Sliicy Documentation

Official documentation for Sliicy

1: Setting up Sliicy

1. Download the appropriate Sliicy version for Windows at https://github.com/Sliicy/sliicy/.
2. Open the Sliicy app.
3. Create a password that you will remember for Sliicy. It is recommended not to use a long password (for reasons, see "Attacks, Section 4").

CAUTION!
Entering an incorrect password will corrupt the conversations used with it. Sliicy does not store your password anywhere. Recovery of a forgotten password is impossible.

4. Anyone you want to contact using Sliicy must be set up properly. It is highly recommended to either personally meet with the contact once in real life, or to mail a USB to the contact containing the keys to initiate communication.
5. If you received a contact file from your friend, skip to step 6. Otherwise, if you are initiating setup:

    a. Select “New Contact”.
    b. Type the name of your friend as well as your own name and press OK. Sliicy will then generate a list of random numbers as well as a randomly shuffled list of words. Two (2) files will be generated.
    c. The file with your name will end in a .slii extension. It should be given to your friend, while the other file with your friend’s name is encrypted by your password and kept by you.

6. If you received a contact file from your friend:

    a. Select “Join Contact”.
    b. Select the file that your friend securely shared with you.
    c. Sliicy will now encrypt the contact file with your password.

7. At this point, the setup is complete. You may want to backup the contact file to somewhere safe in case it is corrupted or lost.

2: Using Sliicy

1. Open the Sliicy app.
2. Enter your password.
3. Select a contact to communicate with.

4. If you are decrypting a message, skip to step 5. Otherwise, if you are encrypting a message:

    a. Enter the desired message.
    b. Select “Encrypt”.
    c. Copy paste the output of the message into whatever form of communication you desire. If you are using SMS, Email, Signal, WhatsApp, or any other instant messaging service, simply paste the encrypted message into it and send to your friend.

5. If you are decrypting a message:

    a. Enter the encrypted message from your friend.
    b. Select “Decrypt”.

CAUTION!
If you see a dialog warning you against opening the message because the signatures don’t match, chances are either the message fully didn’t get sent through, or someone manipulated the message. It is advised <u>NOT</u> to open it. If you choose to open it anyways, be very cautious, and select “No” to the next dialog.

    c. A dialog will ask you if the message looks legitimate. If it looks legitimate, select “Yes”. This will modify your keys for this contact. Otherwise, select “No”, and nothing will be changed. (This dialog prevents spamming, where a hacker would interfere with the communication.)

6. If you want to change the password used to open Sliicy, select “Change Password”:

    a. Select the folder that has all the contact files you use with Sliicy (they are still encrypted with the old password).

b. Type a new password that you will remember.
c. Sliicy will only change contact files in that folder (ending with a .txt and containing the word, “Sliicy”).

3: How Sliicy works

1. Password used for login:

    a. This password is converted from text into a number.
    b. A longer password will yield a larger number. If the password is too large, encryption will be very weak and detectable, from the overall range being significantly different (If the range is 1,000, adding everything by 10,000,000 will expose the real numbers within 10,000,000).
    c. This number is temporarily added to each number stored in the first line of the contact file during operation.
    d. This password is not stored anywhere, so inputting the wrong password can have negative results.

2. Creating a contact:

    a. Two (2) files are generated. In each file, first line is a list of 10,755+ random numbers separated by semicolons. The next lines until the end of the file consist of shuffled lists of words, each in its own grammatical section. Each of these lists are separated by an empty line. Here is an abridged version of a contact file:

Example contact file.txt

```
324;57;-32;77;1001;-28734;-2231;-847

pear
strawberry
grape

Zack
Alice
Tony

New-York
Orlando
Los-Angeles

kick
sip
feel
```

b. The file ending in a .slii is given to the recipient only under physical communication (the internet is unsafe).
c. The recipient joins the conversation and their file is encrypted with their password.

