© Czeslaw Koscielny 2014

Wroclaw School of Information Technology, Poland
e-mail: ckoscielny@horyzont.eu

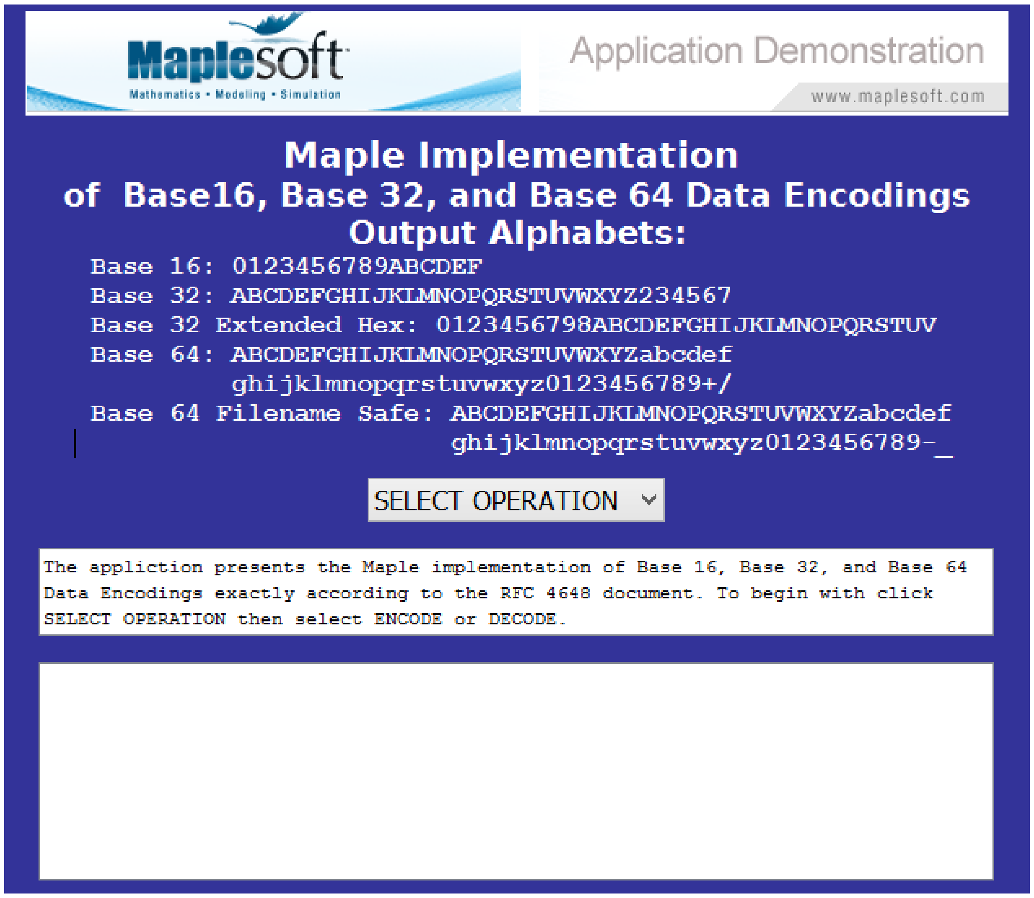
A New Approach to Transport Encryption

Introduction

Living in the global surveillance era, any internet user should himself organize the secrecy of his communication. Therefore, in the submission, it is shown how to use the base conversion as an effective cryptographic transformation because the statistical structure of the encoded file is quite different from that of the input file. As it is known, any encoding scheme usually encodes binary data into base 16, base 32 and base 64 representation to ensure that the data will not be modified during transmission [1]. The presented approach demonstrates how to easily transform the transport encoding application into the tool for a very strong transport encryption. The presented applications **trafilencod.mw** and **trafilencrp.mw** must have permission to save and remove the processed files.

