

# TorComm - Secure P2P Communication

Documentation

Taha Canturk  
kibnakanoto@protonmail.com

2024-05-20

# Contents

<b>1</b>	<b>Key Protector</b>	<b>i</b>
1.1	What Is It . . . . .	i
1.2	Algorithm . . . . .	i
1.3	Security . . . . .	ii

# 1 Key Protector

## 1.1 What Is It

The Key protector app in security folder is used to secure a 32-byte symmetric key, 2-byte port key, 32-byte pepper. The output is in a file named *keys*. The data in this file is used for securing the local data. It needs a 4-32 byte password generated and stored by you.

To set the password, execute the *key* file which would generate the *get\_keys* executable which is the key protector program. Store a copy of *get\_keys* in somewhere secure if you don't want to lose it. If you lose the *get\_keys* and don't have the *keys* file, then your key is forever lost.

## 1.2 Algorithm

The C++ code is in *security/key.cpp*, but the basic idea is as following:

```
1
2 Generate key, pepper, iv
3 Ask user for 4-32 byte password
4 result = pepper ⊕ password
5 Use sha256(result) as symmetric key to encrypt key using chacha
6 Store sha256(result) as sha256(sha256(result))
7 Generate exe for getting key (get_keys):
8     Store sha256(sha256(result)), iv, encrypted key, pepper (excluding 3-bytes)
9
10    Ask user for password:
11        Guess 3 bytes of password of unknown pepper
12        result = pepper ⊕ password
13        Compute sha256(sha256(result)) and compare with stored sha256(sha256(
14        result)).
15        if no match:
16            Continue guessing all possible 3-bytes. Once done, let user guess
17            again
18
19        if user guessed more than once:
20            If guessed 3 or 6 times and while guess count is smaller than 7:
21                Pause for 10s
22            Else if Every 5 guesses:
23                Pause for 30s
24        Sleep(random(1s,5s)) # make it a random range so that timing attacks aren
25        't possible
26
27    If not valid match:
28        If more than 10 password inputs made:
29            Delete everything in current directory
30        Else:
31            Decrypt encrypted key using sha256(result)
32            Write decrypted key to file
```

Figure 1: Key Protector

### 1.3 Security

since 2/3-bytes of the pepper is not stored in the *get\_keys* file, they need to be guessed with every password that is entered. If we say 3 bytes of the data needs to be guessed. then the number of combinations in password is multiplied with  $256^3$ .

e.g. if you have a 4-digit pin as your password, then there are  $10^4$  combinations in your password. Then the total number of combinations in password is  $(256^3)(10^4) = 167772160000$ .

This doesn't mean that your password needs to be smaller, it should still be 6-16 characters of numbers, small/capital letters, and symbols.