CSE303 FUNDAMENTALS OF OPERATING SYSTEMS

ASSIGNMENT 1

MINIX 2.0.4
INSTALLATION & LOGIN SCREEN

AKDENIZ UNIVERSITY

DEPARTMENT OF COMPUTER ENGINEERING

FALL 2025-2026 SEMESTER

YAHYA EFE KURUÇAY

20220808005

efekurucay@hotmail.com

efekurucay.com

26 October, 2025

TABLE OF CONTENTS

1. INTRODUCTION

2. SOURCE CODE ANALYSIS AND TARGET FILE IDENTIFICATION

3. INITIAL ATTEMPT AND ENCOUNTERED PROBLEM

4. MINIX REINSTALLATION PROCESS

- 4.1. BOOT IMAGE CREATION
- 4.2. VIRTUALBOX DISK CONFIGURATION
- 4.3. DISK PREPARATION AND INSTALLATION
- 4.4. COMPLETING MINIX INSTALLATION

5. EDITING AND COMPILING GETTY.C FILE

- 5.1. FILE TRANSFER VIA MOUNT/UMOUNT OPERATIONS
- 5.2. EDITING IN UBUNTU SYSTEM
- 5.3. COMPILATION AND INSTALLATION

6. ADDITIONAL CUSTOMIZATION: HOSTNAME MODIFICATION

7. TESTING AND VERIFICATION

8. CONCLUSION

9. REFERENCES

1. INTRODUCTION

The objective of this study was to modify the source code of the Minix 2.0.4 operating system to change the "login:" text displayed on the login screen to "Giris:". I worked with Minix, a microkernel-based operating system developed for educational purposes, which provided me an ideal platform for gaining system-level programming experience.

2. SOURCE CODE ANALYSIS AND TARGET FILE IDENTIFICATION

To locate the source file controlling the login screen, I examined the Minix source code repository on GitHub. Through my search operations, I determined that the login prompt is defined in the /usr/src/commands/simple/getty.c file. I learned that the Getty program is the system component that manages user login operations and displays the login prompt on the screen.

Figure 2.1: Minix Repo

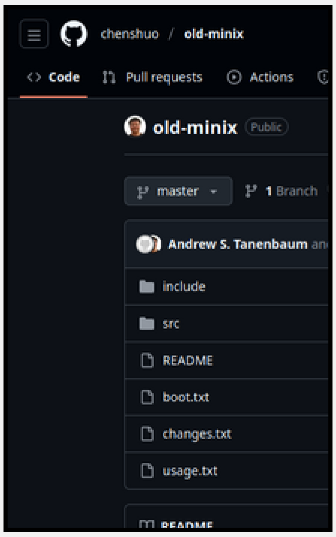
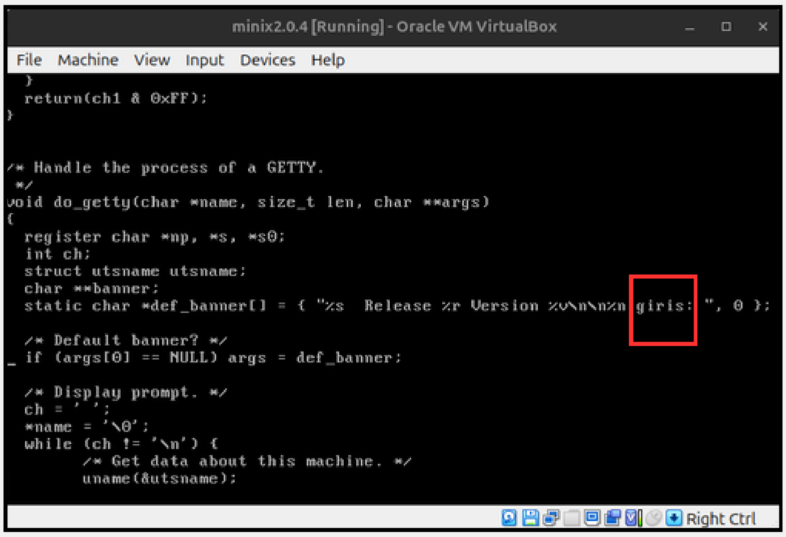