Maple Implementation of Base16, Base 32, and Base 64 Data Encodings

The application **trafilencod.mw** having the **GUI** shown in Fig. 1., allows to convert an arbitrary file into **Base 16**, **Base 32**, **Base 32 Extended Hex**, **Base 64**, and **Base 64 Filename Safe** format according to the document RFC 4648 [1]. The code of the application is mainly placed in the startup code region. The rest of the implementation in two combo boxes is contained. The application is easy to use, and its whole source code is readily available. The implementation is a little sophisticated: it uses ten procedures: **cudir**, **sf2ed**, **fr**, **af2b16f**, **b16f2af**, **af2b32f**, **b32f2af**, **af2b64f**, **b64f2af**, **seli2bi** and one variable **mes** of type string. The first procedure computes the current directory, the second one allows to select a file to be processed, the third one is used to remove unnecessary file. The next six procedures perform encoding and decoding while the last procedure determines the data depending on the base value. The variable **mes** is the message appearing in the text area on opening the application. The name of the encoded file is a concatenation of the name of the input file with the added extension **b16**, **b32**, **fex**, **b64**, or **bfe**, depending on the selected base value.


Maplesoft
Mathematics • Modeling • Simulation
Application Demonstration
www.maplesoft.com
SELECT OPERATION

Maple Implementation of Base16, Base 32, and Base 64 Data Encodings
Output Alphabets:

```
Base 16: 0123456789ABCDEF
Base 32: ABCDEFGHIJKLMNOPQRSTUVWXYZ234567
Base 32 Extended Hex: 0123456798ABCDEFGHIJKLMNOPQRSTUV
Base 64: ABCDEFGHIJKLMNOPQRSTUVWXYZabcdef
         ghijklmnopqrstuvwxyz0123456789+/
Base 64 Filename Safe: ABCDEFGHIJKLMNOPQRSTUVWXYZabcdef
                       ghijklmnopqrstuvwxyz0123456789-_
```

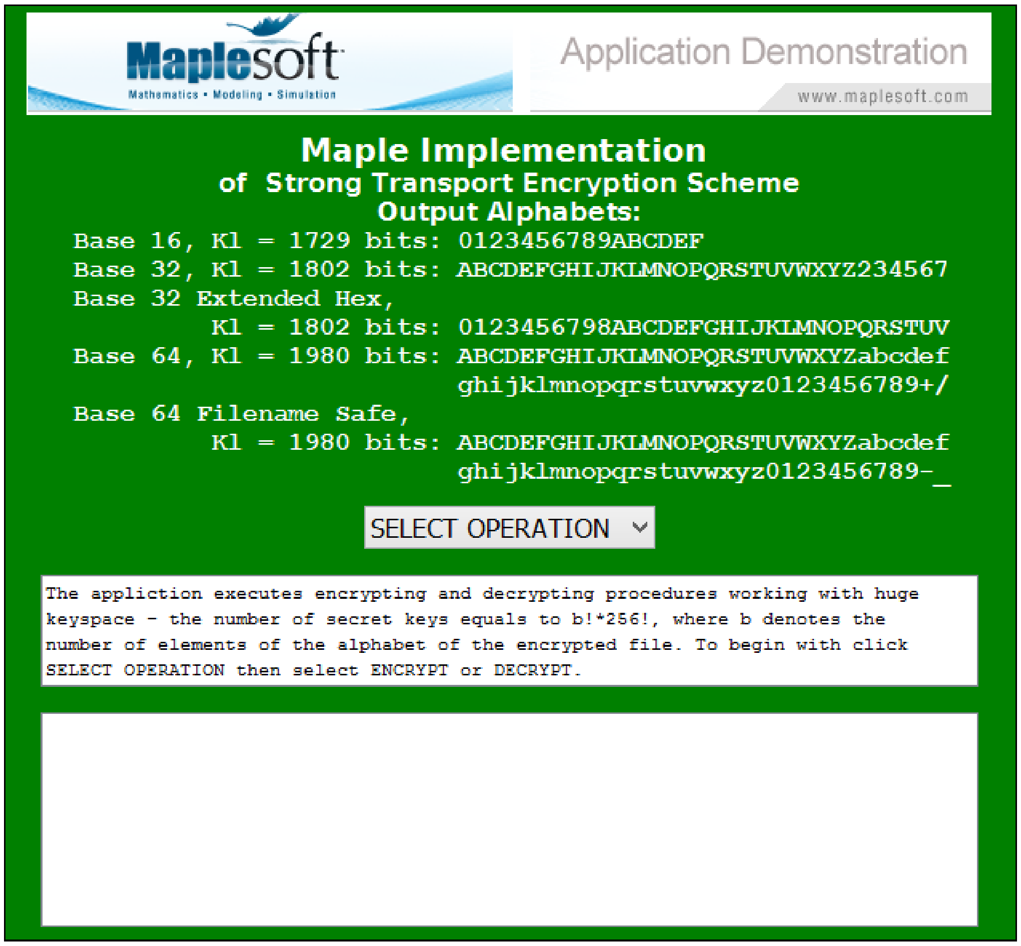
```
The appliction presents the Maple implementation of Base 16, Base 32, and Base 64
Data Encodings exactly according to the RFC 4648 document. To begin with click
SELECT OPERATION then select ENCODE or DECODE.
```

