Making symmetric block cipher meta permutating again

Heinrich Elsigan

April 23, 2026

Contents

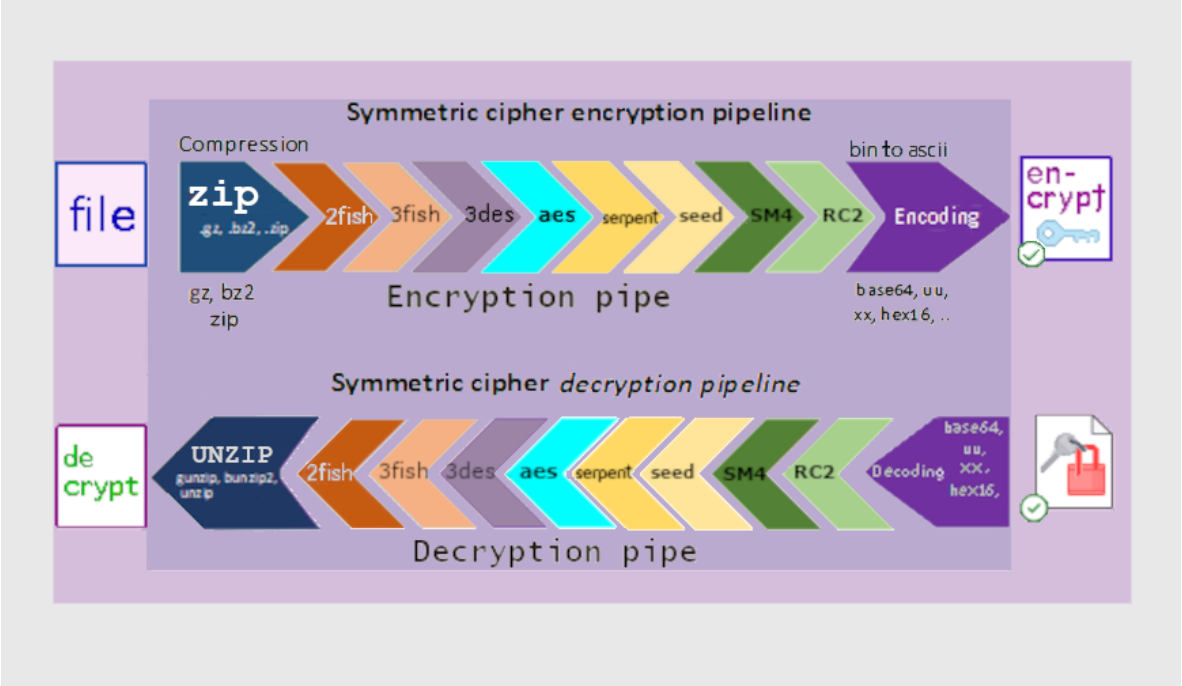

Figure 1: Cipher Pipeline

1 Thanks to

Normally, thanks to are always at the end of each paper, but the people or organizations I benefited from while trying to make AES strong again are more important than a simple proof of concept showing that it works, except on my raw experimental test form.

1. Microsoft C# .Net repositories
   - dot.net, git:dotnet
   - opensource.microsoft.com, git:microsoft
2. Modern crypto libraries and blogs
   - Legion of the Bouncy Castle, git:bcgit
   - libtom.net, git:libtom/libtomcrypt
   - cryptopp.com git:weidai11/cryptopp
   - Bruce Schneier's blog (blowfish,twofish,threefish)
   - GNU Nettle[Els24]

2 Software

Source code of PermAgainCrypt is hosted at Github. Github releases contains compiled and linked .exe files for windows, but most users prefer download from a seperate download site https://cqrxs.eu/.

2.1 Git: PermAgainCrypt minimalistic repository

https://github.com/heinrichelsigan/PermAgainCrypt/

2.2 Download

Go to https://cqrxs.eu/ or https://io.cqrxs.eu/ and choose latest version. You can directly go to the download area https://cqrxs.eu/download/ or https://io.cqrxs.eu/download/. Attention, version before March 2025 have a 3-fish over Aes-Engine encryption bug, because 3-fish uses AES default block size and key length and Bouncy Castle Aes-Engine parameters. It works, but settings AES default engine for 3-fish, isn't so serious and please download version after March 30.