Figure 2.2 : Initial getty.c Source Code View



```
  }
  return(ch1 & 0xFF);
}


/* Handle the process of a GETTY.
 */
void do_getty(char *name, size_t len, char **args)
{
  register char *np, *s, *s0;
  int ch;
  struct utsname utsname;
  char **banner;
  static char *def_banner[] = { "%s  Release %r Version %v\n\n%n giris: ", 0 };

  /* Default banner? */
_ if (args[0] == NULL) args = def_banner;

  /* Display prompt. */
  ch = ' ';
  *name = '\0';
  while (ch != '\n') {
        /* Get data about this machine. */
        uname(&utsname);
```

3. INITIAL ATTEMPT AND ENCOUNTERED PROBLEM

In my first approach, I opened the getty.c file using the vi getty.c command with the Vim editor, and changed the word "login" to "Giris". Following the modification, I planned compilation using /usr/src/Makefile. However, I discovered that my 20MB MinixDist.vdi disk was full due to my previous network experiments. In this situation, I decided to reinstall Minix from scratch to gain disk management experience and to reconfigure the system cleanly.

4. MINIX REINSTALLATION PROCESS

4.1. BOOT IMAGE CREATION

For Minix installation, I first created a boot image. Using the downloaded Intel-2.0.4 folder, I performed the following operations :

```
dd if=/dev/zero of=boot.img bs=1k count=1440
cat Intel-2.0.4/i386/ROOT.MNX Intel-2.0.4/i386/USR.MNX | dd of=boot.img
conv=notrunc
```

4.2. VIRTUALBOX DISK CONFIGURATION

I created two virtual disks in the VirtualBox environment :

- 200MB main Minix installation disk (/dev/c0d0)
- 40MB file transfer disk (minix2.0.4_1.vdi - /dev/c0d1) - I increased this from 20MB in my first installation to 40MB to provide more space for future experiments

I added the created boot.img file to the virtual machine as a floppy disk.

4.3. DISK PREPARATION AND INSTALLATION

I booted the system from floppy and logged in as root. I performed disk partitioning and formatting operations as follows :

```
mkfs /dev/c0d1p0
mount /dev/c0d1p0 /mnt
df
shutdown
```

I copied Minix files to the disk image in the Ubuntu system:

```
sudo mount -o loop,offset=2129408 MinixDist.vdi /mnt
sudo cp Intel-2.0.4/i386/* Intel-2.0.4/src/* /mnt
sudo umount /mnt
```

4.4. COMPLETING MINIX INSTALLATION

I rebooted the virtual machine and completed Minix installation following the instructions in the usage.txt file. During installation, I installed the following packages:

```
mkdir /dist
mount /dev/c0d1p0 /dist
setup /usr </dist/USR.TAZ    NET.TAZ CMD.TAZ FIX.TAZ  AI.TAZ etc
```

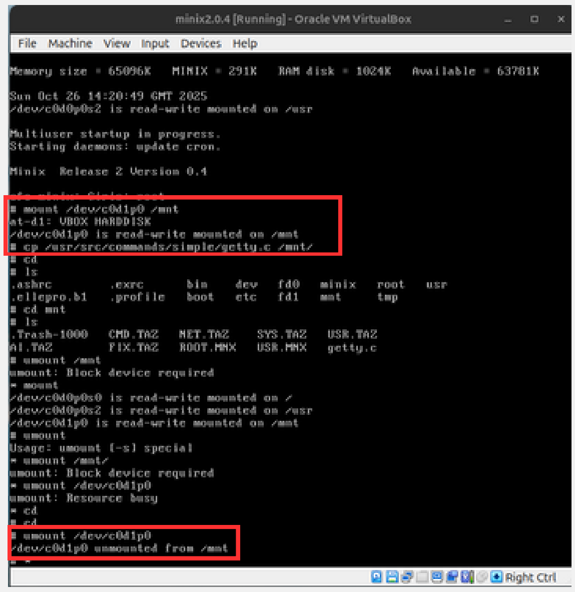
I installed all system packages. Additionally, I added custom folders from my previous experiments (Eliza AI simulation and ask_gemini folder containing Gemini proxy experiment) to the system. After completing the installation, I removed the floppy disk image from the virtual machine.

