Recasting software binaries as "byteplot" and "bigram-dct" images

Install pip packages in the current Jupyter kernel

To install `pip` packages in the current Jupyter notebook and to run `pip` version associated with the current Python kernel, run the following code block that installs all the necessary packages listed in the `requirements.txt` file.

In [ ]:
```
import sys
!{sys.executable} -m pip install -U pip
!{sys.executable} -m pip install -r requirements.txt
```

Import encrypted functions using SOURCEdefender

SOURCEdefender uses password and salt to encrypt the Python source code. For the decryption, these need to be provided.

In [ ]:
```
import sourcedefender
from os import environ
environ["SOURCEDEFENDER_PASSWORD"] = "zQ9bsfAYFrZspCMd"
environ["SOURCEDEFENDER_SALT"] = "YGzqevT7JTJj6meV"
```

The corresponding Python functions from the encrypted files (`*.pye`) can be read as usual.

In [ ]:
```
from get_byteplot_image import *
from get_bigram_dct_image import *
```

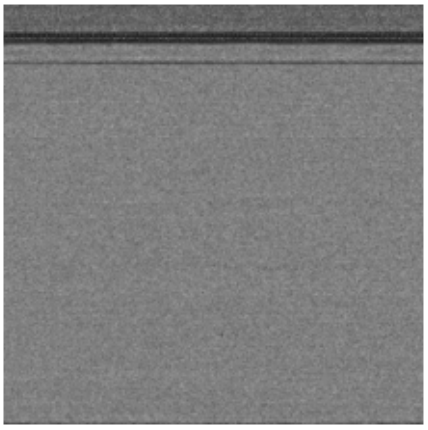
Sample test run to get "byteplot" and "bigram-dct" representations

The provided sample file, `winrar-x64.exe` is an application setup file for WinRAR, and and can be safely considered a *clean* file (VirusTotal report).

In [ ]:
```
f = "winrar-x64.exe"

# Get the "byteplot" representation of the binary file
img_byteplot = get_byteplot_image(f, img_size=256)  # numpy array of dimension (256,

# Get the "bigram-dct" representation of the binary file
img_bigramdct = get_bigram_dct_image(f)  # numpy array of dimension (256, 256)
```

The image representations can be displayed using `matplotlib`.

In [ ]:
```
%matplotlib inline
from matplotlib import pyplot as plt

# Plot "byteplot" representation
plt.imshow(img_byteplot, cmap="gray", vmin=0, vmax=255)
plt.axis("off")
plt.show()
```



In [ ]:

```
# Plot "bigram-dct" representation
plt.imshow(img_bigramdct, cmap="gray", vmin=0, vmax=255)
plt.axis("off")
plt.show()
```