3. Encryption and Decryption:
    a. Sliicy uses forward secrecy to further protect against attacks. This means that every time communication is done, the keys are changed.
    b. If either party encrypts but fails to send a message, or decrypts but doesn't answer "Yes" when the message makes sense, or decrypts a message stating it was forged and anyways answers

"Yes", or modifies the contact file in any way, communication will not be in sync.

c. To remedy this, try to use Sliicy for larger operations rather than a quick response to SMS conversation and the like.
d. The only other way to fix an out of sync problem is to rely on backups from both parties, or to meet in real life and setup again.
e. Encryption and decryption of messages operate mostly in the same way (with a few exceptions):

    i. All words of a message are looked up in the contact file.
    ii. For each word, a new word is picked from the same group (a noun will stay a noun). The new word is picked based upon the type of message, and the numbers in the contact file.
    iii. There is a list of numbers at the top of the contact file. Certain numbers are selected, added together, and form the amount of words to skip by to fetch the new word.
    iv. Factors of the message include the type of word used, the total number of words in the message, the position of the word in the message, and the amount of punctuation used in the message.
    v. Each number in the first line of the contact file pertains to a different operation. The first number is added to all numbers, the second is added to all decimals, the third fourth fifth and sixth are added to each subset of IPv4's respectively, etc… Below is a chart of all numbers' functions:

0	Numbers
1	Decimals

2, 3, 4, 5	IPv4 (Like 10.0.0.1)
6 - 13	IPv6 (Like 2002::1)
14 - 25	Number Signing (0 - 9, . and -)
26 - 58	Symbol Signing (~!@#$% etc)
59 - 110	Letter Signing (A - Z, a - z)
111 - 366	Order (Sequence of words)
367 - 622	Total (Total amount of words)
623 - 654	Symbol Count (How many $#%?.)
655 - x (x = total # of lists)	Types (Noun, Verb, Name…)

vi. Decryption is the same as encryption, except all values are inverted (100 becomes -100). This is to reverse the process of the encryption which can only be successful by the correct recipient.

vii. Numbers used in the message are added to the number corresponding to the total number of words used, the number for the word's position in the message, the number pertaining to the number of punctuation, and the number corresponding to the type of word used. The below chart shows how each number has a substitute:

Each word's position	Example substitute number used
1st	53830
2nd	-2139
3rd	-39481
…	…

Total amount of words used	Example substitute number used
1	-8548
2	7223
3	91658
…	…

viii. Because all the numbers in the contact file are whole numbers, decimals are split in half. The left side of decimals are dealt with as is any other number. The right side is reversed (.07 becomes 70) and added to the corresponding numbers found in the contact file. Finally, the result is reversed again (845 becomes .548) and appended to rejoin the left side which was encrypted separately. If the results of the addition yield a negative number, the minus sign is not removed (unlike math rules which prohibit numbers such as 8.45-, or -19.102-). Rather it is recognized upon decryption as a valid number and is necessary to compute the original numbers.

ix. So as not to confuse new words from regular words, three (3) colons precede all new words in the encrypted output. This prevents accidental generation of real words from the scramble as being recognized as not a new word (if a user enters a new word, “Google” but scrambles to form an actual word, like “Pizzas”, it won’t be recognized as a new word. Therefore, in this case, it would be “:::Pizzas” and recognized as a new word).

x. For all URLs entered into Sliicy, internet is required to look up the corresponding IP address of each URL. The URL is replaced with an IP address.

xi. For each IPv4 address, the subsets are shuffled between 0 - 255 (If the original subset was 74 and the skip was

200, it would now be 18 (18 = 274 - 256). Likewise, the recipient would decrypt it from 18 with a skip of -200 to become 74 [74 = 256 - (18 - 200)]).

xii. For each IPv6 address, the subsets must be converted from hexadecimal to integers. Afterwards, it is treated the same way as IPv4. Finally, it is converted back into hexadecimal and shortened (2002:0:0:0:0:0:0:1 = 2002::1). Brackets [ ] are added to each end as a proper IPv6 address would have.

xiii. Punctuation is not encrypted, because most sentences end with a period. And there is almost no data exposed from punctuation.

xiv. Emoji is currently not encrypted.

