Source Projects

Source Cipher

User Manual

Table of contents

About the author

My name is Edgars Zabroda, I’m 27 years old and I’m self taught programmer, software designer and so on. I’m happy to present the Source Cipher written in Java. I was mainly programming on C++, there is a C++ version of my cipher, but I decided not to release it. I decided to take a challenge and write my cipher in Java and it was sort of easy and yet not, but a lot easier than in C++, the C++ version of the cipher was written for windows because I used Windows API.

I personally love to develop stuff like software, mods for games and so on, I was inspired to develop and construct stuff ever since I was a kid and one time I decided to work with computer and I had lots of fun like playing PC games and other stuff like playing with commands on “command.com”. I really didn’t like Windows 95, 98, but when Windows 2000 got released I liked it due to it stability. My first programming language which I learned was C++, I was mainly programming on Windows, I’ve started to program on Windows XP back in 2007.

How I started to program on Java? Well I was first introduced to Java when I started to create applications for Android and till recently I started to learn Java. My C++ knowledge saved me from learning a new programming language from scratch, I mean I know how to use classes, I know how to use variables and so on and progressed fast on Java.

If you want to download the source code of the Source Cipher please visit https://github.com/EdgarsZabroda/Source-Cipher-Java.

Features

Source Cipher Java version has the same features as C++ version and more like:

1. Display encoding progress bar if user decides to view it.
2. Ability to view SCKF encryption key information.
3. When creating SCKF file select if to use a single key in SCKF file or not.
4. Menu has been trimmed a little to exclude unneeded dialogs.
5. GUI (Graphical User Interface) adjusts to operating system’s standards for example: Source Cipher will have a different GUI on Linux than it will have on Windows even though the code is the same.
6. Improved SCKF file structure.
7. Ability to fit an entire file into memory up to 2GB*

*Read F.A.Q. for more information

F.A.Q.

I. Why I get the warning when I check "Fit Target File to RAM" or when I check "Generate single key"?

These warnings are displayed in case you don't know the dangers that come when either of these check boxes are checked. For example: you check the "Fit Target File to RAM" will warn you that the maximum file size that the code can process.

II. How much memory does the cipher use?

The maximum memory the cipher uses is 512MB and minimum 8MB, it depends on the key size, for example: if you use 64 bytes key it'll use 64MB of memory.

III. Why can I only fit maximum 2GB of the target file?

That's a limitation of Java, Java can only allocate 2GB maximum per object. If you try to fit larger file than 2GB the application may crash or you'll get exceptions.

IV. Why should you recommend to use double key encryption key?

Well the cipher's encoding algorithm is based on XOR encryption and when you try to XOR the null byte, the part of the key gets visible and the attacker can guess the key but when you use the double key encryption key, that part of visible key gets encrypted and the attacker will have difficult time guessing the key.

V. What are SCKF files?

SCKF extension of the file is the cipher's encryption key that you use to encrypt or decrypt the file. SCKF means Source Cipher's Key File.

VI. Why do I get asked if I want to view the progress of the encoding operation?

Well it's meant to notify the user if the user wants to view encoding progress or not. Either way you'll get the notification, when the process is completed that it was a successful process or not.

VII. What structure is the cipher's encryption key file?

Please read the appendix if you're interested to know more about the cipher.

VIII. What will happen if I choose not to view the encryption progress?

The file will be encoded in the background and once the process is complete you'll still get notification if the process was successful or not.

IX. Who created this cipher?

It was me, Edgars Zabroda, I don't work for any company, I code for fun. Source Projects is what I call my virtual organization.

X. What algorithm does the cipher use?

The cipher uses standard XOR operator to encrypt and decrypt files. Read the appendix for more information.

XI. What's the maximum file size can the cipher handle?

In chunks almost any size, the maximum file size is maximum value of the 64 bit variable. If you choose to fit the entire file into the memory than it's 2GB.

XII. What's the use of this cipher?

It was mainly created to encode files that the user doesn't want others to see it like diaries, pictures, music, movies and so on. The user creates a key the user can put it on any any media. I suggest not to put the SCKF file on the same computer and carry the file on the USB flash drive so that no one would able to decrypt the file.

XIII. Is this the final build of the cipher, or there will be the future another cipher?

Well I can't answer much what I'm planning about, maybe MD5 hash check feature would be useful to tie key to the specific file.

XIV. Where can I download the cipher's source code?

The cipher's source code's home page is https://github.com/EdgarsZabroda/Source-Cipher-Java

XV. The cipher doesn’t do anything when I checked the “Fit Target File to RAM”, why?

Maybe because the file size is more than 2GB. Try launching the cipher with command “java -jar “Source Cipher.jar” and see if there are no errors.

XVI. What’s the difference between single key and double key SCKF file?

The main difference is that single key SCKF files are much smaller than double key SCKF files. Read the appendix to find out why if you’re really interested.

