بسم الله الرحمن الرحیم

پروژه اول درس مهارت های پیشرفته کار با کامپیوتر

دکتر دوستی

مهدی وجهی - ۸۱۰۱۰۱۵۵۸

فهرست

سوال ۱

لازم است فایل اجرایی رو داشته باشیم و در سرور قرار بدیم. برای این کار من این مخزن را پیدا کردم. و به آسانی اجرا میشه با ۳ خط دستور زیر:

```
 # On the server
wget https://github.com/dtinth/mosh-static/releases/latest/download/mosh-server
chmod +x mosh-server

# On the client
mosh --server=./mosh-server <username>@<hostname>
```

سوال ۲

نحوه استفاده از LLM

تمامی کد های این بخش به غیر vagrantfile, test.sh با جمنای تولید شده. همچنین گزارش بخش کد کلاینت ssh نیز با جمنای تولید شده.

کد کلاینت ssh

اجزای کد

کد پایتون به چند بخش کلیدی تقسیم شده است:

- توابع مدیریت دستورات محلی: توابعی مانند upload_file ,()get_system_health(), و download_file() منطق مربوط به دستورات خاص را کپسوله می‌کنند. این توابع روی کلاینت اجرا می‌شوند و با سرور تعامل مستقیم ندارند (به جز از طریق client برای SFTP).
- دیکشنری ROLES: این دیکشنری ساختار داده اصلی برای پیاده‌سازی RBAC است.
  - کلیدها: نام نقش‌ها ("admin", "user")
  - مقادیر: دیکشنری‌هایی که مجوزهای نقش را تعریف می‌کنند ("can_execute")
- دیکشنری LOCAL_COMMANDS: این دیکشنری، فراداده مربوط به دستورات محلی را نگهداری می‌کند.
  - کلیدها: نام دستورات محلی ("upload_file", "system_health", و غیره.)
  - مقادیر: دیکشنری‌هایی که ویژگی‌های دستور را مشخص می‌کنند:
    - "func": تابع پایتون مربوط به دستور
    - "remote": یک مقدار بولی که نشان می‌دهد آیا دستور روی سرور اجرا می‌شود یا نه (False برای محلی)

- "sudo": یک مقدار بولی که نشان می‌دهد آیا برای اجرای دستور به sudo نیاز است یا نه
- "args": لیستی از نام آرگومان‌های مورد انتظار تابع

- توابع کمکی:
  - ssh_connect(): اتصال SSH را مدیریت می‌کند. این تابع تلاش می‌کند کلیدهای SSH را در مسیرهای پیش‌فرض پیدا کند.
  - check_permission(): مجوز کاربر را بر اساس نقش و دستور درخواستی بررسی می‌کند.
  - execute_command(): هسته اصلی سیستم است. این تابع تصمیم می‌گیرد که یک دستور چگونه اجرا شود (محلی یا راه دور) و آرگومان‌های مناسب را به تابع مربوطه ارسال می‌کند.
- بلوک "__main__" == __name__ if__:: این بلوک، نقطه ورود اسکریپت است. این بلوک کارهای زیر را انجام می‌دهد:
  - دریافت نقش کاربر از ورودی
  - برقراری اتصال SSH با استفاده از ssh_connect()
  - دریافت دستور و آرگومان‌ها از ورودی کاربر
  - فراخوانی execute_command() برای اجرای دستور
  - نمایش خروجی دستور
  - بستن اتصال SSH

جریان اجرای دستور

1. کاربر نقش و دستور را وارد می‌کند.
2. بلوک "__main__" == __name__ if__": نقش و دستور را دریافت می‌کند.
3. execute_command() فراخوانی می‌شود.
4. check_permission() بررسی می‌کند که آیا کاربر اجازه اجرای دستور را دارد یا نه.
5. اگر دستور در LOCAL_COMMANDS وجود داشته باشد، به عنوان یک دستور محلی در نظر گرفته می‌شود:
   - آرگومان‌ها از args (آرگومان‌های خط فرمان) استخراج شده و به تابع مربوطه ارسال می‌شوند.
   - تابع محلی اجرا می‌شود.
6. اگر دستور در LOCAL_COMMANDS وجود نداشته باشد، به عنوان یک دستور راه دور در نظر گرفته می‌شود:
   - دستور از طریق SSH روی سرور اجرا می‌شود.
   - خروجی (stdout, stderr, exit_code) از سرور دریافت می‌شود.
7. خروجی دستور به کاربر نمایش داده می‌شود.

