PV204 Security Technologies

4th assignment – Disk Encryption.

Supervisor: Dr.Brož

Student: Pedro Gomes, 490830

Due date: 18th April

1. Find the password and unlock the volume.

I generated all the 1000 possible passwords with a python script[1].

After that I created a python wrapper[2] for cryptosetup to try to bruteforce the pv204_assignment volume.

Finally, [2] found a match for: pv204_438

2. Describe which cipher chain (names of ciphers) is used for the volume.

The ciphers used are: *serpent-twofish-aes*
Cipher mode: *xts-plain64*
aes is the cypher most widely used. According to [3] none of the ciphers is broken, but because aes is the most widely used it may get patches faster in case of compromise.

3. Write in hexa format volume salt (for the primary TrueCrypt header, from the first sector).

Taking in consideration that the encrypted is:

b109f3bbbc244eb82441917ed06d618b9008dd09b3befd1b5e07394c706a8bb980b1 ||||| *d7785e5976ec049b46df5f1326af5a2ea6d103fd07c95385ffab0cacbc86 ;* and that the first 64bytes are used for the salt, the salt is:

0	B1 09	1011000100001001
2	F3 BB	1111001110111011
4	BC 24	1011110000100100
6	4E B8	0100111010111000
8	24 41	0010010001000001
10	91 7E	1001000101111110
12	D0 6D	1101000001101101
14	61 8B	0110000110001011
16	90 08	1001000000001000
18	DD 09	1101110100001001
20	B3 BE	1011001110111110
22	FD 1B	1111110100011011
24	5E 07	0101111000000111
26	39 4C	0011100101001100
28	70 6A	0111000001101010
30	8B B9	1000101110111001
32	80 B1	1000000010110001

Hence *: B109 F3BB BC24 4EB8 2441 917E … 80B1*

4. Write in hexa master keys for all used ciphers:

```
b1 09 f3 bb bc 24 4e b8 24 41 91 7e d0 6d 61 8b
90 08 dd 09 b3 be fd 1b 5e 07 39 4c 70 6a 8b b9
80 b1 d7 78 5e 59 76 ec 04 9b 46 df 5f 13 26 af
5a 2e a6 d1 03 fd 07 c9 53 85 ff ab 0c ac bc 86

b1 09 f3 bb bc 24 4e b8 24 41 91 7e d0 6d 61 8b
90 08 dd 09 b3 be fd 1b 5e 07 39 4c 70 6a 8b b9
80 b1 d7 78 5e 59 76 ec 04 9b 46 df 5f 13 26 af
5a 2e a6 d1 03 fd 07 c9 53 85 ff ab 0c ac bc 86

b1 09 f3 bb bc 24 4e b8 24 41 91 7e d0 6d 61 8b
90 08 dd 09 b3 be fd 1b 5e 07 39 4c 70 6a 8b b9
80 b1 d7 78 5e 59 76 ec 04 9b 46 df 5f 13 26 af
```

5. Shortly describe (max. 10 sentences) what is wrong with the keys (or salt). Are the keys independent and randomly generated?

We can see that both the primary and secondary keys are equal. Also, the salt used is the same for both keys. The keys should be randomly generated , but they do not seem to be, otherwise they would have been different in our case.

6. Bonus: try to check how the key(s) were generated (hint: just use Google).

The keys are generated using the TrueCrypt random number generator function[3]. It extracts random data from several sources such as: Keystrokes or mouse movements. This data obtained is divided into (32bit format % 28) and put into a Pool.

The keys are generated based on the data on the Pool.

References:

[1]: Pedro Gomes**,** passwords_generator.py, file used to generate pwds, see it attached on the zip file
[2]: Pedro Gomes, pw_cracker2.py, wrapper used to interact with cryptsetup, see it attached on the zip file
[3]: admin, https://www.truecrypt71a.com/documentation/technical-details/random-number-generator/ , TrueCrypt random number generator function.