Fig. 1. Graphical user interface of the application **trafilencod.mw**

The graphical user interface has been constructed using the **DocumentTols** package. It has a form of the table with four rows, two text areas and two combo boxes visible at the moment when they should be used. To execute the operation of encoding/decoding again the user should click the icon on the worksheet toolbar.

Maple Implementation of Transport Encryption

The application **trafilencrp.mw** having the **GUI** shown in Fig. 2., performs the task of strong transport encryption, using insignificantly modified worksheet **trafilencod.mw**. The first three procedures in the startup code region are the same as in **trafilencrp.mw**. The procedures **af2b16f**, **b16f2af**, **af2b32f**, **b32f2af**, **af2b64f**, **b64f2af**, are here subtly changed and have one instruction more than the same procedures in the application **trafilencod.mw** (the reader can easily notice the difference). They have the same names but here they perform the task of cryptographic procedures with cryptographic transformations in the form of two substitutions and one base conversion. The procedure **seli2bi** is exchanged for the procedure **keygen**, and the variable **pass** serves as a secret encryption/decryption key. This way the secret key is embedded in the application. The name of the encrypted file is a concatenation of the name of the input file with the added extension **e16**, **e32**, **eex**, **e64**, or **efe**, depending on the base value selected. It should be noted that it is possible to encrypt the same input file several times.


Maplesoft
Mathematics • Modeling • Simulation
Application Demonstration
www.maplesoft.com
SELECT OPERATION

Maple Implementation
of Strong Transport Encryption Scheme
Output Alphabets:

```
Base 16, Kl = 1729 bits: 0123456789ABCDEF
Base 32, Kl = 1802 bits: ABCDEFGHIJKLMNOPQRSTUVWXYZ234567
Base 32 Extended Hex,
         Kl = 1802 bits: 0123456798ABCDEFGHIJKLMNOPQRSTUV
Base 64, Kl = 1980 bits: ABCDEFGHIJKLMNOPQRSTUVWXYZabcdef
                         ghijklmnopqrstuvwxyz0123456789+/
Base 64 Filename Safe,
         Kl = 1980 bits: ABCDEFGHIJKLMNOPQRSTUVWXYZabcdef
                         ghijklmnopqrstuvwxyz0123456789-_
```

```
The appliction executes encrypting and decrypting procedures working with huge
keyspace - the number of secret keys equals to b!*256!, where b denotes the
number of elements of the alphabet of the encrypted file. To begin with click
SELECT OPERATION then select ENCRYPT or DECRYPT.
```

Fig. 2. Graphical user interface of the application **trafilencrp.mw**

The graphical user interface is here almost the same as in the worksheet

trafilencod.mw. It has been constructed using the **DocumentTols** package and has a form of the table with four rows, two text areas and two combo boxes visible at the moment when they should be used. To execute the operation of encrypting/decrypting again the user should click the icon on the worksheet toolbar. It is evident that it is possible to encrypt the same input file several times. It is also recommended to test this application using various passwords, e.g.:

```
> pass := "Do geese see God? ";#(a palindrome)
```

To do it, it suffices to remove the first character # in the last statement in the startup code region before using the application.