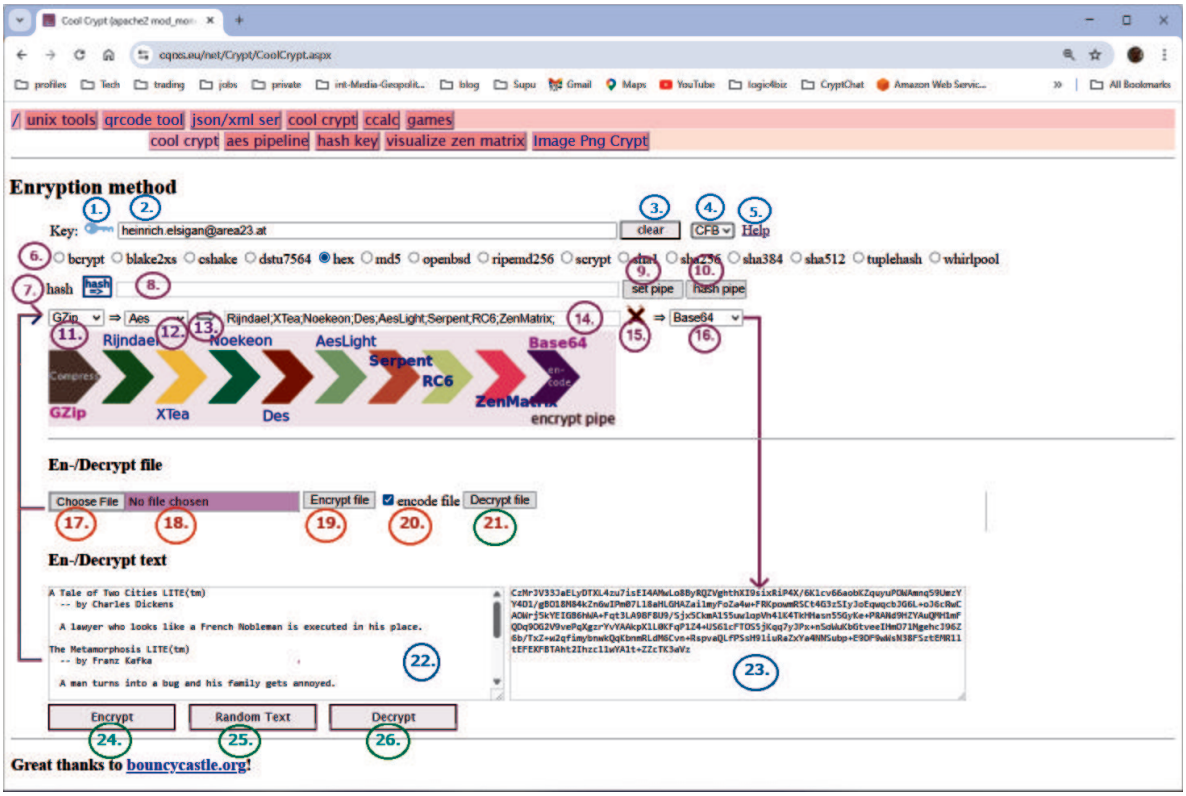

Figure 2: Symmetric Cipher WebForm

3 Different Forms

3.1 Documentation online en-/decrypt form

- cqrxs.eu/net/Crypt
- area23.at/net/Crypt

1. *ImageButton* **Key**: When clicking key your entered key will be stored temporary in session.
2. *Textbox* **secret key**: Enter your Email address and secret key.
3. *Buton* **Clear**: Clear and reset the entire form.
4. *DropDown* **CipherMode2** with Options *CBC*, *CFB* and full deterministic *ECB* (without IV)
5. *Hyperlink* **Help**: Show this help
6. *RadioButtonList* **KeyHashes** Choose the hash method to hash your secret key.
7. *ImageButton* **Hash**: Clicking will hash your key and display hashed key in textbox.
8. *TextBox* **Hashtext** (readonly): Displays your hashed key.
9. *Button* **Set Pipe**: sets symmetric cipher pipe, dependent only on your entered key
10. *Button* **Hash Pipe**: sets symmetric cipher pipe, dependent primary on calculated hash and secondary on your entered key.
11. *DropDown* **ZipTypes** Choose encryption type (Please only GZip or Zip or None)
12. *DropDown* **CipherTypes** Choose a symmetric cipher to add it to the symmetric cipher pipe.

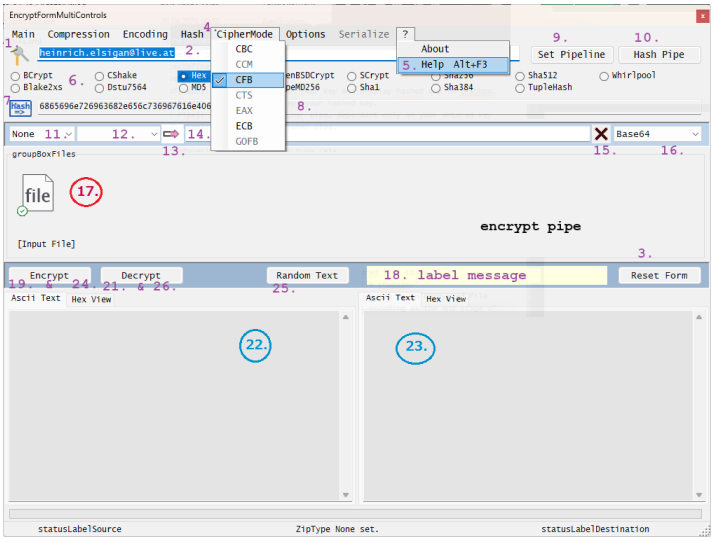

Figure 3: WinForm C#

13. *ImageButton* **add algo**: Clicking on will add the in c. DropDown CipherTypes selected symmetric cipher algorithm to CipherPipe.
14. *TextBox* **CipherPipe** (readonly): Displays the current Cipher Pipe algorithms.
15. *ImageButton* **Clear Pipe**: Clicking on will clear only the entire Cipher Pipe
16. *DropDown* **EncodingTypes**: Choose the final binary to ascii encoder, default is Base64. Beware of using uuencode in Web, because <> will be interpreted as possible html injection.
17. *Selector* **Choose File** to upload a plain or an encrypted file to be processed
18. *Label* **Filename** displays uploaded filename or "No file choosen."
19. *Button* **Encrypt file** encrypts a plain file with cipherpipe to an encrypted file
20. *Chechbox* **encode file** if checked, performs bin2ascii encoding at the end stage of pipeline, if not checked binary data after last stage of cipherpipeline will be written.
21. *Button* **Decrypt file** decrypts an encrypted uploaded file. You must care, that the settings match to the file, when it was encrypted
22. *TextArea* **Source**: paste or enter Text here.
23. *TextArea* **Destination**: (readonly) After clicking Encrypt or Decrypt processed text will appear in the text area.
24. *Button* **Encrypt**: Encrypts text from TextArea Source and displays encrypted text in *TextArea* **Destination**.
25. *Button* **Random Text**: Adds a short fortune to *TextArea* **Source**.
26. *Button* **Decrypt**: Decrypts text in TextArea Source and display decrypted text in TextArea Destination.

3.2 Documentation of WinForm

You can also go directly to github.com/heinrichelsigan/PermAgainCrypt/releases to download newest Gui Form x86 and x64. ...

