int i;
    printf("\n Enter the 10 numbers:\n");
    for ( i=0;i<max;i++)
    {
    scanf("%d", &array[i]);
        }
    for(i=0;i<max;i++)
    {
    sum+=sqrt(array[i]);
    }
        avg=sum/10.0;
        printf("\n The average of the sum of the square roots =%f\n ", avg);
        return 0;
}
```

Further used gcc from GNU collection to compile and execute the c program which is logged in exec.script. Used ldd project1 command to get familiar with the libraries used for program execution which is logged in lib.script

```
Script started on 2023-01-20 13:00:40-06:00 [TERM="xterm-256color" TTY="/dev/pts/0" COLUMNS="80" LINES="24"]
]777;notify;Command completed;exit\]777;precmd\]0;centos@localhost:/home/centos\]7;file://localhost/home/centos\\[?2004h[root@localhost centos]# ldd p r oject1
[?2004l
]777;preexec\   linux-vdso.so.1 (0x00007ffdb65fe000)
      libm.so.6 => /lib64/libm.so.6 (0x00007f4f27680000)
      libc.so.6 => /lib64/libc.so.6 (0x00007f4f27400000)
      /lib64/ld-linux-x86-64.so.2 (0x00007f4f27767000)
]777;notify;Command completed;ldd project1\]777;precmd\]0;centos@localhost:/home/centos\]7;file://localhost/home/centos\\[?2004h[root@localhost centos]# exit
[?2004l
]777;preexec\exit

Script done on 2023-01-20 13:00:55-06:00 [COMMAND_EXIT_CODE="0"]
```

To familiarize with GNU gdb debugger, I created a. gdbinit file with the following content.

Set auto-load safe-path /
file simple
break main
break sqrt
info registers

GDB provides the ‘set auto-load safe-path’ setting to list directories trusted for loading files not explicitly requested by user. Each directory can also be a shell wildcard pattern. Without this command warnings would be generated by the debugger. Debugged the program and used run, step, next and continue. The debugging process is logged in a script (gdb.script). Here is a

snapshot of the script: -

```
For more information about this security protection see the
"Auto-loading safe path" section in the GDB manual.  E.g., run from the shell:
[m[?2004h--Type <RET> for more, q to quit, c to continue without paging--c
[?2004l
      info "(gdb)Auto-loading safe path"
[?2004h(gdb) file pro ject1
[?2004l
Reading symbols from [32mproject1[m...
[?2004h(gdb) start
[?2004l
Temporary breakpoint 1 at [34m0x40115e[m: file [32mproject1.c[m, line 7.
Starting program: /home/centos/project1
[Thread debugging using libthread_db enabled]
Using host libthread_db library "[32m/lib64/libthread_db.so.1[m".

Temporary breakpoint 1, [33mmain[m () at [32mproject1.c[m:7
7           [32mint[m max[31m=[m[35m10[m[31m;[m
[?2004hMissing separate debuginfos, use: dnf debuginfo-install glibc-2.34-54.el9.x86_64
[?2004l
[?2004h(gdb) next
[?2004l
8           [32mdouble[m sum[31m=[m[35m0[m[31m;[m
[?2004h[?2004l
[?2004h(gdb) step
[?2004l
11          [01mprintf[m[31m([m[31m"[m[35m\n[m[31m Enter the 10 numbers:[m[35m\n[m[31m"[m[31m);[m
[?2004h[?2004l
[?2004h(gdb) continue
[?2004l
Continuing.

 Enter the 10 numbers:
1
2
3
4
5
6
7
```