5. EDITING & COMPILING GETTY.C

5.1. FILE TRANSFER VIA MOUNT/UMOUNT OPERATIONS

Unlike my first approach, I performed file transfer using mount/umount operations instead of vim. This method provided me with a better learning experience in file management :
In Minix system, I executed the commands in Figure 5.1.1

Figure 5.1.1: Mount and umount op.



minix2.0.4 [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Right Ctrl

```
Memory size = 65096K   MINIX = 291K   RAM disk = 1024K   Available = 63781K

Sun Oct 26 14:20:49 GMT 2025
/dev/c0d0p0s2 is read-write mounted on /usr

Multiuser startup in progress.
Starting daemons: update cron.

Minix  Release 2 Version 0.4

[illegible]
# mount /dev/c0d1p0 /mnt
at-d1: VBOX HARDDISK
/dev/c0d1p0 is read-write mounted on /mnt
# cp /usr/src/commands/simple/getty.c /mnt/
# cd
# ls
.ashrc        .exrc      bin    dev   fd0   minix   root   usr
.ellepro.b1   .profile   boot   etc   fd1   mnt     tmp
# cd mnt
# ls
.Trash-1000   CMD.TAZ   NET.TAZ    SYS.TAZ   USR.TAZ
AI.TAZ        FIX.TAZ   ROOT.MNX   USR.MNX   getty.c
# umount /mnt
umount: Block device required
# mount
/dev/c0d0p0s0 is read-write mounted on /
/dev/c0d0p0s2 is read-write mounted on /usr
/dev/c0d1p0 is read-write mounted on /mnt
# umount
Usage: umount [-s] special
# umount /mnt/
umount: Block device required
# umount /dev/c0d1p0
umount: Resource busy
# cd
# cd
# umount /dev/c0d1p0
/dev/c0d1p0 unmounted from /mnt
# _
```

mkdir /mnt
mount /dev/c0d1p0 /mnt
cp /usr/src/commands/simple/getty.c /mnt/
umount /dev/c0d1p0

5.2. EDITING IN UBUNTU SYSTEM

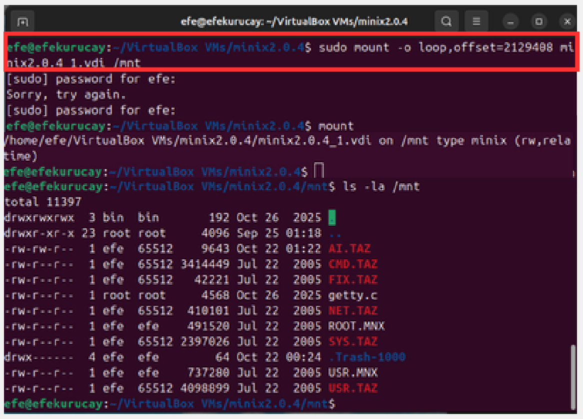
After shutting down Minix, I switched to Ubuntu. I mounted the VDI disk in Ubuntu and retrieved the getty.c file :

sudo mount -o loop,offset=2129408 minix2.0.4_1.vdi /mnt

Using a text editor, I changed the word "login" in the getty.c file to "Giris".
After completing my edit: sudo umount /mnt

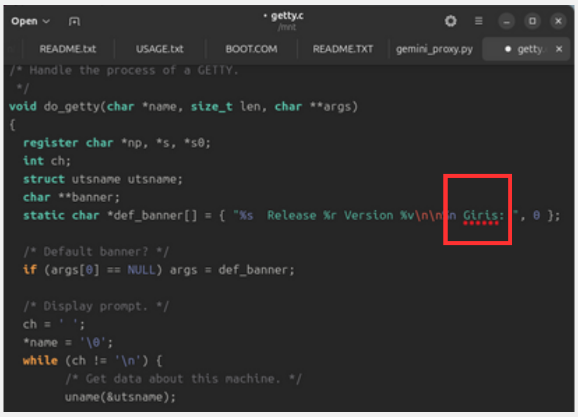
I checked the mount command output to verify that the disk was successfully unmounted.