Example

To demonstrate how the worksheets **trafilencrp.mw** and **trafilencod.mw** work, the file **wavfile.wav** has been chosen as an input file. Its byte-frequency is shown in Fig. 3.

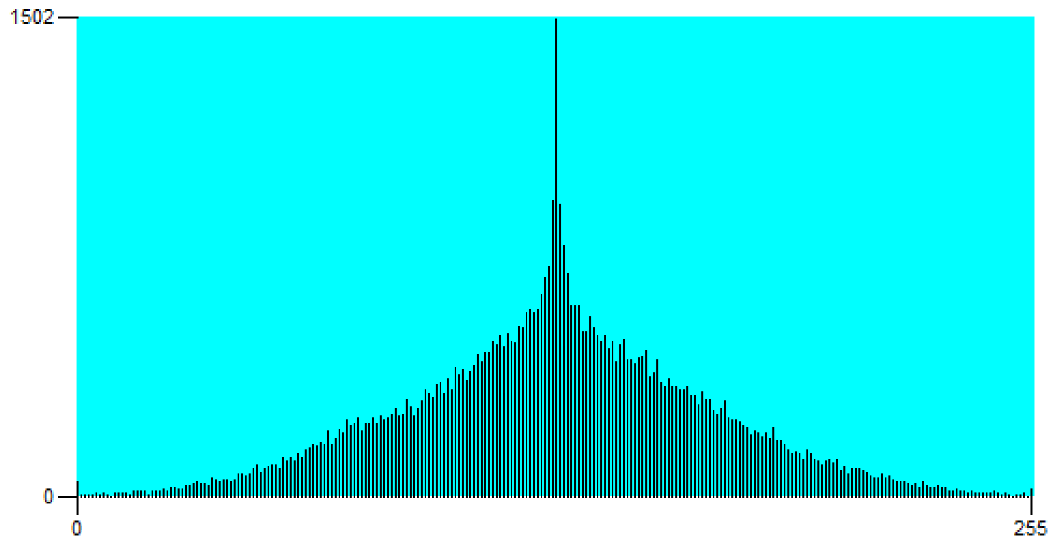
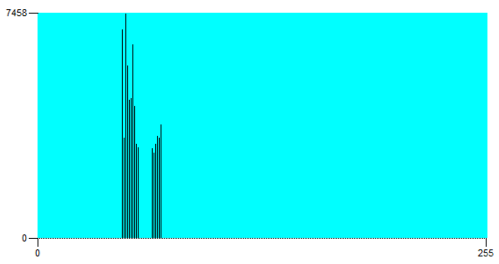


Fig. 3. Byte frequency in the file **wavfile.wav** of size 55490 B

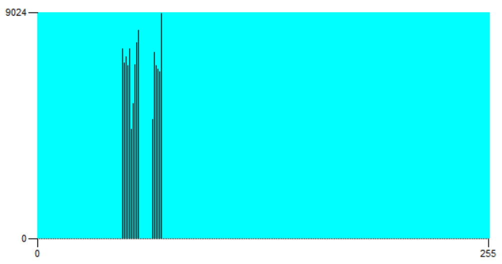
In the figures beneath the reader can see the byte frequencies of the encoded and encrypted input file and the messages on the execution of the application. In any message are contained the exact output file name and the encoding/decoding or encrypting/decrypting rates. It is worth to know that the author used the computer with Procesor Intel(R) Core(TM) i7-4771 CPU @ 3.50GHz, 3500 MHz, 4 cores,working under Window 8.1.