XVII. Why is the “Encode” and on some dialog’s buttons are disabled?

Well it’s disabled for the reason to prevent errors, the text fields for them to get enabled are empty, to enable them just fill the text.

XVIII. Where to put an issue if I encounter a bug?

Post it here: https://github.com/EdgarsZabroda/Source-Cipher-Java/issues

How to create an Encryption Key

To even begin encrypting stuff you must first create a SCKF. This page will guide you how to do that:

I. Open SCKF file dialog

This is rather simple, there are 2 ways to do that.

Method 1:

Click the create button as shown in image 1:

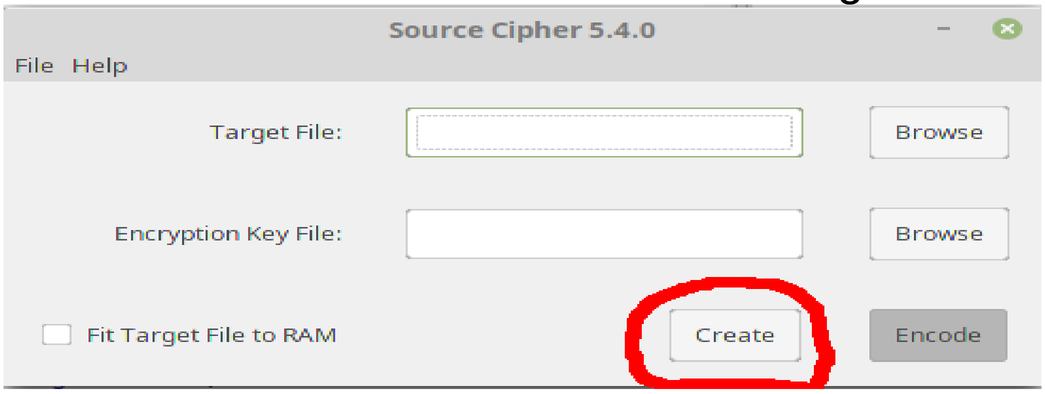


Image 1

Method 2:

I. Press “File” menu as demonstrated in image 2.

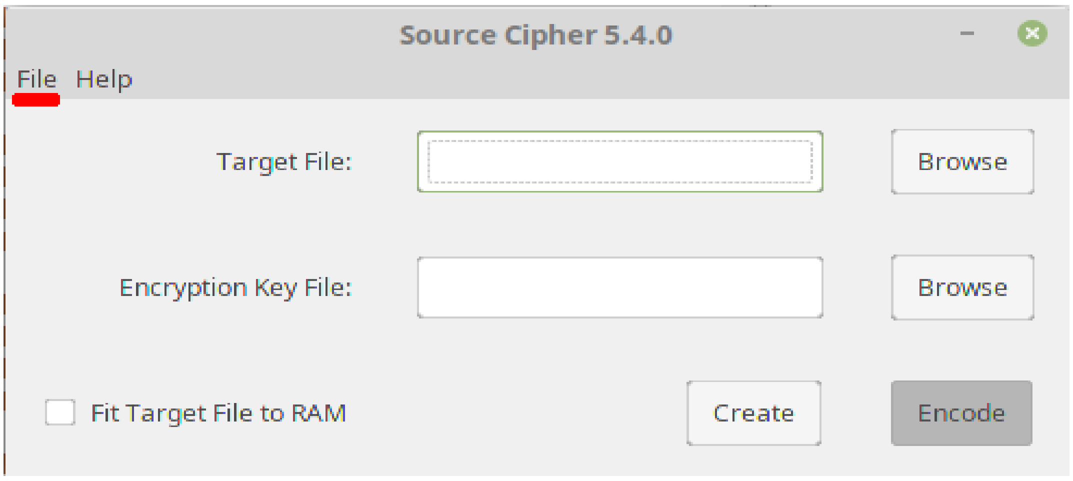


Image 2

II. Select the “Create Encryption Key” menu item.

II. Specify where to save the SCKF file

There are 2 ways to do that and that’s either specifying it manually or let the application to guide you.

Method 1: Entering the path manually:

If you wish to specify the path and file name manually, enter the data as shown in picture 3:

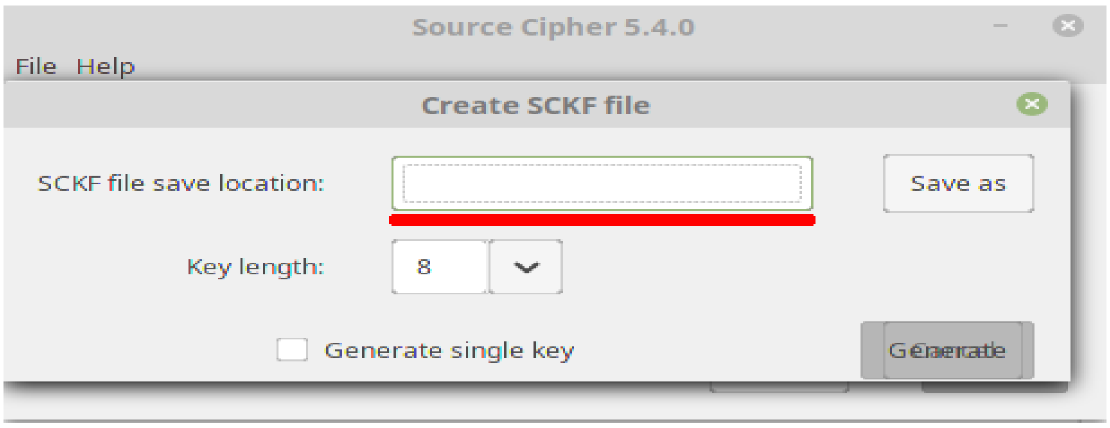


Image 3

Method 2: using application guided method:

If you want the application to guide you where to save the SCKF file click “Save as” button just like in the image 4:

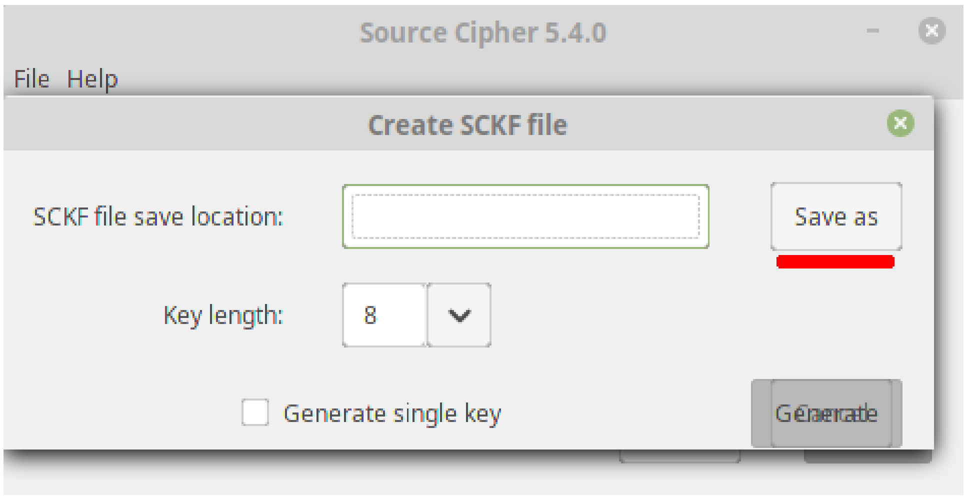


Image 4

III. Select key size

Select the key size that you desire just like in image 5:

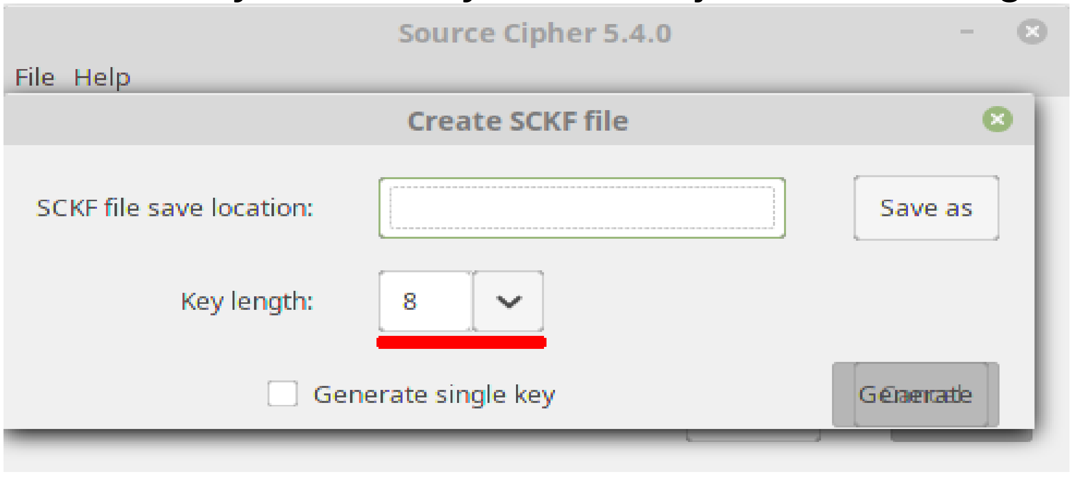


Image 5

IV. Select to generate the single key or not

If you wish to create a secure SCKF file I suggest to skip this step, but if you really want to create a single key instead of double key SCKF file then check the box as shown in image 6:

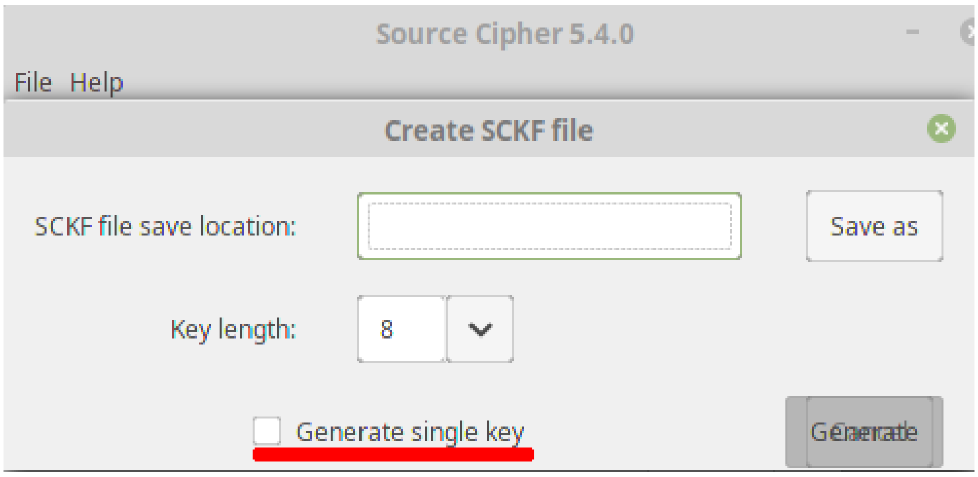


Image 6

V. Click “Generate” button

Once everything is set up the “Generate button should be enabled, press the button and the message box should appear that the SCKF file generation was successful or not.

Encrypting the file

Once you created an SCKF file, you can now encrypt files. This guide will show you how:

I. Select the target file

There are two ways to do that: first is to specify it manually or use the application guided method.

Method 1: Manually specify it:

Enter the path and file location into the field “Target File” like shown in image 1.

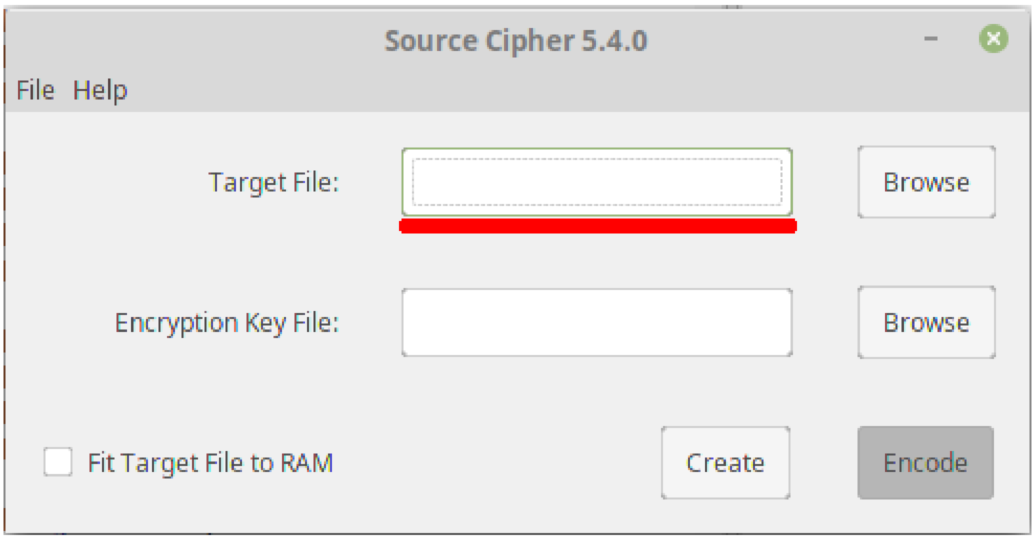


Image 1

Note that this method is highly unrecommended unless you know what are you doing. If you don’t know what your doing use second method.

Method 2: Using application guided method:

This method is highly recommended to use, the application will guide you on your quest of selecting the target file. Click “Browse” button to select the target file, just like in image 2.