Figure 5.2.1: Mount and list op.



```
efe@efekurucay:~/VirtualBox VMs/minix2.0.4$ sudo mount -o loop,offset=2129408 mi
nix2.0.4_1.vdi /mnt
[sudo] password for efe:
Sorry, try again.
[sudo] password for efe:
efe@efekurucay:~/VirtualBox VMs/minix2.0.4$ mount
/home/efe/VirtualBox VMs/minix2.0.4/minix2.0.4_1.vdi on /mnt type minix (rw,rela
time)
efe@efekurucay:~/VirtualBox VMs/minix2.0.4$
efe@efekurucay:~/VirtualBox VMs/minix2.0.4/mnt$ ls -la /mnt
total 11397
drwxrwxrwx  3 bin  bin       192 Oct 26  2025 .
drwxr-xr-x 23 root root     4096 Sep 25 01:18 ..
-rw-rw-r--  1 efe  65512    9643 Oct 22 01:22 AI.TAZ
-rw-r--r--  1 efe  65512 3414449 Jul 22  2005 CMD.TAZ
-rw-r--r--  1 efe  65512   42221 Jul 22  2005 FIX.TAZ
-rw-r--r--  1 root root     4568 Oct 26  2025 getty.c
-rw-r--r--  1 efe  65512  410101 Jul 22  2005 NET.TAZ
-rw-r--r--  1 efe  efe    491520 Jul 22  2005 ROOT.MNX
-rw-r--r--  1 efe  65512 2397026 Jul 22  2005 SYS.TAZ
drwx------  4 efe  efe        64 Oct 22 00:24 .Trash-1000
-rw-r--r--  1 efe  efe    737280 Jul 22  2005 USR.MNX
-rw-r--r--  1 efe  65512 4098899 Jul 22  2005 USR.TAZ
efe@efekurucay:~/VirtualBox VMs/minix2.0.4/mnt$
```

Figure 5.2.2: getty.c on Ubuntu Text editor



```c
/* Handle the process of a GETTY.
 */
void do_getty(char *name, size_t len, char **args)
{
  register char *np, *s, *s0;
  int ch;
  struct utsname utsname;
  char **banner;
  static char *def_banner[] = { "%s  Release %r Version %v\n\n\n Giris: ", 0 };

  /* Default banner? */
  if (args[0] == NULL) args = def_banner;

  /* Display prompt. */
  ch = ' ';
  *name = '\0';
  while (ch != '\n') {
        /* Get data about this machine. */
        uname(&utsname);
```

5. EDITING & COMPILING GETTY.C

5.3. COMPILATION AND INSTALLATION

I restarted Minix and transferred the updated file back to the system:

```
mount /dev/c0d1p0 /mnt
cp /mnt/getty.c /usr/src/commands/simple/
```

I performed compilation: `make install`

My compilation completed without errors or warnings. After the operation, I unmounted the disk: `umount /dev/c0d1p0`