گرفتن backup

برای بکاپ گرفتن ۲ شل اسکریپت تعریف شده. یکی برای خود فراید بکاپ است و روند فشرده سازی و ذخیره را انجام می دهد (با استفاده از tar) و دیگری برای تنظیم زمان اجرا و مشخص کردن محل ذخیره است. برای مشخص کردن محل ذخیره متغیر مربوطه را تنظیم می کند و برای زمان اجرا یک cron job تعریف می کند.

نتیجه

چندین تست طراحی شده که بخش های مهم این سوال را پوشش داده. نتایج در ادامه آورده شده.

```
@@@@@@@@@@Test 1@@@@@@@@@@@@
Test 1: user sys health
user: user, command: system_health
Enter your role (admin/user): Successfully connected to 192.168.56.10 as
normal_user
Enter the command and arguments:
--- Executing Command: system_health as user ---
CPU Usage: 2.3%
Memory Usage: 59.0%
Disk Usage: 88.6%
@@@@@@@@@@Test 1.5@@@@@@@@@@@@
Test 1.5: user invalid command
user: user, command: ls -ltrh
Enter your role (admin/user): Successfully connected to 192.168.56.10 as
normal_user
Enter the command and arguments:
--- Executing Command: ls as user ---
Permission denied: Role 'user' cannot execute command 'ls'
@@@@@@@@@@Test 2@@@@@@@@@@@@
Test 2: user upload file
user: user, command: upload_file ./send_file
/home/normal_user/Downloads/send_file
Enter your role (admin/user): Successfully connected to 192.168.56.10 as
normal_user
Enter the command and arguments:
--- Executing Command: upload_file as user ---
File uploaded successfully to /home/normal_user/Downloads/send_file
@@@@@@@@@@Test 3@@@@@@@@@@@@
Test 3: user upload file invalid path
user: user, command: upload_file ./send_file /home/normal_user/send_file
Enter your role (admin/user): Successfully connected to 192.168.56.10 as
normal_user
Enter the command and arguments:
--- Executing Command: upload_file as user ---
Error: Uploads for 'user' role are restricted to the
/home/normal_user/Download directory.
```

```
@@@@@@@@@@Test 4@@@@@@@@@@@@
Test 4: admin upload file
user: admin, command: upload_file ./backup_script.sh
/home/admin_user/backup_script.sh
Enter your role (admin/user): Successfully connected to 192.168.56.10 as
admin_user
Enter the command and arguments:
--- Executing Command: upload_file as admin ---
File uploaded successfully to /home/admin_user/backup_script.sh
@@@@@@@@@@Test 4-2@@@@@@@@@@@@
Test 4-2: admin upload file
user: admin, command: upload_file ./schedule_backup.sh
/home/admin_user/schedule_backup.sh
Enter your role (admin/user): Successfully connected to 192.168.56.10 as
admin_user
Enter the command and arguments:
--- Executing Command: upload_file as admin ---
File uploaded successfully to /home/admin_user/schedule_backup.sh
@@@@@@@@@@Test 5@@@@@@@@@@@@
Test 5: user download file
user: user, command: download_file ./receive_file
/home/normal_user/Downloads/send_file
Enter your role (admin/user): Successfully connected to 192.168.56.10 as
normal_user
Enter the command and arguments:
--- Executing Command: download_file as user ---
File downloaded successfully to ./receive_file
@@@@@@@@@@Test 6@@@@@@@@@@@@
Test 6: admin chmod file
user: admin, command: chmod +x ~/*.sh
Enter your role (admin/user): Successfully connected to 192.168.56.10 as
admin_user
Enter the command and arguments:
--- Executing Command: chmod as admin ---
--- Exit Code: 0 ---
@@@@@@@@@@Test 7@@@@@@@@@@@@
Test 7: admin set backup schedule
user: admin, command: sudo ./schedule_backup.sh "* * * * *" "/var/backup"
Enter your role (admin/user): Successfully connected to 192.168.56.10 as
admin_user
Enter the command and arguments:
--- Executing Command: sudo as admin ---
--- Standard Output ---
Backup of /etc and /home scheduled to run with interval '* * * * *' and
output directory '/var/backup'.
Any previous backup schedule has been replaced.
You can check your cron jobs with 'crontab -l'.
```

```
--- Standard Error ---
no crontab for root

--- Exit Code: 0 ---
@@@@@@@@@@Test 7-2@@@@@@@@@@@@
Test 7-2: admin check backup schedule
user: admin, command: sudo crontab -l
Enter your role (admin/user): Successfully connected to 192.168.56.10 as
admin_user
Enter the command and arguments:
--- Executing Command: sudo as admin ---
--- Standard Output ---
* * * * * OUTPUT_DIR="/var/backup" /home/admin_user/backup_script.sh

--- Exit Code: 0 ---
@@@@@@@@@@Test 7-3@@@@@@@@@@@@
Test 7-3: admin check backup result
user: admin, command: sleep 30 && ls -ltrh /var/backup
Enter your role (admin/user): Successfully connected to 192.168.56.10 as
admin_user
Enter the command and arguments:
--- Executing Command: sleep as admin ---
--- Standard Output ---
total 572K
-rw-r--r-- 1 root root 572K Apr 10 21:50 backup_20250410_215001.tar.gz

--- Exit Code: 0 ---
```

