Can you write a script to perform a forensically sound hard disk image acquisition from a series of external SSD drives that have been in a RAID? However, I don't know what level of RAID has been used. Can the script determine what level of RAID is used and reconstruct a viable single disk image?

I can provide you with a Python script that utilizes the 'mdadm' and 'dd' command-line tools to help you determine the RAID level and create a disk image. However, please note that this script assumes you have basic knowledge of Linux, 'mdadm', and 'dd' tools, and that you are working on a Linux system. Also, be aware that running these commands may require administrative privileges.

```python
#!/usr/bin/env python3
import os
import subprocess

# Function to determine the RAID level
def get_raid_level(devices):
    for device in devices:
        try:
            output = subprocess.check_output(['mdadm'
            if b'RAID Level : raid' in output:
                level = output.split(b'RAID Level : r
                return level.decode()
        except subprocess.CalledProcessError:
            pass
    return None

# Function to create a disk image
def create_disk_image(raid_device, image_file):
    try:
        subprocess.run(['dd', f'if={raid_device}', f'
        print(f'Disk image created successfully: {ima
    except subprocess.CalledProcessError as e:
        print(f'Error creating disk image: {e}')

# List of external SSD drives (replace with your SSD
devices = ['/dev/sdb', '/dev/sdc', '/dev/sdd', '/dev/

# Determine the RAID level
raid_level = get_raid_level(devices)
if raid_level:
    print(f'Detected RAID level: {raid_level}')
else:
    print('Could not determine the RAID level. Exitin
    exit(1)

# Assemble RAID
try:
    subprocess.run(['mdadm', '--assemble', '--run', '
except subprocess.CalledProcessError as e:
    print(f'Error assembling RAID: {e}')
    exit(1)

# Create the disk image
output_image_file = 'raid_disk_image.img'
create_disk_image('/dev/md0', output_image_file)

# Stop the RAID array
subprocess.run(['mdadm', '--stop', '/dev/md0'])
```

Replace the 'devices' list with the device names of your external SSD drives. The script will first determine the RAID level and then assemble the RAID array. After that, it creates a disk image using the 'dd' command and stores it in the 'raid_disk_image.img' file. Finally, it stops the RAID array. Make sure to handle the output image file carefully, as it contains the acquired data.

Is that the most optimised way of doing it?

The script provided in my previous response demonstrates a simple approach for acquiring a disk image from a RAID array of unknown level. However, there are more optimized and specialized tools available to perform this task in a more efficient and forensically sound manner. One such tool is Guymager, a free forensic imager for media acquisition.

Guymager creates a forensically sound disk image while providing features like multi-threading, hashing, and compression. It also supports imaging multiple devices in parallel. Guymager supports both local drives and drives in RAID arrays, making it a suitable choice for your use case.

To install Guymager on Ubuntu, you can follow these steps:

1. Add the repository:

```bash
sudo add-apt-repository ppa:upubuntu-com/guymager
```

2. Update the package list:

```sql
sudo apt-get update
```

3. Install Guymager:

```arduino
sudo apt-get install guymager
```

Once installed, you can run Guymager by typing '**guymager**' in the terminal or searching for it in the applications menu. With Guymager, you can select the RAID device or individual drives and create a disk image in a more optimized and forensically sound manner.

For other Linux distributions or more information, visit the Guymager project page:
https://sourceforge.net/projects/guymager/

Please note that if you are working with hardware RAID, you might need additional tools or drivers to access the RAID array. In this case, consult the RAID controller manufacturer's documentation for further guidance.

Can you integrate the usage of guymager into the script up above?

Integrating Guymager into the script provided earlier requires installing Guymager and then using it via the command-line interface instead of the GUI. Here's an updated version of the script that uses Guymager for disk imaging:

```python
#!/usr/bin/env python3
import os
import subprocess

# Function to determine the RAID level
def get_raid_level(devices):
    for device in devices:
        try:
            output = subprocess.check_output(['mdadm'
            if b'RAID Level : raid' in output:
                level = output.split(b'RAID Level : r
                return level.decode()
        except subprocess.CalledProcessError:
            pass
    return None

# Function to create a disk image using Guymager
def create_disk_image(raid_device, image_file):
    try:
        subprocess.run(['guymager', '--source', raid_
        print(f'Disk image created successfully: {ima
    except subprocess.CalledProcessError as e:
        print(f'Error creating disk image: {e}')

# List of external SSD drives (replace with your SSD
devices = ['/dev/sdb', '/dev/sdc', '/dev/sdd', '/dev/

# Determine the RAID level
raid_level = get_raid_level(devices)
if raid_level:
    print(f'Detected RAID level: {raid_level}')
else:
    print('Could not determine the RAID level. Exitin
    exit(1)

# Assemble RAID
try:
    subprocess.run(['mdadm', '--assemble', '--run', '
except subprocess.CalledProcessError as e:
    print(f'Error assembling RAID: {e}')
    exit(1)

# Create the disk image
output_image_file = 'raid_disk_image.img'
create_disk_image('/dev/md0', output_image_file)

# Stop the RAID array
subprocess.run(['mdadm', '--stop', '/dev/md0'])
```

In this updated script, the `**create_disk_image**` function now uses Guymager for disk imaging with the `**--no-gui**` option. This option tells Guymager to run in the command-line mode. The script assembles the RAID array, then calls Guymager to create a disk image using the raw format and SHA-256 hash for verification. Finally, it stops the RAID array.

Please note that you will need to install Guymager and its command-line interface dependencies as mentioned in my previous response before running the updated script.