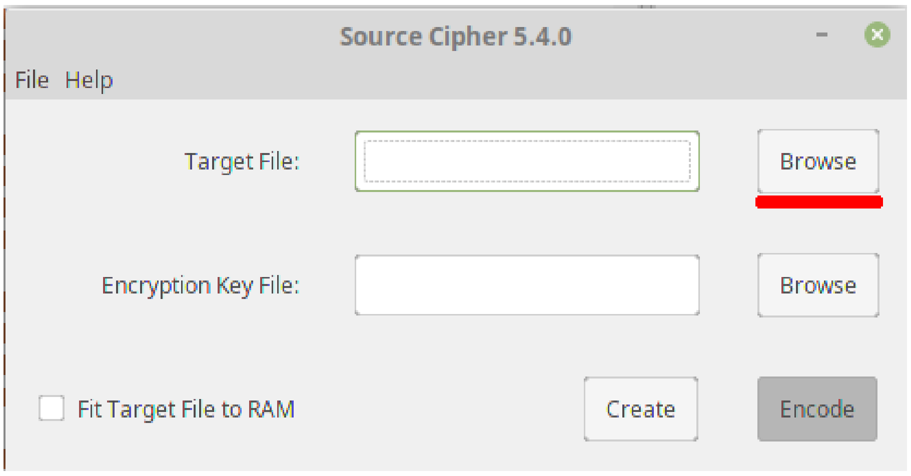


Image 2

II. Select the SCKF file to use for encryption

In order to encode the target file you must specify what SCKF file you'll use to encode the file. There are 2 ways to do that:

1. Specify the SCKF file and it's path manually
2. Specifying the SCKF file by using application guided method.

Method 1: Manually specifying the SCKF file and it's path:

This method is highly unrecommended unless you know what are you doing, but if you do know then enter the SCKF file and it's path into the "Encryption Key File" field just like in image 3.

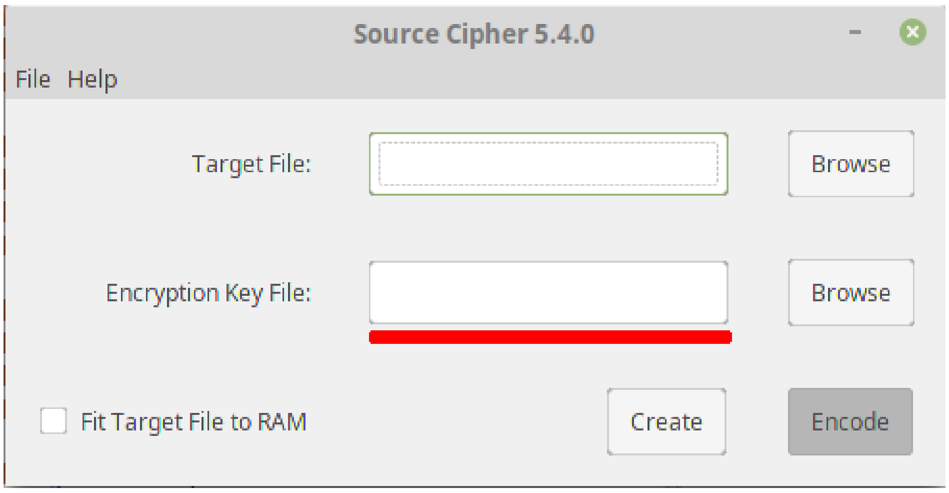


Image 3

Method 2: application guided method:

This method is highly recommended, not only does it saves time specifying the path manually, but also guaranties that the SCKF file and it's path are valid. To use this method click browse button on the "Encryption Key File" field just like in image 4.

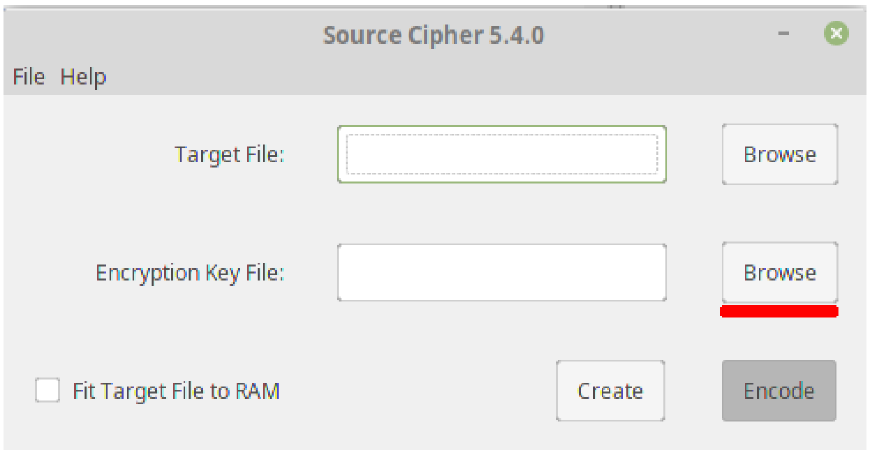


Image 4

III. Check the “Fit Target File to RAM” check box or not

This step is optional and not recommended if the target file is bigger than 2GB. What this does is that it swallows the entire file into RAM and encodes it faster. If this option isn’t selected then the file will be encoded in chunks, read the F.A.Q. section about this feature. If you select this check box you’ll get the warning, the check box is located at the very bottom of the cipher as shown in image 5.