سوال ۳

نحوه استفاده از LLM

کد پایتون برای رسم نمودار استفاده است با جمنای تولید شده. همچنین کد های c که برای تست عملکرد نوشته شده نیز جمنای تولید کرده.

مانیتور و مدیریت پردازه ها

برای سنجش سیستم و غیرفعال کردن پردازه های پرمصرف کد resource_manager.py را نوشتیم. ساختار کد خیلی پیچیده نیست. طبق فاصله زمانی ای که مشخص شده سیستم را پایش می کند اگر منابع پیش از حد گفته شده بود یک پیام در wall می نویسد و پردازه پر مصرف را اگر CPU باشد متوقف و اگر mem باشد می کشد زیرا توقف آن تاثیری رو سیستم ندارد. همچنین به پردازه ها با شماره ی پایین و همچنین خودش کاری ندارد. تنظیمات آن در آدرس etc/resource_manager/config.json/ قرار دارد. برای تبدیل کردن آن به یک سرویس فایل زیر را می نویسیم:

```
[Unit]
Description=Resource Manager Service
After=network.target

[Service]
Type=simple
User=root
ExecStart=/usr/local/sbin/resource_manager.py
Restart=always

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

در این فایل مشخص شده که این سرویس بعد از شبکه بالا می آید و قبل از multiuser. همچنین در هنگام پایان و خطا مجدد ریست می شود.

بعد اضافه کردن سرویس تنها کافیست که آن را فعال کنیم.

همچنین لاگ های سیستم به صورت پیش فرض در var/log/resource_manager/ ذخیره می شود. یک فایل برای وضعیت رم و cpu که ردیف های زیر را دارد:

```
[timestamp, cpu_usage, memory_usage]
```

فایل دیگر پردازه هایی است که پرمصرف بوده اند و دارای ردیف های زیر است:

```
[timestamp, process.info['pid'], process.info['name'], status, kill]
```

نمایش پردازه های متوقف

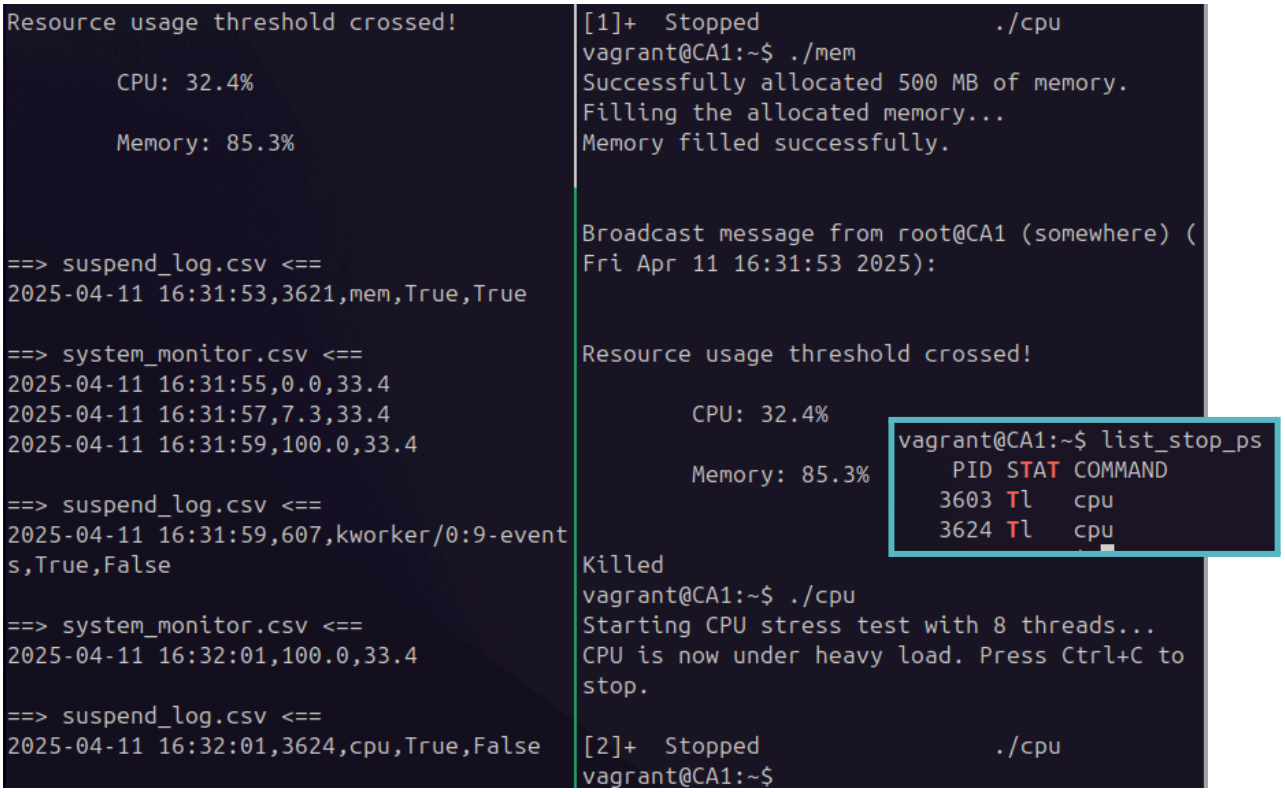
برای نمایش پردازه های متوقف یک alias می نویسم که بسیار ساده است.

