Task 2 - Social Engineering

(Computer Forensics, Metadata Analysis, Encryption Tools)

Points: 40

The name of the encrypted file you found implies that it might contain credentials for the journalist's various applications and accounts. Your next task is to find information on the journalist's computer that will allow you to decrypt this file.

Provided files

- Archive of data from jouralist's computer (for tasks 1-3) (`home.zip`)

Prompt

- Enter the password that decrypts the encrypted file.

Prerequisites

```
~$ apt install exif python3
```

Solution

We begin by inspecting the file `pwHints.txt` to look for clues.

```
~/home/JadeOrchids745$ cat pwHints.txt
bank: anniversary + pet name
email: favorite band + favorite painter + sister's bday
house alarm: favorite dessert
keychain: pet name + pet bday
login: mom's bday + dad's nickname
music: first date + favorite dessert
work server: pet name + anniversary
blog: favorite color + pet name
stepinator: color + petName + anniversary +fdate
```

The password hint for the journalist's `keychain` implies that the password to decrypt `credentials` consists of the journalist's pet's name and birthday.

We begin by looking through the journalist's blog articles.

```
~/home/JadeOrchids745$ cd Documents/Blog-Articles
~/home/JadeOrchids745/Documents/Blog-Articles$ cat blogIntro.txt
Journalist by day. New Blogger by night. Traveler when able. Pet lover...always.

Jade here. Welcome to the 'Diary of an Exotic Furry Voyager'. Outside of work, my two favorite things are
traveling the world and getting to come home to my favorite furry little friend, and the best friend on the
planet, Zara. This blog is going to account for some of my travels and more importantly, the animals I meet on
the way. Hope you enjoy!
```

The journalist mentions coming home to his favourite furry friend `Zara` so that would be his pet's name. The blog articles have no mention of Zara's birthday.

Next, we look through the journalist's pictures, specifically those under the `Pets` directory.

```
~/home/JadeOrchids745$ cd Pictures/Pets
```

```
~/home/JadeOrchids745/Pictures/Pets$ ls
couchChillin.jpg  loaf.jpg  shenanigans.jpg
```

All three of these files are cat pictures, presumably of Zara. Moreover, `shenanigans.jpg` seems to be a birthday picture.

We may therefore conjecture that this photograph was taken on Zara's birthday. We inspect the picture's metadata.

```
~/home/JadeOrchids745/Pictures/Pets$ exif shenanigans.jpg
EXIF tags in 'shenanigans.jpg' ('Intel' byte order):
--------------------+------------------------------------------------------------
Tag                 |Value
--------------------+------------------------------------------------------------
X-Resolution        |72
Y-Resolution        |72
Resolution Unit     |Inch
Date and Time (Origi|2019:09:10 19:39:12
Date and Time (Digit|2019:09:10 19:39:12
Color Space         |sRGB
Pixel X Dimension   |2100
Pixel Y Dimension   |1500
Exif Version        |Exif Version 2.1
FlashPixVersion     |FlashPix Version 1.0
--------------------+------------------------------------------------------------
```

We see that the photograph was taken on 10 Sep 2019.

We create a list of possible strings the password could be. We consider a password consisting of three strings: the pet's name, birth day, and birth month. We consider the two standard capitalisations of the pet's name: `Zara` and `zara`. We then consider the different ways of representing the month September: `Sep`, `sep`, `09`, and `9`. We also account for the fact that the day may come before or after the month. Finally, we consider a few possible characters that may delimit these three strings: `''`, `'-'`, and `' '`.

We can implement a quick Python script to generate a wordlist of possible passwords.

```
#!/usr/bin/python3
# make_wordlist.py
import sys

with open(sys.argv[1], 'w') as f:
    for name in ['Zara', 'zara']:
        for delim1 in ['', '-', ' ']:
            for month in ['Sep', 'sep', '09', '9']:
                for delim2 in ['', '-', ' ']:
                    print(name+delim1+month+delim2+'10\n'+name+delim1+'10'+delim2+month, file=f)
```

We can now use this script to generate our `wordlist`.

```
~$ ./make_wordlist.py wordlist
```

In the previous task, we saw that the `credentials` files was symmetrically encrypted with GPG using AES256 as the cipher. We may therefore create a simple Bash script to soft-brute-force the GPG passphrase using the `wordlist` we generated.

```
#!/usr/bin/bash
# crack.sh

while read password; do
   echo -n "Trying: $password... "
   gpg --batch --no-tty -o $2 --cipher-algo AES256 --passphrase $password -d $1 &> /dev/null
   if [ $? -eq 0 ]; then
      echo 'SUCCESS'
      exit 0
   else
      echo 'FAILED'
   fi
done < $3
```

We now run the script.

```
~$ ./crack.sh credentials keychain wordlist
Trying: ZaraSep10... FAILED
Trying: Zara10Sep... FAILED
Trying: ZaraSep-10... FAILED
Trying: Zara10-Sep... FAILED
Trying: ZaraSep 10... FAILED
Trying: Zara10 Sep... FAILED
Trying: Zarasep10... FAILED
Trying: Zara10sep... FAILED
Trying: Zarasep-10... FAILED
Trying: Zara10-sep... FAILED
Trying: Zarasep 10... FAILED
Trying: Zara10 sep... FAILED
Trying: Zara0910... SUCCESS
```

We see that the password `Zara0910` successfully decrypted the file.

Answer

- Enter the password that decrypts the encrypted file.
  - `Zara0910`

Author

- **Aviv Brook**