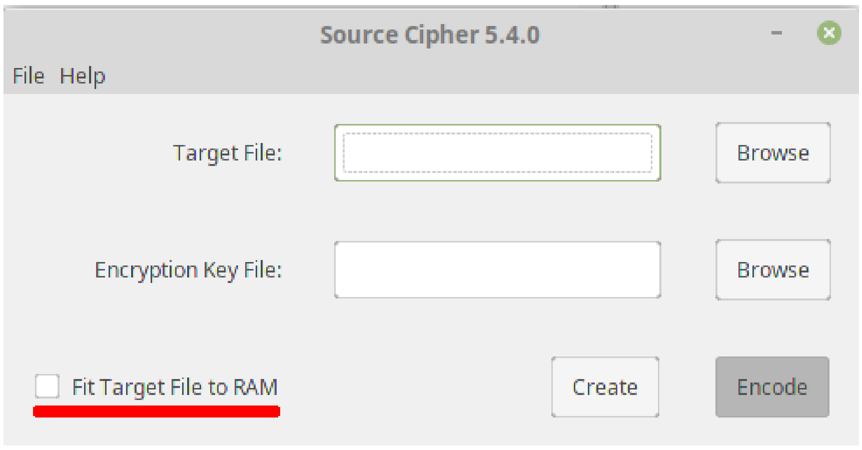
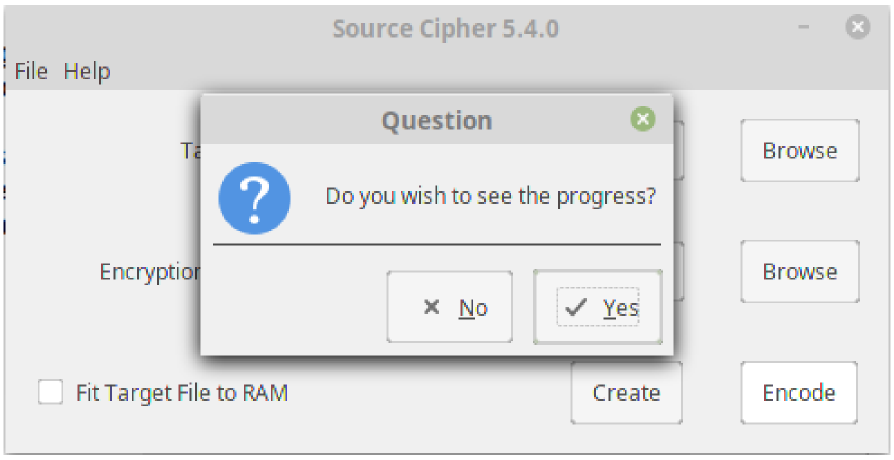


Image 5

IV. Click “Encode” button to encrypt the target file

Once you done everything you’ll notice that the “Encode” button becomes clickable. Once you click it a message box will appear just like in image 6.



Do you wish to see the progress?

Image 6

What this message box does is it asks you if you wish to see the encoding process button. If you choose yes you’ll see the progress bar and “Close” button dialog. While the encoding process is being done you’ll notice that the “Close” button is disabled and it only gets enabled once the encoding process is complete. If you choose no then the encoding process will run in background and if it’ll take a long time to process it you won’t see anything. Don’t close the cipher just yet or you’ll corrupt the target file irreversibly. Once the encoding process completes, you’ll get a message box that reports how the process went, regardless if you choose to see the progress or not.

Now you know how to encrypt the file, if you want to decrypt the file, simply repeat everything that you did to encrypt the file.

How to view SCKF file information

Well you might be curious to find out what's SCKF file content is, that's why I developed this feature. You won't need the hex editor to view the content of the SCKF file, this feature will show you this:

I. Does the selected SCKF file uses single key or double key encryption method.

II. The key size.

III. Primary key content.

IV. If SCKF file uses double key encryption method then the Secondary key will be shown.

To view the SCKF file content follow this steps:

I. Open "SCKF Information" dialog

To open the "SCKF Information" dialog press "File" menu on the cipher's main window as shown in the image 1.

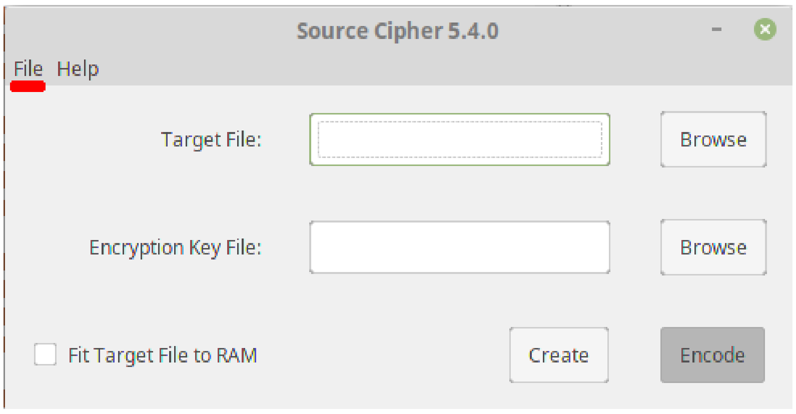


Image 1

Once you press the "File" menu select "View Encryption Key data". This will popup the "SCKF Information" dialog.