f. Signing is done for message integrity. This is accomplished by adding up the total value of the message and appending this value to the end of the message during encryption. The signing number is calculated as follows:

   i. For every word found in the message, it has an index in the contact file. This number will be different per people per conversation. Before encrypting, the first word's value is subtracted from the second word's value which is added to the third's etc… (First word - second word + third - fourth + fifth - sixth…)

   ii. For every number or decimal found in the message, its parts are added to the signature (such that 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, decimal (.), and minus (-) are assigned a different value).

   iii. For every URL, it is converted into an IPv4 or IPv6 address.

   iv. For every IPv4 or IPv6 address, each subset is added to the signature's value.

v. All punctuation remain unencrypted, and each symbol has a random numerical value assigned which verifies if they are tampered with.
vi. For all new words, each of their letters are assigned different values, added together to the signature. Because it is an uncommon word (by not being part of the Sliicy word database), it anyways can be anything (very impossible to predict an abnormal word).

4. Changing a password:

    a. When changing a password, all contact files must be updated (they all were encrypted with the old password).
    b. The new password is set, and all the contact files selected are updated accordingly.
    c. Sliicy immediately accepts the new password and logging out is not required.
    d. Logging into Sliicy is now done with the new password.

5. Storage:

    a. Sliicy stores the contacts files wherever you choose.
    b. Sliicy by default will save new contacts in the same folder that Sliicy was opened from.
    c. It is impossible to recover a forgotten password without a backup or use of the friend’s file decrypted.

4: Attacks

The following are known attacks against Sliicy encryption:

1. MITM (Man-in-the-Middle) attacks
    a. Adding false data
        i. An attacker cannot add false data to messages without correctly guessing the exact signature that would have been generated had that data actually been there. Since each signature can consist of over -9999999 to 9999999, the likelihood is zero. The attacker does not possess the contact file necessary to determine the new value.
    b. Modifying data
        i. An attacker cannot modify a word to become a different word or change the characters of a new word without correctly guessing the new signature. See “Adding false data.”
    c. Removing data
        i. Because each piece of the message plays a vital role in the signature, removing data cannot be done without correctly guessing the new signature. See “Adding false data”.
    d. Forging the signature
        i. An attacker cannot change the signature appended to the message without removing certain data from the message to ensure the new signature is exact. Because the data is unique to each user (the word “cat” can be the 10th word, or the 10000th word), the attacker cannot guess the new signature.
    e. Removing the signature
        i. An attacker cannot remove the signature without the recipient realizing that the message now is unsigned. Users are warned against opening unsigned messages, and the user is alerted of the presence of manipulation or mistake.
    f. Denial of service (DOS)

        i. An attacker may attempt to halt communication by constantly disrupting the messages, but the user will be informed immediately in the following way: Consider Alice and Bob are trying to communicate. Eve successfully stops a message sent from Bob to Alice. Because both Alice's and Bob's keys must stay in sync, when Alice sends a message to Bob (unaware of the disruption), Bob will receive an error because he has a new set of words from when he tried to message Alice. Eve will have therefore alerted Alice and Bob of her presence.

2. Acquisition of the encrypted messages
    a. If an attacker acquires the messages in their encrypted form, there is absolutely no way to ever figure out what was actually being said. A single detail can change a whole story (Red or white, Bob or Alice, up or down, attack or defend, etc). No computer, no matter how fast it will ever be, will be able to predict the entire message.
3. Acquisition of the decrypted messages
    a. If an attacker knows the details of any number of messages sent, they still do not know the values which make the skip become the new message (just as $x = a + b + c + d$ may have infinite combinations which form x). They therefore cannot predict future messages.
4. Acquisition of the contact file
    a. An attacker who infiltrates either computer still has the problem of the password needed to decrypt the contact file. This password is computed into a number, positive or negative, from 0 to however large the password is. The attacker would need to attempt every possible password, and manually read each message under that password to see if the password is correct. This would take a very long amount of human hours and is nearly impossible if the user picked a hard password.
5. Acquisition of the password and contact file and messages sent

    a. An attacker who coerces the recipients or successfully cracks the password of the contact file would still have problems decrypting old messages because of forward secrecy. Unless the user made backups, older encrypted messages can't be recovered.

6. Altering the contact file
    a. An attacker who inputs an incorrect password into the contact file and sends or receives any number of messages will ruin the synchronization of both parties.