```
alias list_stop_ps="ps axo pid,stat,comm | grep T"
```

بررسی عملکرد

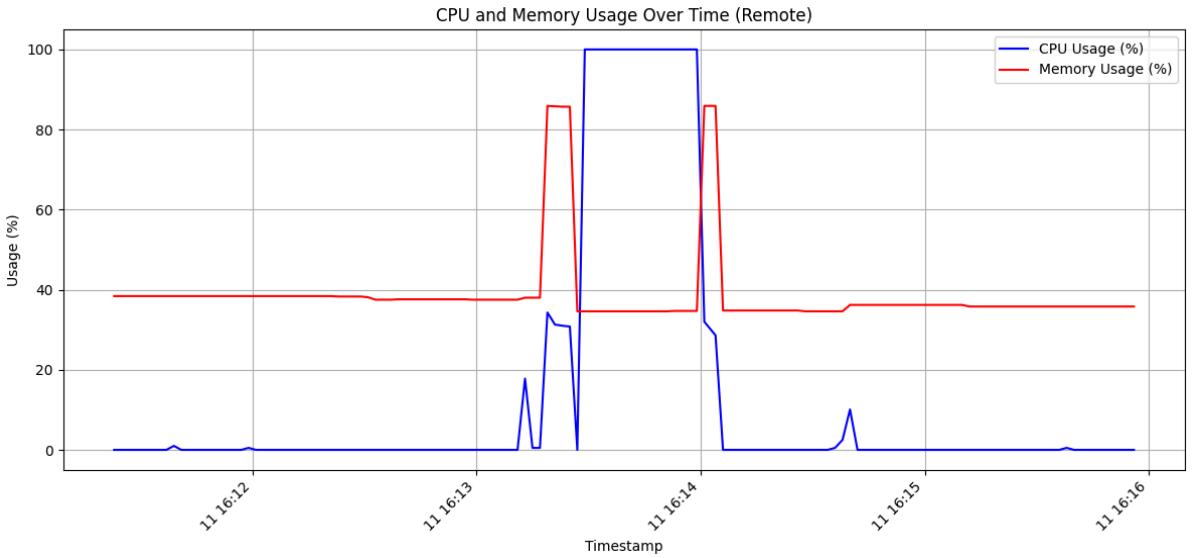
برای بررسی عملکرد دو فایل نوشته شده که یکی از حافظه استفاده می کند و یکی از cpu. نتیجه را می توانید در تصویر زیر مشاهده کنید که موارد زیر انجام شده:

- ثبت لاگ استفاده از سیستم
- ثبت لاگ توقف و کشته شدن پردازه ها
- اطلاع رسانی رد شدن از حد مجاز
- توقف و کشته شدن پردازه ها
- نمایش لیست پردازه های متوقف



رسم نمودار استفاده

فایل پایتون دیگر نیز نوشته شده برای نمایش نمودار استفاده. که در آن به سرور sftp میزند و فایل مربوطه را دریافت می کند و آن را رسم می کند. نمونه ای از خروجی در تصویر زیر مشاهده می کنید.



سوال ۴

تشریحی

مکانیزم بوت لودر GRUB

به صورت کلی وظیفه grub اجرا کردن os است و این مرحله بعد از UEFI قرار می گیرد. عملا ما می توانیم به صورت مستقیم خود کرنل را بارگیری کنیم اما استفاده از grub مزایایی دارد که از آن استفاده می کنیم. به صورت کلی grub بعد از بارگیری شدن منویی از کرنل ها و سیستم عامل ها می آورد و می توانیم بین آنها انتخاب کنیم.

در اینجا مواردی مثل این از کدام هارد و با چه قالبی MBR/GPT و مواردی مثل این که کرنل کجاست و ریشه را کجا در نظر بگیرد. UUID دیسک چه عددی هست و محل ramdisk برای بارگذاری اولیه مشخص شده. Grub موارد فوق را طبق انتخاب کاربر بارگذاری می کند و اجرا سیستم را تحویل می دهد. به صورت جزیی تر گام های زیر طی می شود:

1. ثابت افزار grub را بارگذاری می کند.
2. موارد اولیه grub بارگذاری می شود.

3. موارد اولیه مقدار دهی می شود. در این نقطه فایل سیستم ها و دیسک در دسترس قرار می گیرند.
4. بوت پارتیشن پیدا می شود و تنظیماتش بارگذاری می شود.
5. کاربر می تواند تنظیمات را عوض کند.
6. تنظیمات اجرا می شود.
7. باقی مانده ماژول ها بارگذاری می شود.
8. دستورات مربوط و بوت اجرا می شود و سیستم به os تحویل داده می شود.

BIOS/UEFI

این دو، دو دسته از ثابت افزار هستند و در rom یا e2prom و موارد مشابه ذخیره می شوند. بعد از روشن شدن سیستم این برنامه ها بارگیری می شوند و سخت افزار سیستم را بررسی می کند (اصطلاحا POST می گویند) و صحت و سلامت آنها را بررسی می کنند و در صورت مشکل با مواردی مانند بوق زدن این موضوع را اطلاع می دهد. در POST سلامت CPU و رجیستر های آن، سلامت خود را با checksum بررسی می کند، DMA ،RAM، و موارد مشابه. سپس یکسری تنظیمات سیستم را همراه با وضعیت سیستم در اختیار قرار می دهد.BIOS از MBR استفاده می کند و UEFI از GPT. همچنین UEFI از بوت امن، فرمت FAT32 و رابط گرافیکی هم پشتیبانی می کند. در نهایت آنها سیستم را به سیستم عامل یا بوت لودر تحویل می دهند.

بارگذاری و اجرای کرنل

مراحل بارگذاری به صورت مختصر به شرح زیر است:

1. روشن شدن سیستم
2. بارگذاری BIOS/UEFI
3. بررسی سلامت سیستم
4. بوت شدن از محل مشخص شده
5. شناسایی شدن پارتیشن EFI
6. لود شدن بوت لودر
7. مشخص کردن تنظیمات و این که کدام کرنل اجرا شود
8. بارگذاری کرنل و مقدار دهی اولیه

مراحل اجرا کرنل برای سیستم عامل های مختلف متفاوت است اما به صورت کلی گام های زیر را دارد:

- شناسایی سخت افزار
- بارگذاری درایور ها و شناسایی دیسک
- اجرای اولین پردازه
- اجرای رابط کاربری و رفتن به user mode

MBR

Partition table جدولی است که شامل ۴ خانه که آدرس شروع ۴ پارتیشن اصلی را مشخص می کند. هرکدام از خانه ها ۱۶ بایت هستند.
Boot signature دو بایت آخر MBR است که مقادیر 0x55,0xAA دارند که جهت بررسی سلامت MBR توسط BIOS استفاده می شود.
Winhex یک برنامه ویندوزی که disk editor و hex editor هست.

عملی

تغییر زمان انتظار

در فایل تنظیمات `GRUB_TIMEOUT` را برابر مقدار مورد نظر قرار می دهیم. لازم به ذکر است که بعد از تمامی تغییرات دستور `update-grub` را اجرا می کنیم.