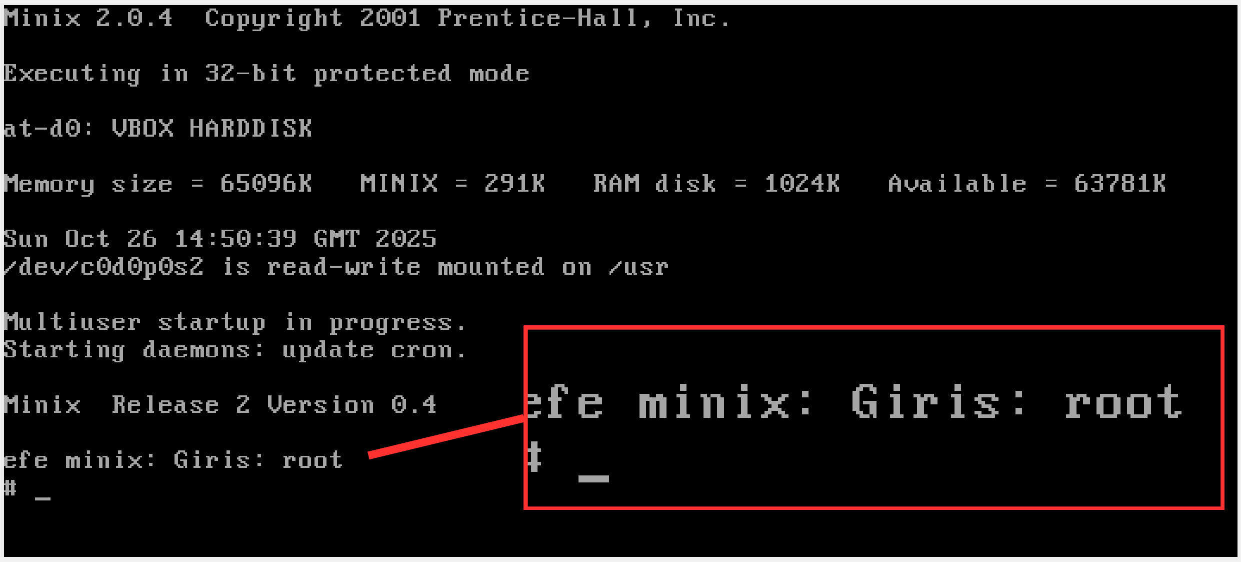
6. ADDITIONAL CUSTOMIZATION: HOSTNAME MODIFICATION

For system customization purposes, I wanted to change the hostname displayed as "noname". By examining the source code in the GitHub repository, I determined that hostname information is stored in the /etc/hostname.file file. Using the Vim editor, I changed the "noname" value in this file to "efe minix:". When I rebooted the system, the change was successfully applied.

7. TESTING AND VERIFICATION

After rebooting the system, I verified that "Giris:" was displayed on the login screen instead of "login:". My hostname change also worked as expected, displaying the "efe minix:" prompt on the screen.

Figure 7.1: Minix login screen after the updating



```
Minix 2.0.4  Copyright 2001 Prentice-Hall, Inc.

Executing in 32-bit protected mode

at-d0: VBOX HARDDISK

Memory size = 65096K   MINIX = 291K   RAM disk = 1024K   Available = 63781K

Sun Oct 26 14:50:39 GMT 2025
/dev/c0d0p0s2 is read-write mounted on /usr

Multiuser startup in progress.
Starting daemons: update cron.

Minix  Release 2 Version 0.4

efe minix: Giris: root
# _
```