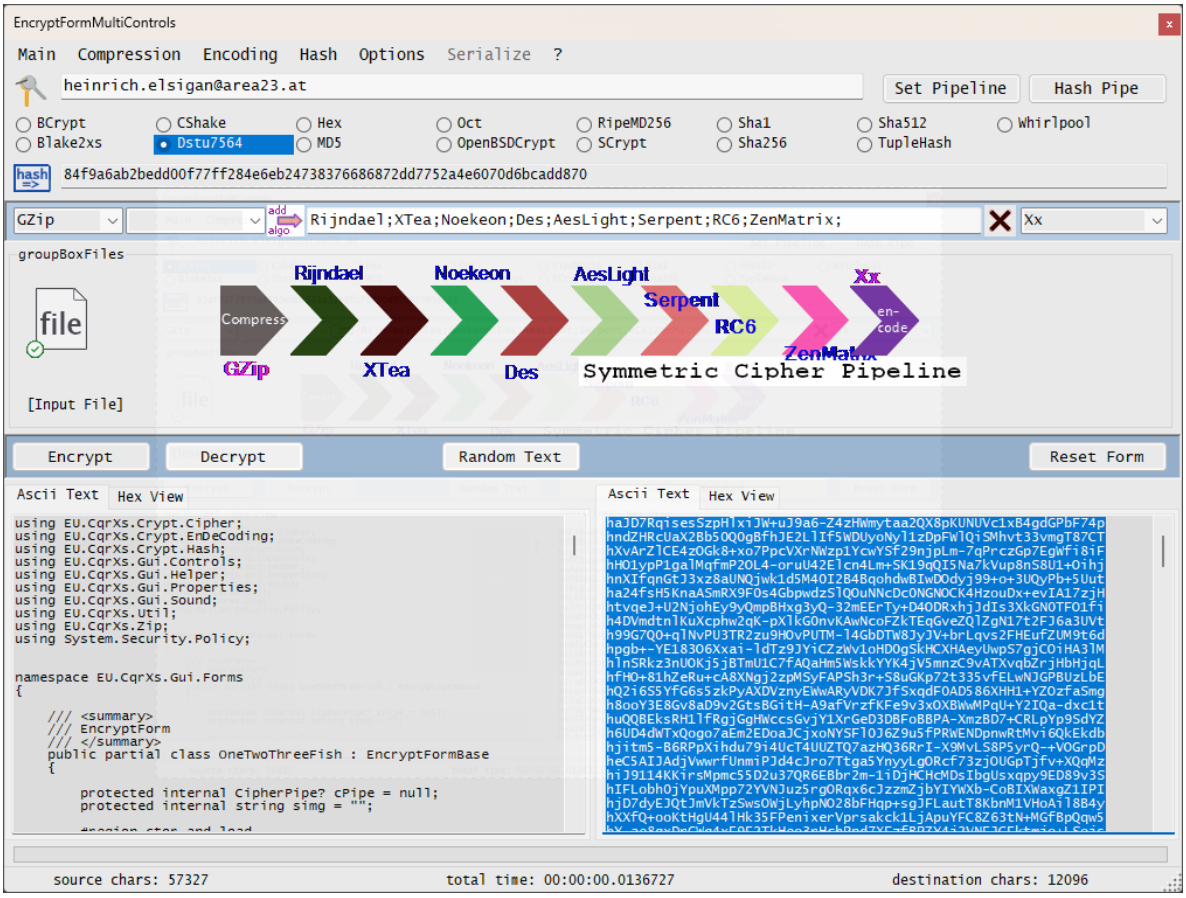

Figure 4: WinForm C#

3.3 Java

3.3.1 Java PreRequesites JDK release date > 2018 installed

1. openjdk-21 to openjdk-26
2. Oracle OpenJDK 26
3. IBM Semeru 26 (AdoptOpenJDK OpenJ9)
4. Amazon Coretto 26
5. BellSoft Liberica JDK 26
6. Eclipse Temurin (AdoptOpenJDK HotSpot)
7. Alibaba Dragonwell 21.0.10
8. GraalVM Community Edition 25.0.2
9. JetBrains Runtime 25.0.2
10. Microsoft OpenJDK 25.0.2

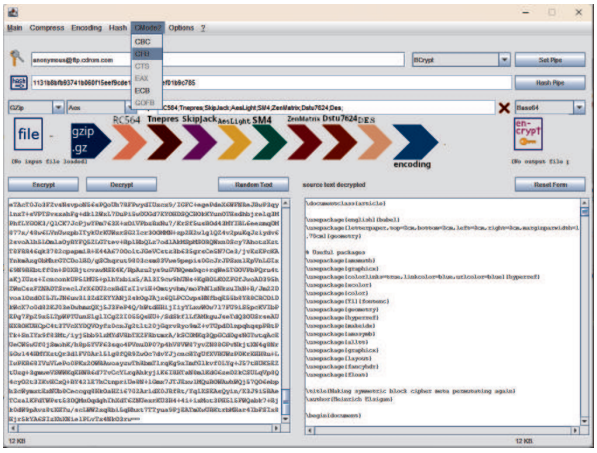

Figure 5: javax.swing.JFrame.java

3.3.2 Java on Windows: Compile, build, launch

Change following lines in winmake.bat depdendent on your JDK installation and bouncy-castle version:

```
echo Setting bouncy-castle jar
SET BCJAR=bcprov-lts8on-2.73.10.jar
echo Setting Path and CLASSPATH
SET Path=%Path%;%USERPROFILE%\.jdks\openjdk-25\bin
SET CLASSPATH=%CLASSPATH%;%USERPROFILE%\.jdks\openjdk-25\lib
SET MYCLASSPATH=%CLASSPATH%;.\;.\%BCJAR%;.\eu\cqrxs\;.\eu\cqrxs\gui\;.\eu\cqrxs\fw\net\;.\eu\cqrxs\fw\util\;.\eu
```

run: winmake.bat
which calls finally

```
java.exe -classpath %MYCLASSPATH% eu\cqrxs\gui\CqrJdFrame.java
java.exe -classpath %MYCLASSPATH% eu\cqrxs\console\CryptConsole.java
```

3.3.3 Java on Linux (similiar on any other Unix or unix based MacOs on Apple)

Change Makefile dependent on your JDK installation and bouncy-castle version, then run make

```
make clean; make all; make launch
```

Optional change these lines in jmake.sh depdendent on your JDK installation and bouncy-castle version: remember: classpath seperator under unix is : and NOT semicolon ;

```
echo "setting bouncy castle jar and MYCLASSPATH"
BCJAR=bcprov-lts8on-2.73.10.jar
MYCLASSPATH="$CLASSPATH:./:./$BCJAR:./eu/cqrxs/:./eu/cqrxs/gui/:./eu/cqrxs/fw/net/:./eu/cqrxs/fw/util/:./eu/cqrx
```

run: sh jmake.sh
which calls finally:

```
JAVAFLAGS=--enable-native-access=ALL-UNNAMED
echo "launching eu/cqrxs/gui/CqrJdFrame.java"
java $(JAVAFLAGS) -cp $(MYCLASSPATH) eu/cqrxs/gui/CqrJdFrame.java

echo "run eu/cqrxs/console/ConsoleApp"
java $(JAVAFLAGS) -cp $(MYCLASSPATH) eu/cqrxs/console/CryptConsole.java
```

3.3.4 Java is JFrame based

- https://docs.oracle.com/javase/8/docs/api/javax/swing/JFrame.html
- https://github.com/openjdk-mirror/jdk7u-jdk/blob/master/src/share/classes/javax/swing/JFrame.java