```
- name: Set grub timeout to {{ timeout }} seconds
  ansible.builtin.lineinfile:
    path: /etc/default/grub
    regexp: '^GRUB_TIMEOUT='
    line: 'GRUB_TIMEOUT={{ timeout }}'
  notify: Update grub
```

همچنین جهت نمایش منو آن را از مخفی خارج می کنیم.

نتیجه:

اضافه کردن مدخل سیستم عامل جدید

ابتدا در Vagrantfile دیسک یک اوبونتو دسکتاپ را به vm خود متصل می کنیم.

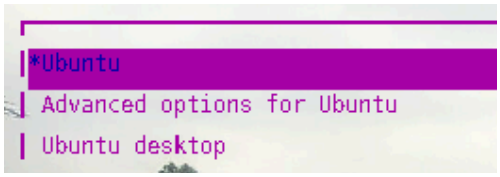
```
vb.customize ["storageattach", :id, "--storagectl", "SATA Controller",
"--port", 1, "--device", 0, "--type", "hdd", "--medium",
File.expand_path("~/VirtualBox VMs/ubuntu.vdi")]
```

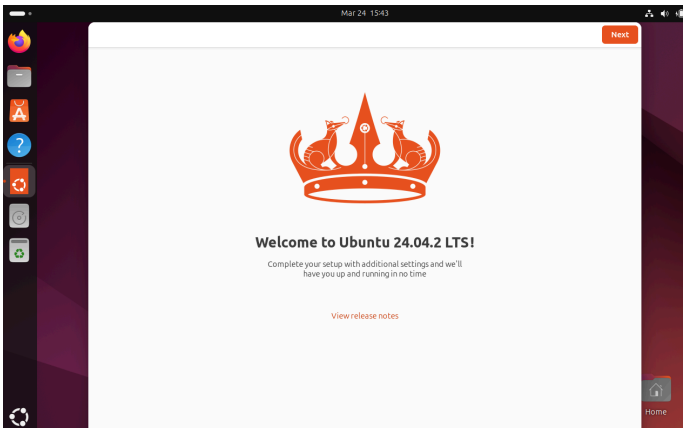
در فایل مربوطه که در کد مشخص است. محل ذخیره شدن،UUID پارتیشن، آدرس کرنل و رم دیسک را مشخص می کنیم.

```
- name: Add ubuntu desktop entry
  ansible.builtin.blockinfile:
    path: /etc/grub.d/40_custom
    block: |
```