8. CONCLUSION

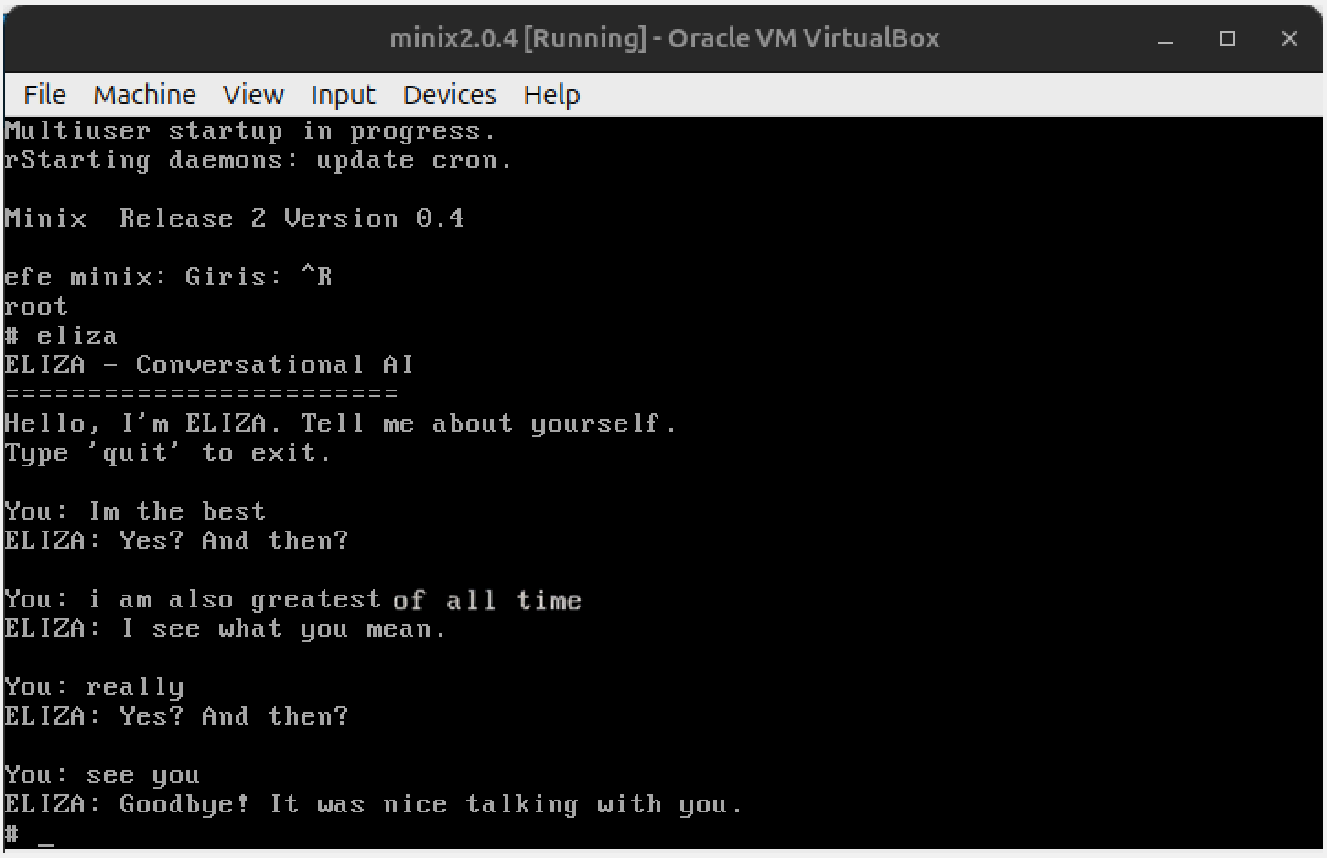
In this study, I successfully customized the login prompt by modifying the source code of the Minix 2.0.4 operating system. Throughout the process, I gained experience in the following areas:

- Source code analysis and target file identification
- VirtualBox virtual machine configuration
- Disk partitioning and file system creation
- File system management using mount/umount operations
- Minix source code compilation process
- Editing system configuration files

In particular, I gained in-depth knowledge on how the mount/umount mechanism is used for file transfer between different operating systems. The disk space shortage problem provided me with an opportunity to reinstall the system and plan better. All my modifications were successfully compiled and tested.

9. REFERENCES

- Spivey, M. (2011). Installing Minix 2 on VirtualBox. Oxford University Computing Services.
- Chen, S. Old Minix Source Code Repository. GitHub.
    - https://github.com/chenshuo/old-minix/
- Minix 2.0.4 Intel Distribution.
    - http://www.minix3.org/previous-versions/


minix2.0.4 [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help

```
Multiuser startup in progress.
rStarting daemons: update cron.

Minix  Release 2 Version 0.4

efe minix: Giris: ^R
root
# eliza
ELIZA - Conversational AI
========================
Hello, I'm ELIZA. Tell me about yourself.
Type 'quit' to exit.

You: Im the best
ELIZA: Yes? And then?

You: i am also greatest of all time
ELIZA: I see what you mean.

You: really
ELIZA: Yes? And then?

You: see you
ELIZA: Goodbye! It was nice talking with you.
# _
```