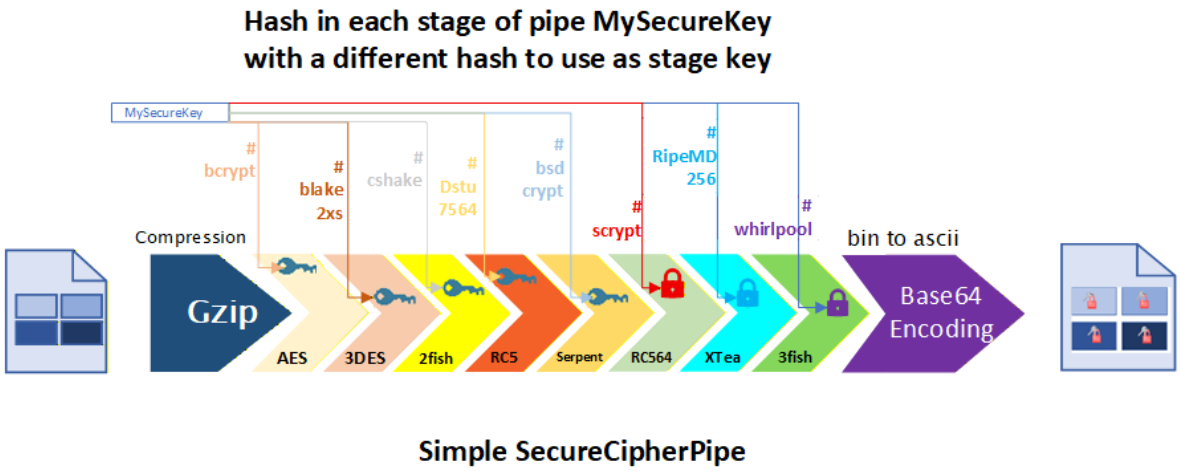

Figure 6: Simple SecureCipherPipe

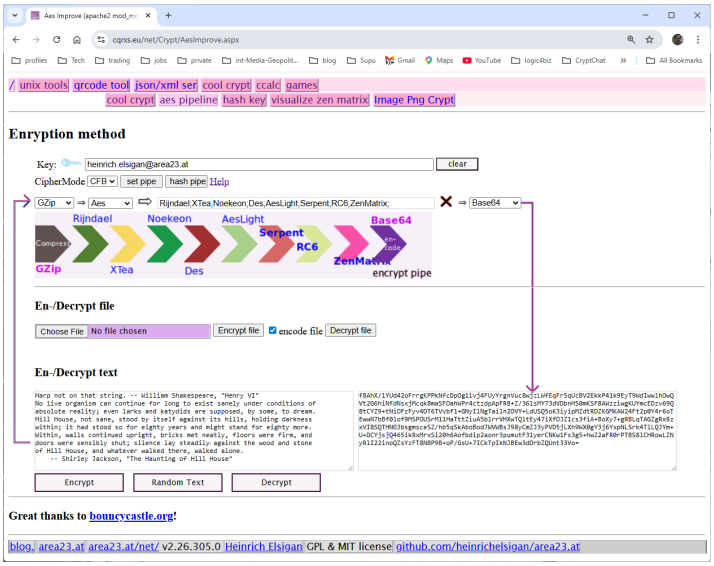

Figure 7: WebForm AesPipeline Simple

3.4 Simple mode - SecureCipherPipe

In the simple mode SecureCipherPipe is used as CipherPipe. Each stage of the pipe will not be encrypted with the same secure key; furthermore, each pipe stage will be encrypted with a secure hashed key with a different hash.

WebForm for simple mode is https://area23.at/net/Crypt/AesImprove.aspx or https://cqrxs.eu/net/Crypt/AesImprove.aspx

...

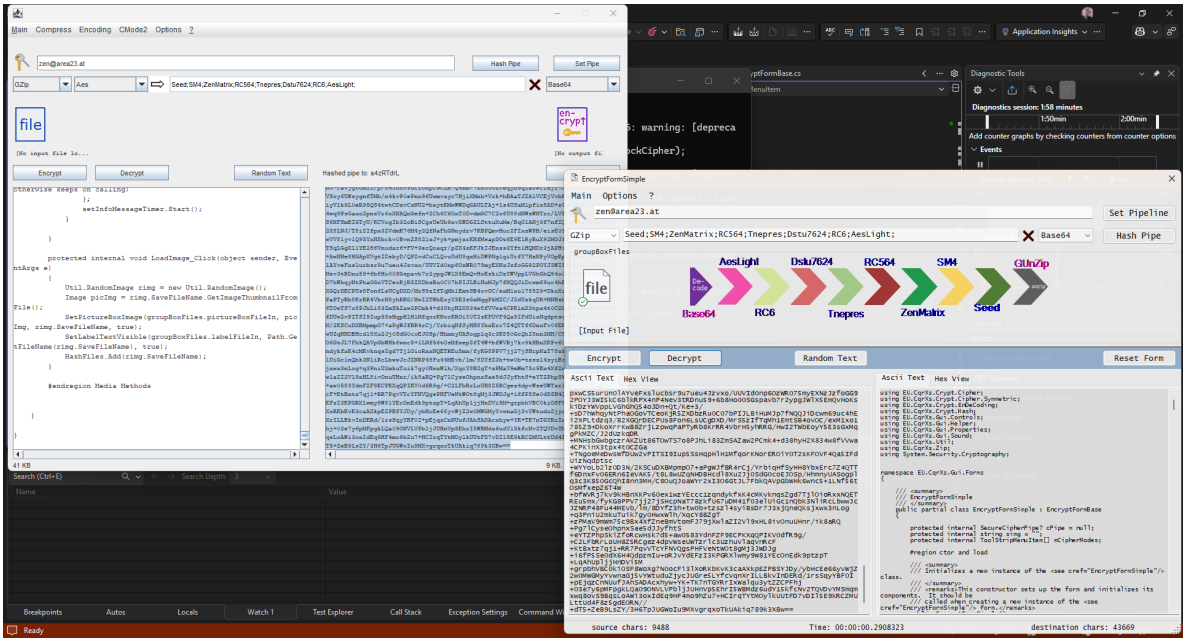

Figure 8: En-/Decryption in simple mode between C# and Java

3.5 En-/Decryption between C# Java in Simple mode

We can see, that simple mode is intercompatible between java and C#. Here you see a text encrypted in simple mode in java and decrypted in C#.

...

