INTERACTIVE HANDS-ON LAB
On DATA COLOURING

1. Open a terminal window
2. Download image to be coloured from
   http://indico.ictp.it/event/8561/session/9/contribution/34/material/0/0.jpg

3. Open a text editor and type in the following script; then save in a file called **fcg.sh**

```
#!/bin/sh
echo "Enter file-name with data to be coloured [Enter]: "
read name
echo "Enter password key [Enter]: "
read key1
echo "Enter private key file of data-owner [Enter]: "
read key2
echo "Enter public key file of recipient/could-service [Enter]: "
read key3
type=$(file -b --mime-type "$name")
filemd5=$(openssl md5 "$name" | cut -f2 -d' ' | cut -c1-25)
key2md5=$(openssl md5 "$key2" | cut -f2 -d' ' | cut -c1-25)
key3md5=$(openssl md5 "$key3" | cut -f2 -d' ' | cut -c1-25)
drops=$filemd5$key2md5$key3md5
dropsfile="/tmp/$$"
echo "$drops" > "$dropsfile"
coloured=$(dirname "$name")/coloured-$(basename "$name")
if [ "$type" = "image/jpeg" -o "$type" = "image/pnm" -o "$type" = "application/pnm" ]
 then
        outguess -k "$key1" -d "$dropsfile" "$name" "$coloured"
 else
        echo "Not supported file format"
fi
if [ -s "$coloured" ]
 then
        echo "Data was successfully coloured and saved to file \"$coloured\""
 else
        echo "Colouring was NOT successful: Something went wrong."
fi
```

4. Change permission (make it executable) for the file using the command:

```
chmod +x fcg.sh
```

NOTE: The forward colour generator (fcg.sh) command-line shell-script generates the colour drops and uses them for colouring the original data.

5. Open a text editor and type in the following script; then save in a file called **bcg.sh**

```
#!/bin/sh
 echo "Enter file containing COLOURED data [Enter] :"
 read coloured
 echo "Enter file containing ORGINAL (UNCOLOURED) data [Enter] :"
 read name
 echo "Enter password key [Enter]:"
 read key1
 echo "Enter file with PKI private-key of data-owner [Enter]: "
 read key2
 echo "Enter file with PKI public-key of recipient user/cloud-service [Enter]: "
 read key3
 type=$(file -b --mime-type "$coloured")
 filemd5=$(openssl md5 "$name" | cut -f2 -d' ' | cut -c1-25)
 key2md5=$(openssl md5 "$key2" | cut -f2 -d' ' | cut -c1-25)
 key3md5=$(openssl md5 "$key3" | cut -f2 -d' ' | cut -c1-25)
 drops=$filemd5$key2md5$key3md5
 dropsfile="drops-$$"
 echo "$drops" > "$dropsfile"
 name2="${name}.txt"
 if [  "$type" = "image/jpeg" -o "$type" = "image/pnm" -o  "$type" = "application/pnm" ]
  then
        outguess -k "$key1" -r "$coloured" "$name2"
  else
        echo " $type is an unsupported File type/format"
 fi
 if [   -s "$name2"  -a    -s   "$dropsfile"   ]
  then
         echo "Extracted drops=\"$name2\""
         echo "Generated drops=\"$dropsfile\""
         diff  "$name2" "$dropsfile"
         # the diff command returns 0 if there is a match and 1 if otherwise
          if [ $? -eq 0 ]
          then
                echo "Extracted and generated colour drops  match"
                echo "DIGITAL FINGERPRINT VERIFICATION PASSED"
          else
               echo "Extracted drops do NOT match generated drops"
               echo "DIGITAL FINGERPRINT VERIFICATION FAILED"
           fi
  else
         echo "Unidentified drops from Coloured image"
         echo "DIGITAL FINGERPRINT VERIFICATION FAILED"
 fi
```

6. Change permission (make it executable) using the command:

```
chmod +x bcg.sh
```

NOTE: The backward colour generator script (bcg.sh) shell script extracts colour drops from a coloured file and compares with colour drops generated directly from the input parameters.

7. Creating your personal asymmetric keys

```
ssh-keygen -t rsa
```

Just press the enter key at passphrase request

8. Install Outguess if needed

```
sudo apt -get install outguess
```

The above command is for Ubuntu Linux, please adjust for other operating systems.

9. Running the scripts

At the terminal run the following commands

```
./fcg.sh
./bcg.sh
```

10. Test integrity detection by modifying a copy of the coloured file:

At the prompt give the following command

```
cp "File1.jpg" "File2.jpg"
```

Modify file using gimp and save

```
gimp   “File2.jpg”
```

Run ./bcg.sh script (as in step 9) but passing in File2.jpg as file with coloured data.

NOTE: A GUI/Picture viewer such as okular can be used to see that "File2" presents same image as File1. Even though the file has been modified it still looks the same.

Original sample image was obtained August 2018 from: https://upload.wikimedia.org/wikipedia/commons/6/6a/Mona_Lisa.jpg