II. Select the desired SCKF file

This is done in two ways:

1. Specify the SCKF file and the path manually.
2. Use application's guided method.

Method 1: specifying SCKF file and it's path manually:

This method is not recommended for the users that don't know what their doing, but if you know what you're doing enter the SCKF file name and it's path in the "SCKF File" field as shown in image 2.

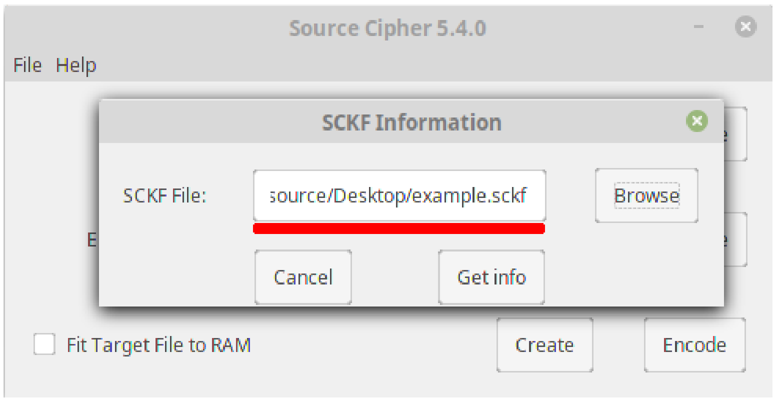


Image 2

Method 2: use the application guided method:

This method is highly recommended, not does it only saves time by typing the data manually but will also guaranty that the data in the text box will be correct.

To use application's guided method simply click browse button as shown in image 3.

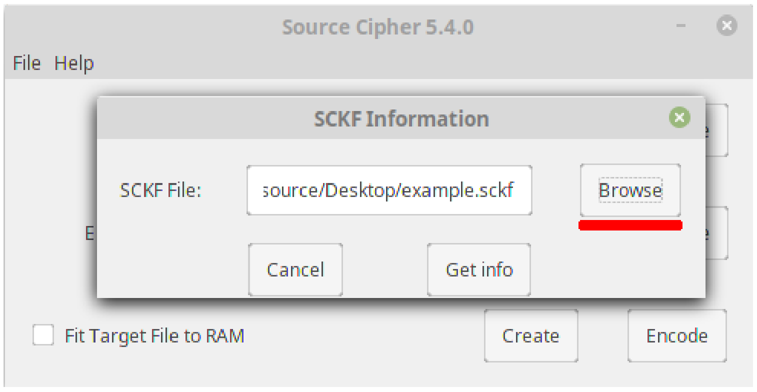
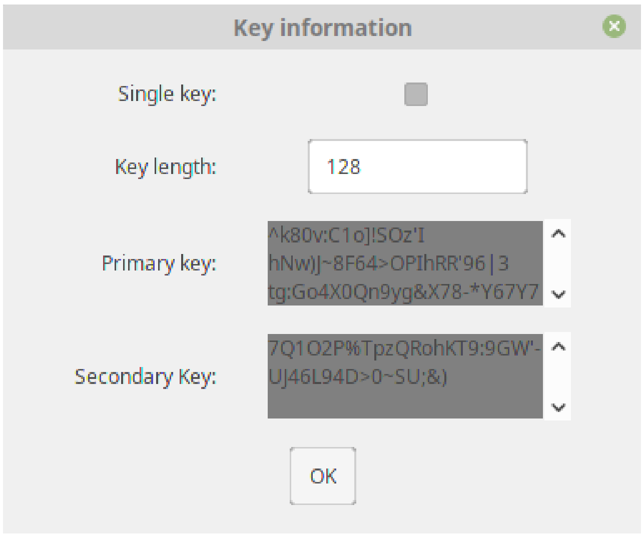


Image 3

III. Click "Get info" button

Once the "SCKF File" field contains data, the "Get info" button becomes clickable. Once you click this button another dialog will popup with the SCKF content just like in image 4.


Key information

Single key: ☐

Key length: 128

Primary key:
```
^k80v:C1o]!SOz'I
hNw)J~8F64>OPIhRR'96|3
tg:Go4X0Qn9yg&X78-*Y67Y7
```

Secondary Key:
```
7Q1O2P%TpzQRohKT9:9GW'-
UJ46L94D>0~SU;&)
```

Image 4

No need for hex editor to view the SCKF file content's and wondering where everything is. But if you wish to SCKF file content's in hex editor read the "Appendix" the "SCKF file structure".