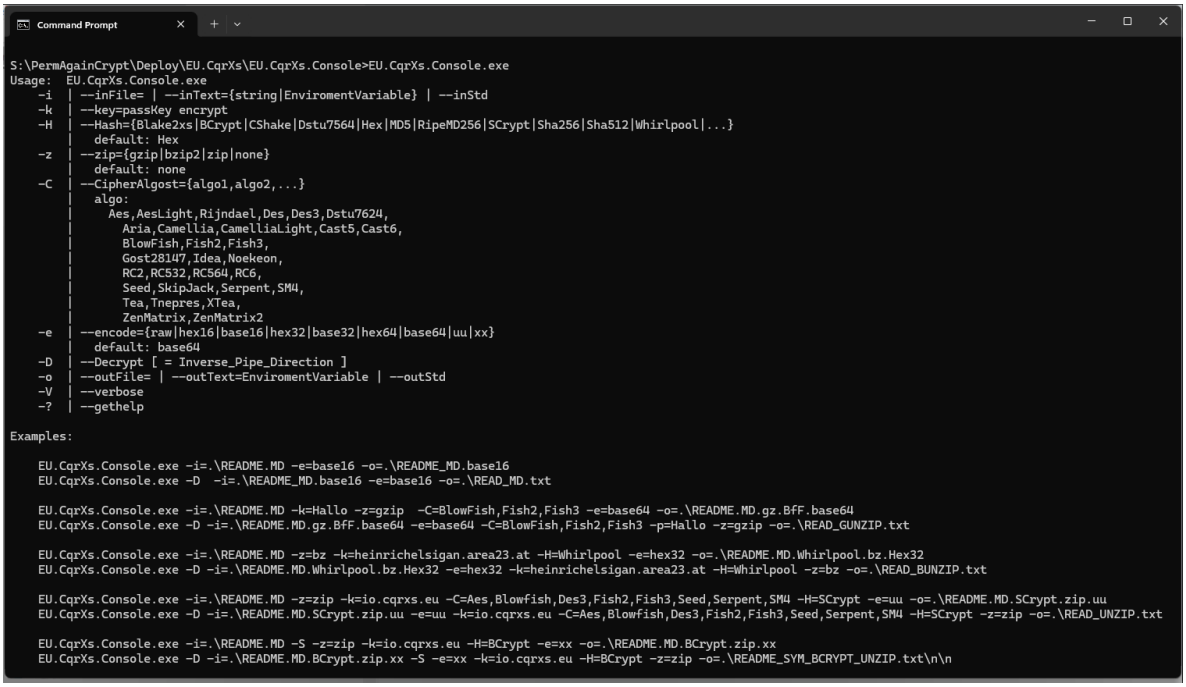

Figure 9: Windows EU.CqrXs.Console.exe

4 Console and cmd programs

4.1 Windows Console

Download the lastest Windows Console from io.cqrxs.eu/download/EU.CqrXs.Console/

4.1.1 Usage: EU.CqrXs.Console.exe

```
Usage:  EU.CqrXs.Console.exe
    -i  | --inFile= | --inText={string|EnviromentVariable} | --inStd
    -k  | --key=passKey encrypt
    -H  | --Hash={Blake2xs|BCrypt|CShake|Dstu7564|Hex|MD5|RipeMD256|SCrypt|Sha256|Sha512|Whirlpool|...}
        |   default: Hex
    -z  | --zip={gzip|bzip2|zip|none}
        |   default: none
    -C  | --CipherAlgost={algo1,algo2,...}
        |   algo:
        |     Aes,AesLight,Rijndael,Des,Des3,Dstu7624,
        |       Aria,Camellia,CamelliaLight,Cast5,Cast6,
        |       BlowFish,Fish2,Fish3,
        |       Gost28147,Idea,Noekeon,
        |       RC2,RC532,RC564,RC6,
        |       Seed,SkipJack,Serpent,SM4,
        |       Tea,Tnepres,XTea,
        |       ZenMatrix,ZenMatrix2
    -e  | --encode={raw|hex16|base16|hex32|base32|hex64|base64|uu|xx}
        |   default: base64
    -D  | --Decrypt [ = Inverse_Pipe_Direction ]
    -o  | --outFile= | --outText=EnviromentVariable | --outStd
    -S  | -simpleMode
    -V  | --verbose
    -?  | --gethelp
```

4.1.2 Examples: EU.CqrXs.Console.exe

```
EU.CqrXs.Console.exe -i=.\README.MD -e=base16 -o=.\READMD.base16
EU.CqrXs.Console.exe -D  -i=.\READMD.base16 -e=base16 -o=.\READ_MD.txt

EU.CqrXs.Console.exe -i=.\README.MD -k=Hallo -z=gzip  -C=BlowFish,Fish2,Fish3
    -e=base64 -o=.\README.MD.gz.BfF.base64
EU.CqrXs.Console.exe -D -i=.\README.MD.gz.BfF.base64 -e=base64 -C=BlowFish,Fish2,Fish3
    -p=Hallo -z=gzip -o=.\READ_GUNZIP.txt

EU.CqrXs.Console.exe -i=.\README.MD -z=bz -k=heinrichelsigan.area23.at
    -H=Whirlpool -e=hex32 -o=.\README.MD.Whirlpool.bz.Hex32
EU.CqrXs.Console.exe -D -i=.\README.MD.Whirlpool.bz.Hex32 -e=hex32
    -k=heinrichelsigan.area23.at -H=Whirlpool -z=bz -o=.\READ_BUNZIP.txt

EU.CqrXs.Console.exe -i=.\README.MD -z=zip -k=io.cqrxs.eu
    -C=Aes,Blowfish,Des3,Fish2,Fish3,Seed,Serpent,SM4 -H=SCrypt
    -e=uu -o=.\README.MD.SCrypt.zip.uu
EU.CqrXs.Console.exe -D -i=.\README.MD.SCrypt.zip.uu -e=uu
    -k=io.cqrxs.eu -C=Aes,Blowfish,Des3,Fish2,Fish3,Seed,Serpent,SM4
    -H=SCrypt -z=zip -o=.\READ_UNZIP.txt

EU.CqrXs.Console.exe -i=.\README.MD -S -z=zip -k=io.cqrxs.eu -H=BCrypt
    -e=xx -o=.\README.MD.BCrypt.zip.xx
EU.CqrXs.Console.exe -D -i=.\README.MD.BCrypt.zip.xx -S -e=xx -k=io.cqrxs.eu
    -H=BCrypt -z=zip -o=.\README_SYM_BCRYPT_UNZIP.txt
```

5 Theory

5.1 8-staged symmetric block cipher pipeline

An eight staged symmetric block cipher crypto pipeline to improve advanced encryption standard based on meta DES, 3DES with P-Box S-Box.

The following image shows you an example of a symmetric cipher 8 staged encryption pipe and the corresponding decryption inverse pipe.

Before entering the encryption pipe, the file can be zipped to avoid huge amount of symmetric cipher blocks and after exiting the encryption pipe the file can be ascii encoded with base64 mime, uuencode, xxencode or hex16, because symmetric chiphered binary files might lose their block padding.

Implementation is based on my blog article: Making symmetric cipher encryption meta permutating again[Els24], including the follwing symmetric cipher algorithms:

- Aes, AesLight, Rijndael
- Bruce Schneier's BlowFish, 2-Fish, 3-Fish
- Camellia, CamelliaLight
- Cast5, Cast6
- National security agency's Des, 3-Des, SkipJack
- Dstu7624
- Ghost, Idea, Noekeon
- RC2, RC532, RC564, RC6
- SEED, SM4
- Serpent, Tnepres
- Tea, XTea
- and my own simplest symmetric block cipher alogrithms: ZenMatrix, ZenMatrix2

5.2 What are advantages and disadvantages of Symmetric Block Cipher

5.2.1 Advantages

Since symmetric block cipher ciphers each block in the same encrypting way

parallel processing can be implemented quiet easy with average performance bust on huge multi-processor machines.