```
        menuentry 'Ubuntu desktop' {
            set root='hd1,gpt2'
            linux   /boot/vmlinuz-6.11.0-17-generic root=UUID={{ UUID }}
ro quiet splash $vt_handoff
            initrd  /boot/initrd.img-6.11.0-17-generic
        }
    notify: Update grub
```

نتیجه:





اضافه کردن مدخل برای اجرا سیستم عامل به همراه اسکریپت

ابتدا init خود را می نویسیم. در آخر آن هم init اصلی را اجرا می کنیم.

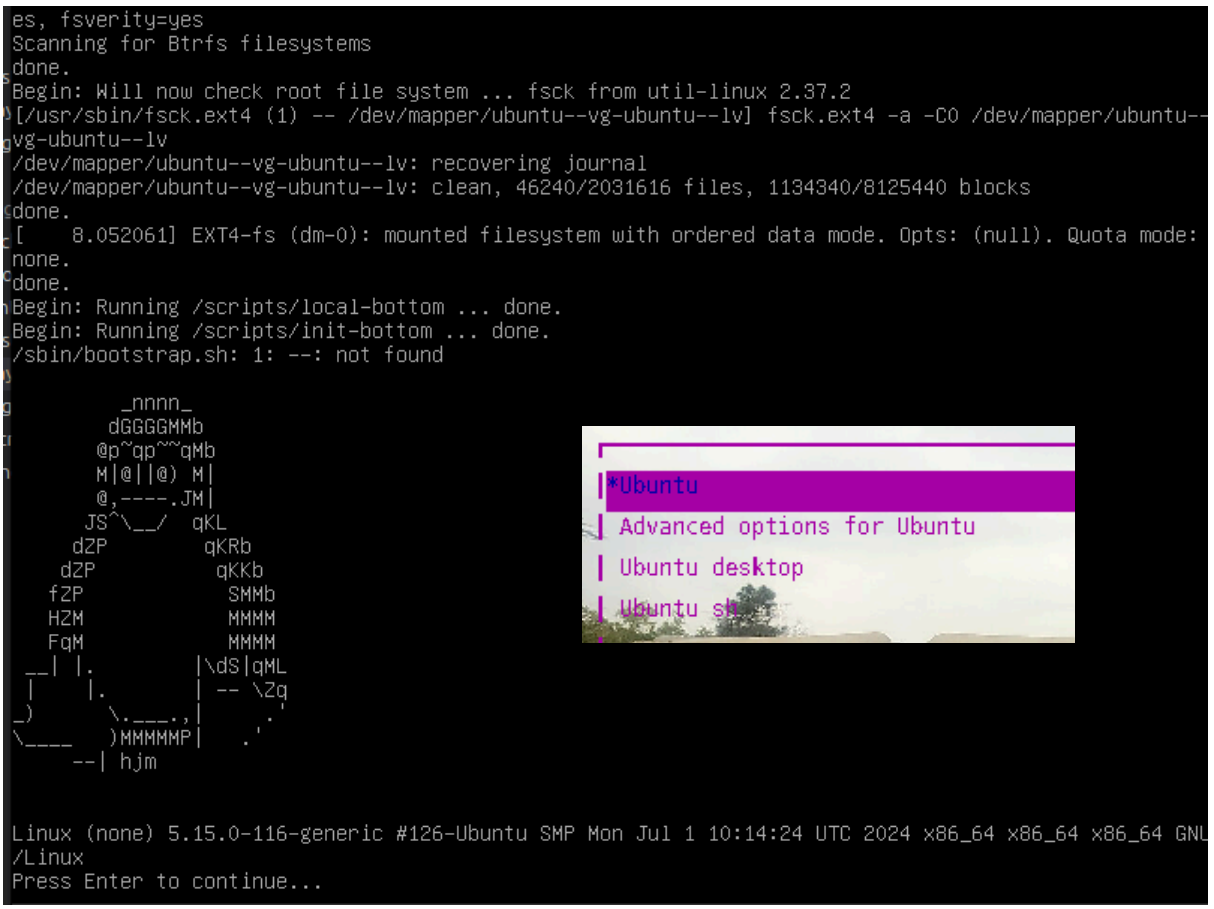
```
#!/bin/sh

echo "LINUX LOGO"

uname -a

read -p "Press Enter to continue..." tmp
exec /sbin/init
```

مانند قسمت قبل است فقط به جای init اصلی init خودمان را به آن می دهیم.

```
 - name: Add ubuntu with bootstrap.sh entry
   ansible.builtin.blockinfile:
     path: /etc/grub.d/40_custom
     block: |
       menuentry 'Ubuntu sh' {
         set root='hd0,gpt2'
         linux   /vmlinuz-5.15.0-116-generic
root=/dev/mapper/ubuntu--vg-ubuntu--lv init={{bootstrap_path}} ro
net.ifnames=0 biosdevname=0
```

```
        initrd  /initrd.img-5.15.0-116-generic
      }
    marker: "# {mark} sh ANSIBLE MANAGED BLOCK"
  notify: Update grub
```

نتیجه:



سفارشی سازی GRUB

فونت و عکس را در /boot/grub می ریزیم و تنظیمات را اعمال می کنیم. همچنین با دستور `grub-mkfont` فرمت فونت را به pf2 که مناسب grub است تبدیل می کنیم.