The idea of writing the cipher in Java

Well personally I was wondering how to create a GUI application in Java, so I opened google and searched for “Java GUI programming for begginers” and found one interesting youtube channel of one user, I checked his videos and started learning from his tutorials. He’s tutorials were a bit buggy, for example: I was writing one GUI program and the controls didn’t show up until I resize the main window. Once I learned what I needed I started “Source Cipher” project. I was learning a lot while I was programming.

The IDE I used for creating the cipher was “Eclipse”, I knew “Eclipse” for a long time, but never used it in detail much. Good thing that my C++ knowledge saved me from learning everything from scratch and that Java had the same syntax as C++ or else I would have to learn this programming language from start. Now that I got accustomed to the “Eclipse” I loved it very much because it’s a very powerful IDE. I used google’s version Android’s SDK “Eclipse” variant, I was programming in Java and made a few APKs, but now I forgot how to code for android.

So the Java version of the cipher was somehow a challenge that I took. Programming the encryption algorithm was easy, but the GUI part was making me to google for information and now if I forget something I can use the cipher’s source code as a “pointer” (C++ slang) that “points” me to the part where I can remember where I forgot something.

It was fun developing the cipher in Java. I accidentally learned how to make a JAR executable file that Java uses to execute it’s code. I also learned how to create multiple packages in java and import them to another java file which was located in the other package.

About the cipher

Cipher was based on it’s C++ brother. The Java version of the cipher has been striped from unnecessary dialogs and putted into this manual’s F.A.Q. section. The SCKF file has been improved, for example the C++ used 4 bytes of the bit which tells the cipher if the key is single or double, not like Java version where Java version uses 1 byte, so technically you can’t use C++ variant’s SCKF file. Also C++ version was restricted to Windows, unlike the Java version that can be used on multiple operating systems.

Also it was easier to design GUI in code, unlike when I use C++ version where I had to specify the coordinates of each control, I used Visual Basic form designer and copied the coordinates into C++. But in Java I create everything on the grid and drawing the GUI was much easier.

Why I created the cipher in the first place? Well I want to protect some files that I don’t want anyone to see. I could use PGP, but when I was interested in encryption way before I started programming Java. I will port the C++ version of the cipher into Linux OS. The Java version of the cipher was programmed in Windows and in Linux.

If you ask me what I like more, C++ or Java, I would answer that I like them both, although Java imposed some limits that aren’t putted in C++ for example:

1. Java doesn’t have unsigned option that I assign on the C++ variables.

2. No multiple inheritance. In Java you subclass the class and subclassed class gets public and protected data.

3. There are no friend classes that allows to get other class’s private data.

But here what I liked about Java:

` 1. No need to program a destructor since Java has garbage collector and I don’t need to worry about memory leaks.

2. Java has builtin byte type variables, but yet they’re signed.

3. Long and Int type variables in C++ are the same size, but not in Java, Int is 32 bit long and Long is 64 bit Long.

4. You don’t need to deallocate the memory, Java does this automatically.

5. Java is pure OOP programming language, unlike C++ is hybrid language.

Even with these feature additions and features restriction doesn’t make me love one programming language over the other, I love these programming languages equally. So now you either compiled the source code of the cipher or downloaded it already compiled and used help to extract this guide. I used HxD to copy it as Java and hard coded this guide into this cipher ;).

The guy’s youtube channel that I mentioned before is:
https://www.youtube.com/channel/UCeLd7huJYhPHv4bc28dSbPA/videos
and the other guy’s channel I forgot where I putted it.

SCKF file structure

The SCKF file contains very small header which only consists of 3 bytes. Here’s it’s structure:

Header

1 Byte.	Tells the cipher that the SCKF file uses either double or single key. If value is 0 then the cipher uses two keys and if the value is 1 then the cipher uses one key.
1 Short (2 bytes).	Tells the cipher how big is the key.

Body

Array bytes of one byte.	The primary key that the cipher uses. The size is set in second part of header.
Array bytes of two bytes.	These are shorts, they are indexes of the primary key array, these indexes shuffles the primary key which becomes a secondary key. If the first byte of the header is 1 then this section doesn’t get generated.

And now you know the structure of SCKF files, I have my own key which is 512 bytes long and uses double encryption. I use my own created cipher, you can’t decrypt the files that easily without to much work and without SCKF file even though you have the source code, you still need SCKF file to decrypt it 100% without damaging integrity of the file.

Credits

Programming, designing and everything that went into development are to me (Edgars Zabroda) head of a virtual organization “Source Projects”.

Big thanks to you Mr. Technoist on youtube for teaching me to even start programming GUI in Java and I forgot your name but thank you for teaching me how to use GridBagLayout class, thank you guys and girls at “Stack Overflow” for sharing the knowledge on how to perform various tasks and solve problems and many other people across the internet for posting your examples.

Without you guys and girls, the Source Cipher wouldn’t exist in Java.