5.2.2 Disadvantages

Since we often know concrete structure of a file header, because of more static tableized structure inside file header and the specific binary format signature (MIT magic cookie)

REPLAY algorithms could be used by trying encode only the 1st symmetric cipher block with some heuristic headers and possible keys.

5.2.3 Most blockcipher algorithms break on a lot of 0 byte inside

Most blockcipher algorithms break, when you fill a $>$ 2x BLOCKSIZE (byte)0 inside a text or some other file. You can generate such a NULL block with linux dd:

```
sudo nice -n -17 dd if=/dev/zero of=/mnt/h/zeros.txt bs=4k  count=64
```

That is why we have added a gzip, bzip2, zip before to compress illegal character blocks inside the file to encrypt.

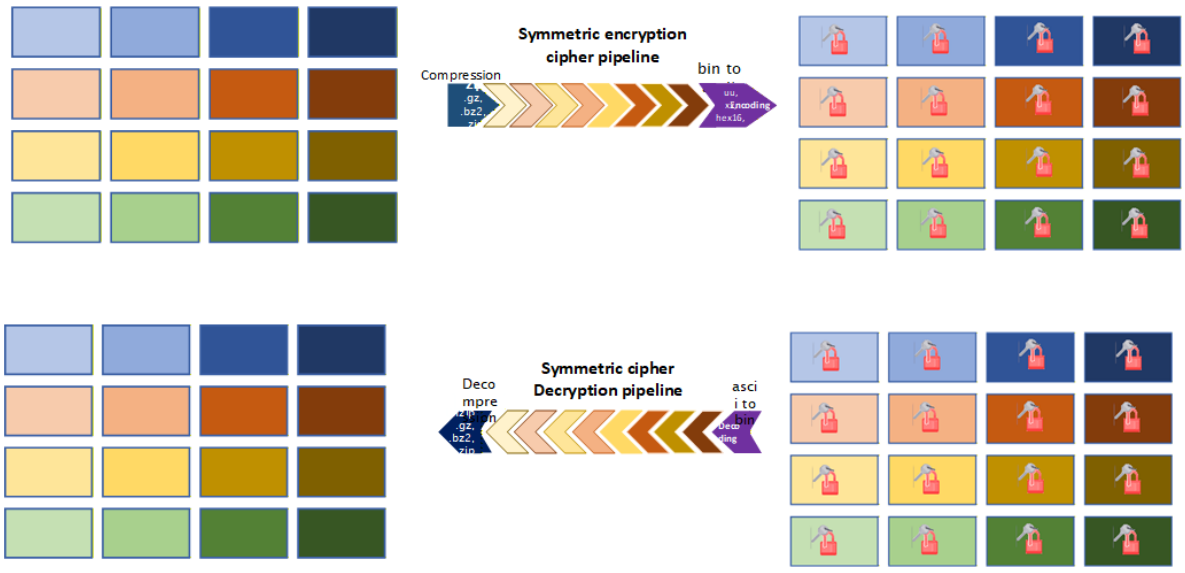

Figure 10: Symmetric BlockCipher Structure

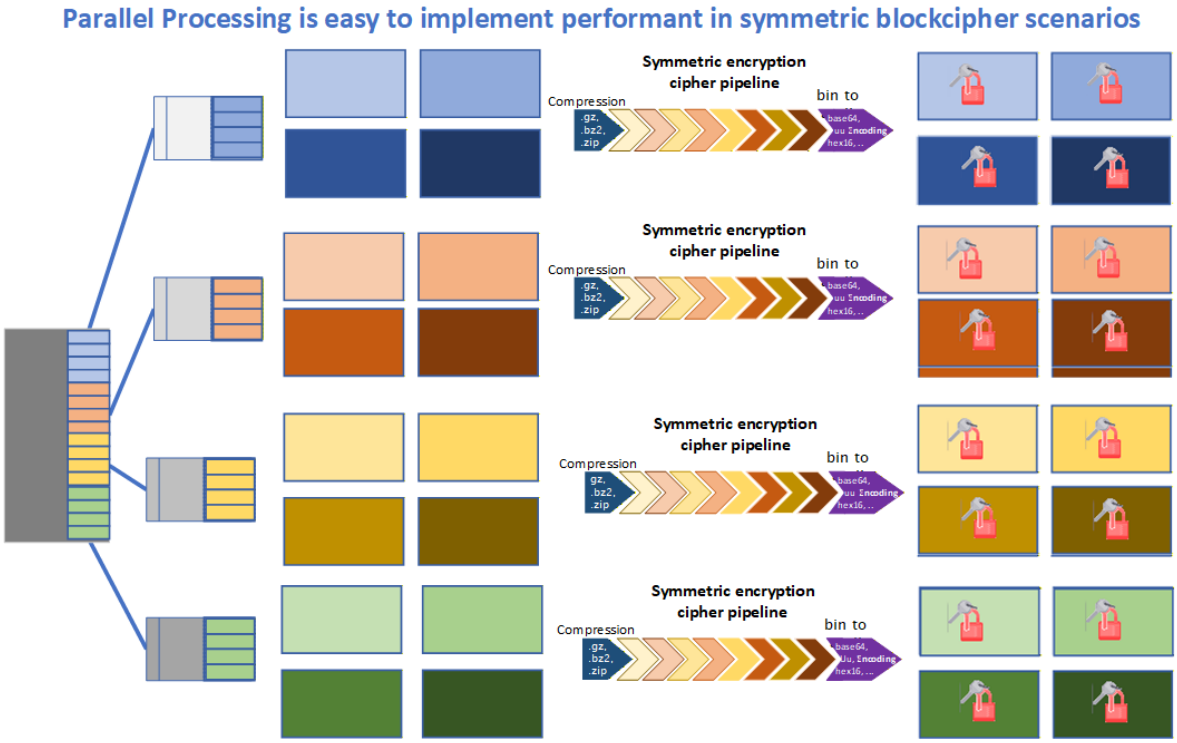

Figure 11: Symmetric BlockCipher Parallelism

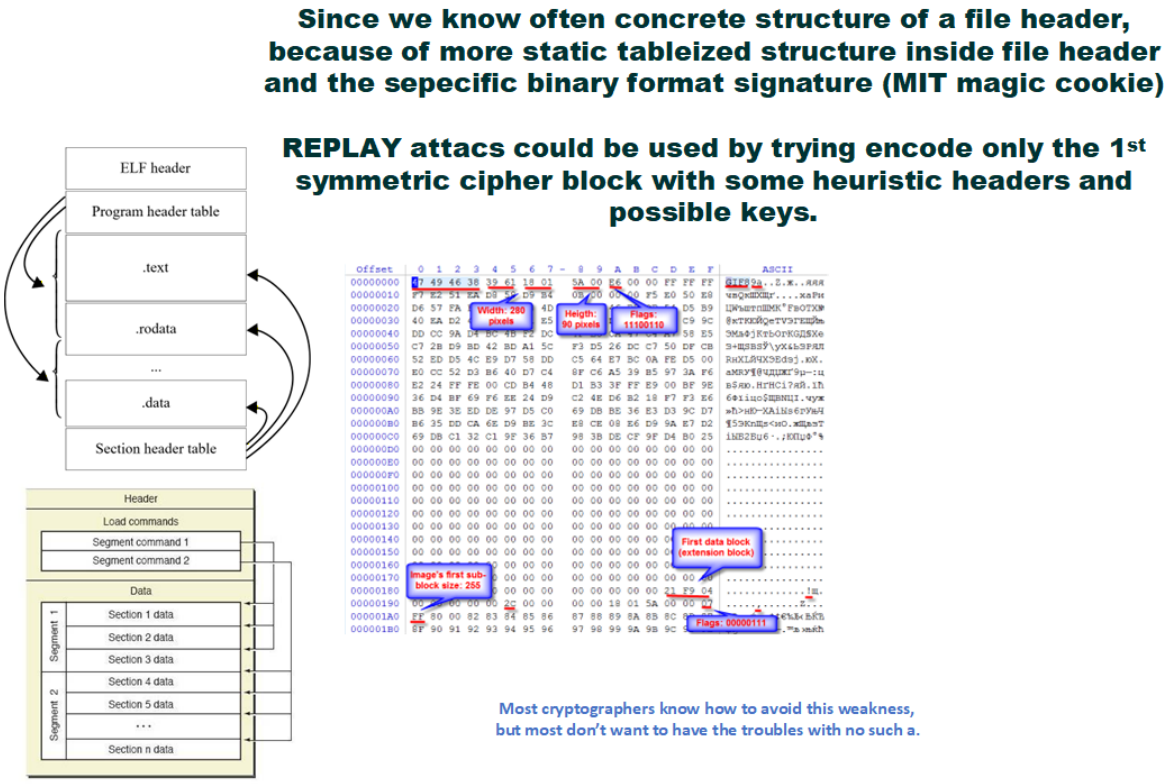