```
- name: Set background img grub
  ansible.builtin.lineinfile:
    path: /etc/default/grub
    regexp: '^GRUB_BACKGROUND='
    line: 'GRUB_BACKGROUND="{{background_path}}"'
  notify: Update grub

- name: Set grub font
  ansible.builtin.lineinfile:
```

```
    path: /etc/default/grub
    regexp: '^GRUB_FONT='
    line: 'GRUB_FONT="{{ grub_dir }}font.pf2"'
  notify: Update grub

- name: Convert font to pf2
  ansible.builtin.command: "grub-mkfont {{ font_path }} -o {{ grub_dir
}}font.pf2 -s 12"

- name: Set grub color
  ansible.builtin.blockinfile:
    path: /etc/grub.d/40_custom
    block: |
      set color_normal=magenta/black
      set color_highlight=blue/magenta
    marker: "# {mark} color ANSIBLE MANAGED BLOCK"
  notify: Update grub
```

نتیجه:

تحلیل عملکرد

خروجی زمانی در ۲ فایل گذاشته شده. به صورت مختصر بدون تنظیمات ما سیستم در ۴.۵ ثانیه به تارگت GUI می رسد ولی بعد تنظیمات ۱۶ ثانیه. بعد از اعمال تغییرات سیستم ۱۳ ثانیه منتظر شبکه مانده و این باعث افزایش زمان بوت شده ولی باقی موارد تقریبا تغییری نداشته اند. (حتی کمی سریع تر شده) با وجود انجام چندین باره آن نسخه دست نخورده تاخیری برای شبکه ندارد ولی بعد از انجام اولین تغییر این تاخیر ایجاد می شود. دلیل آن هم احتمالا بعضی از تغییراتی است که ansible ایجاد کرده.

دستورات و لینک های استفاده از LLM

https://g.co/gemini/share/be1380f0aadb

سوال ۲ رو کدش رو بزن
می خوام اسکریپت پایتون کامند رو از ورودی بخونه و نتیجه رو توی خروجی نشون بده
می خوام اول برنامه انتخاب کدم ادمین باشم یا کاربر معمولی همچنین کلید ssh رو خودش پیدا می کنه
نمی خواد پاس بدی پسورد هم نگیر

---

https://g.co/gemini/share/754305914539

پیاده‌سازیات وماسیونسرور (پشتیبان گیریو به روزرسانیسیستم) شماباید یک اسکریپت بنویسید که به صورتدوره ایاز دایرکتوري هایمهم مانند etc/ و home/ پشتیبان گیریکند. این اسکریپت بایدفایل هایپشتیبان را فشرده کرده و در یک دایرکتوري مشخص ذخیره کند. اسکریپت شما باید run_interval و output_dir رابه عنوانکانفیگ قبول کند و بتوان این تنظیمات را بدون تغییر کد اعمال کرد و دوباره سیستم را راه اندازیکرد.

یک bash script بنویس براش

توی کد فارسی ننویس

الان کد رو انگلیسی بده

یک اسکریپت بنویس که run_interval و output_dir بگیره و اعمال کنه

یعنی اسکریپ اصلی همون قبلی باشه این این دوتا چیز رو تنظیم کنه

می خوام برای تست هر ۱ دقیقه یکبار بکاپ بگیره باید چیکار کنم؟

---

https://chat.deepseek.com/a/chat/s/2d66768c-0f39-4ef9-93bd-610a2f591531

یه برنامه c بنویس که ۵۰۰ مگ حافظه بگیره
متن های توی کد رو انگلیسی بنویس
یه برنامه بنویس که CPU رو خیلی درگیر کنه

/usr/bin/ld: /tmp/ccccvkRa.o: in function `cpu_intensive_task': cpu.c:(.text+0x32): undefined reference to `sin' /usr/bin/ld: cpu.c:(.text+0x50): undefined reference to `cos' /usr/bin/ld: cpu.c:(.text+0x77): undefined reference to `tan' /usr/bin/ld: cpu.c:(.text+0x99): undefined reference to `sqrt' collect2: error: ld returned 1 exit status

---

https://g.co/gemini/share/60bd601c683e

یه فایل csv دارم به این شکل

14:03:20,25.7,38.3 11-04-2025

14:03:22,0.0,38.4 11-04-2025

که این هست

[timestamp, cpu_usage, memory_usage]

می خوام با matplot بکشمش

جفتش رو توی یک نمودار بنداز

فقط نکته اینه که این فایل روی یک سرور دیگه هست می خوام بره از اون بخونه

برای اتصال به سرور از ssh استفاده می کنم

رمز هم نمی خواد چون کلید ssh رو سرور هست و مشکلی نداره

با توجه به این موارد بازنویسی کن