Figure 12: Symmetric BlockCipher Header 1st Block

5.3 Mathematical theory

Consider that there are full bijective desterministic inversible functions,
where the inverse function of $y_x = F(x, ...)$ is $x_y = f(y, ..)$
then inverse function
to $y_x = F(G(H(I(J(K(L(M(N(x, ...)))))))))$
is $x_y = n(m(l(k(j(i(h(g(f(y, ...)))))))))$.

You can see a mappings from ascii-8 $\longmapsto$ ascii-8,
also always as Matrix from R256 $\longmapsto$ R256
or hexadecimal from Rx100 $\longmapsto$ Rx100.
you can see mappings from UTF-8 $\longrightarrow$ UTF-8

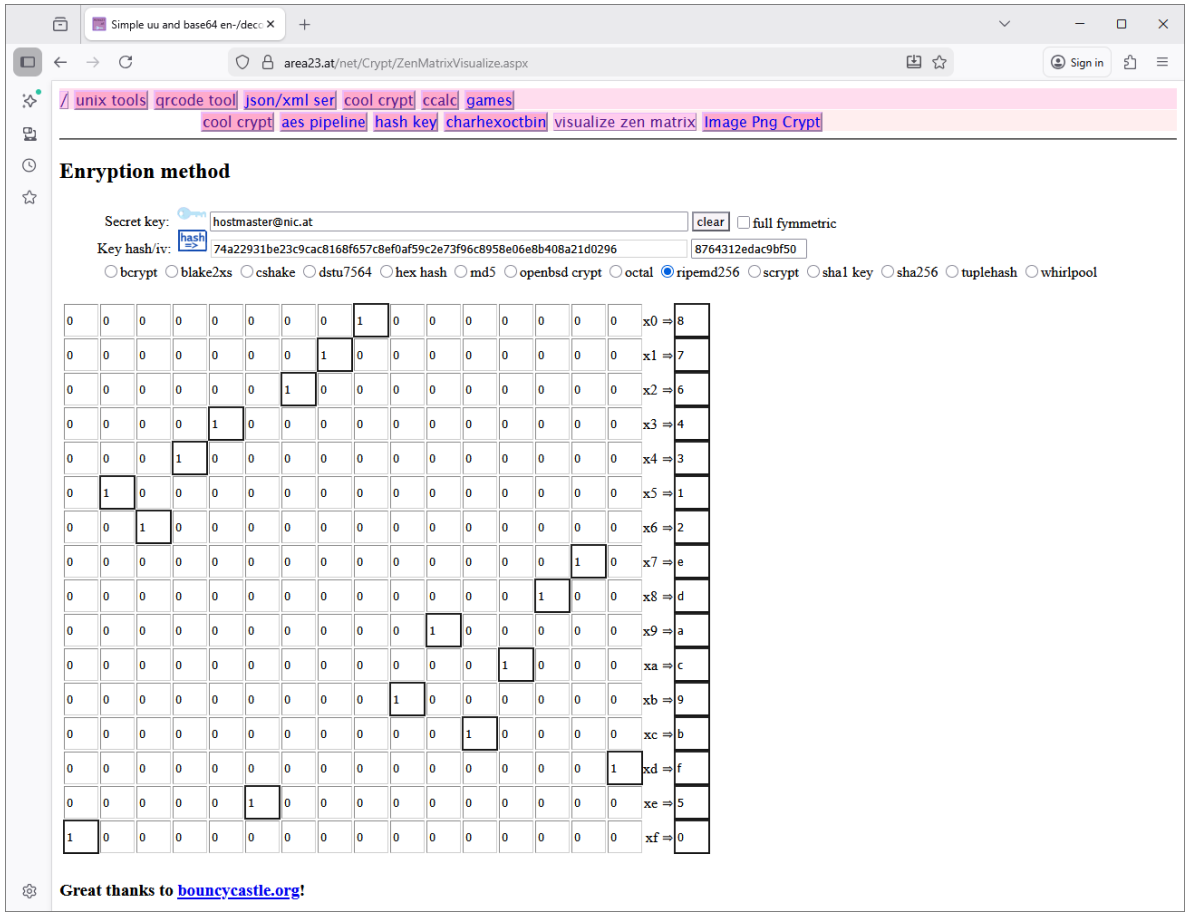

Figure 13: ZenMatrix Visualize

5.4 ZenMatrix a simplest possible algortihm to basically understand symmetric blockcipher

Starting from a no permutating 1-matrix, where projection is same as base, ZenMatrix generates a permutating mapping with blocksize 16 a matrix with only one 1 per row and column (rest is 0) to change position inside block and value offset.

https://area23.at/net/Crypt/ZenMatrixVisualize.aspx

x	x0	x1	x2	x3	x4	x5	x6	x7	x8	x9	xA	xB	xC	xD	xE	xF
x0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
x1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
x2	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
x3	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
x4	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0
x5	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0
x6	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0
x7	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0
x8	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0
x9	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0
xA	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0
xB	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0
xC	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0
xD	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0
xE	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0
xF	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

When entering hostmaster@nic.at with hash ripemd256 and not fully symmetric checked, the matrix will look like this:

x	x0	x1	x2	x3	x4	x5	x6	x7	x8	x9	xA	xB	xC	xD	xE	xF	Mx
x0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	8
x1	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	7
x2	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	6
x3	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	4
x4	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	3
x5	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
x6	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	2
x7	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	E
x8	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	D
x9	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	A
xA	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	C
xB	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	9
xC	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	B
xD	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	F
xE	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	5
xF	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

6 Code Examples

6.1 C# Code Example

```
string s = "This is another boring example text to test CipherPipe en-/decryption.";
string hash = KeyHash.Hex.Hash(key), key = "myKey", hashIv = KeyHash.Hex.Hash(key);

// 1. C\# constructing a CipherPipe with parameter key and hash
CipherPipe cPipe = new CipherPipe(key, hash,
    EncodingType.Base64, ZipType.None, KeyHash.Hex, CipherMode2.CFB);

// 2. constructing with an array of ciphers, e.g. BlowFish;TwoFish;ThreeFish
string algos = "BlowFish;TwoFish;ThreeFish";
CipherEnum[] pipeAlgos = CipherEnumExtensions.ParsePipeText(algos);
cPipe = new CipherPipe(pipeAlgos, 8,
    EncodingType.Base64, ZipType.None, KeyHash.Hex, CipherMode2.CFB);

// 3. calling CipherPipe encrypt text
string encryptext = cPipe.EncrpytTextGoRounds(s, key, hashIv,
    EncodingType.Base64, ZipType.None, KeyHash.Hex, CipherMode2.CFB);

// 4. calling CipherPipe encrypt bytes
byte[] cbytes = cPipe.EncryptEncodeBytes(System.Text.Encoding.Utf8.GetBytes(s),
    key, hashIv, EncodingType.Base64, ZipType.None, KeyHash.Hex, CipherMode2.CFB);

// 5. calling CipherPipe decrypt text
string dcrypt = cPipe.DecryptTextRoundsGo(encryptext, key, hashIv,
    EncodingType.Base64, ZipType.None, KeyHash.Hex, CipherMode2.CFB);

// 6. calling CipherPipe decypt bytes
byte[] outBytes = cPipe.DecodeDecrpytBytes(cbytes, key, hashIv,
    EncodingType.Base64, ZipType.None, KeyHash.Hex, CipherMode2.CFB);
```

6.2 Java Example

```
boolean reverseDirection = false;       // variables we need later for cipher pipe
ZipType zipType = ZipType.None;
EncodeEnum encodingType = EncodeEnum.None;
KeyHash keyHash = KeyHash.Hex;
String inString = "Hallo, byte[] inside String will converted soonly!", outString = "";
String passKey = "MySecureKey", optCryptAlgos = "Aes,Des,Des3,Blowfish,Fish2,Fish3";
String[] algos = optCryptAlgos.split(",;:");
byte[] outBytes = null, inBytes = inString.getBytes(StandardCharsets.UTF_8);

CipherPipe pipe;                        // Create cipher pipe 4 en-/decrypting
if (passKey == null || passKey.isEmpty() || algos.length > 0) {
    pipe =  new CipherPipe(algos, Constants.MAX_PIPE_LEN,
            encodingType, zipType, keyHash, CipherMode2.CFB);  // CFB is default CipherMode
    verbout("Created pipe without passkey: " + pipe.getPipeString());
} else {
    pipe = new CipherPipe(passKey, keyHash.hash(passKey),
            encodingType, zipType, keyHash, CipherMode2.CFB); // CFB instead old ECB
    verbout("Created pipe with passkey=" + passKey + " pipe=" + pipe.getPipeString());
}
if (!reverseDirection) {                // encrypt
    PrintPipe(pipe, reverseDirection);
    try {                               // CipherPipe encrypt encode
        passKey = (passKey.length() == 0) ? " " : passKey;
        outBytes = pipe.encryptEncodeBytes(inBytes,
                    passKey, keyHash.hash(passKey),
                    encodingType, zipType, keyHash, CipherMode2.CFB);
    } catch (Exception exi) {
        exi.printStackTrace();
    }
    outString = new String(outBytes);
} else {                                // decrypt
    String inString = new String(inBytes);
    PrintPipe(pipe, reverseDirection);
    try {                               // CipherPipe decode decrypt
        passKey = (passKey == null || passKey.isEmpty()) ? "" : passKey;
        outBytes = pipe.decodeDecrpytBytes(inBytes,
                    passKey, (passKey.isEmpty() ? "" : keyHash.hash(passKey)),
                    encodingType, zipType, keyHash, CipherMode2.CFB);
    } catch (Exception exi) {
        exi.printStackTrace();
    }
}
```

6.3 Good luck!

We hope you find PermAgainCrypt useful, and good luck. To contact me, use the contacts at https://heinrichelsigan.area23.at.

Abstract

Hi, I'm Heinrich Elsigan I am interested in C#, Java, MSSQL, .Net Core, Android and politics, society and the future; currently working as freelancer (one person company) and planning a secure endpoint 2 endpoint chat and looking to collaborate on reviews for other repositories and projects. Article written in LaTeX.

1. personal tech and political blog blog.area23.at
2. GitHub repositories github.com/heinrichelsigan/
3. StackOverflow stackoverflow.com/users/12213151/heinrich-elsigan
4. Curriculum vitae heinrichelsigan.area23.at/cv
5. live demo .Net area23.at/net

References

[Els24] Heinrich Elsigan. Making symmetric cipher encryption meta permutating again. *Google Blogger*, 2024.