Byte frequency in the encoded file
G:/wavfile.wav.p16
of size ifs = 110980

The file **G:/wavfile.wav** of size ifs = 55490 B
has been encoded in 5.703 s
and saved as **G:/wavfile.wav.p16**.
The size of the encoded file: ofs = 110980 B.
Encoding rate: 9730 B/s. ofs/ifs = 2.

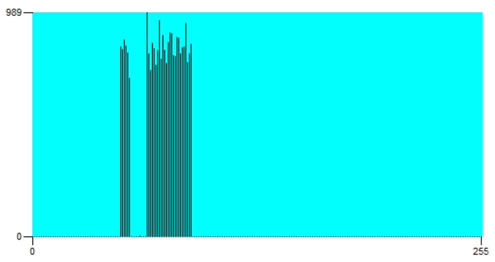
The encoded file **G:/wavfile.wav.p16**
of size ifs = 110980 B
has been decoded in 6.625 s
and saved as **G:/wavfile.wav**.
The size of the decoded file: ofs = 55490 B.
Decoding rate: 16752 B/s. ofs/ifs = .50000000.



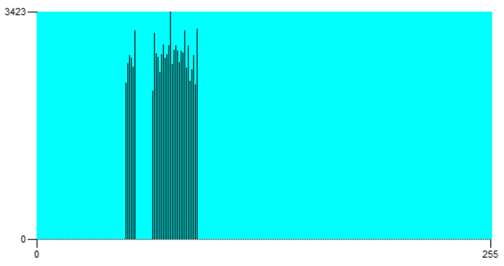
Byte frequency in the encrypted file
G:/wavfile.wav.e16
of size 110980 B

The file **G:/wavfile.wav** of size ifs = 55490 B
has been encrypted in 5.672 s
and saved as **G:/wavfile.wav.e16**.
The size of the encrypted file: fout = 110980 B.
Encrypting rate: 19566 B/s. ofs/ifs = 2.
.
The encrypted file **G:/wavfile.wav.e16**
of size ifs = 110980 B
has been decrypted in 6.641 s
and saved as **G:/wavfile.wav.**
The size of the decrypted file: ofs = 55490 B.
Decrypting rate: 16711 B/s. ofs/ifs = .50000000.

Fig. 4. The effect of encoding and encrypting of the input file with the **Base 16** option



Byte frequency in the encoded file



Byte frequency in the encrypted file
G:/wavfile.wav.e32 of size 88784 B

G:/wavfile.wav.p32 of size 88784 B

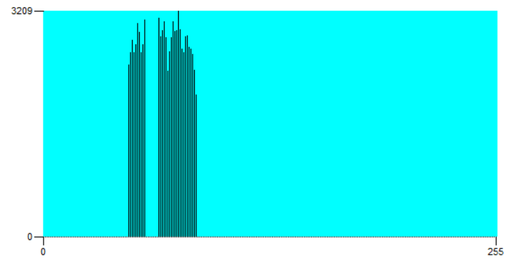
The file **G:/wavfile.wav** of size ifs = 55490 B
has been encoded in 5.172 s
and saved as **G:/wavfile.wav.p32.**
The size of the encoded file: ofs = 88784 B.
Encoding rate: 10729 B/s. ofs/ifs = 1.6000000

The encoded file **G:/wavfile.wav.p32**
of size ifs = 88784 B
has been decoded in 5.797 s
and saved as **G:/wavfile.wav.**
The size of the decoded file: ofs = 55490 B.
Decoding rate: 15316 B/s. ofs/ifs = .62500000.

The file **G:/wavfile.wav** of size 55490 B
has been encrypted in 5.109 s
and saved as **G:/wavfile.wav.e32.**
The size of the encrypted file: ofs = 88784 B.
Encrypting rate: 17378 B/s. ofs/ifs = 1.6000000

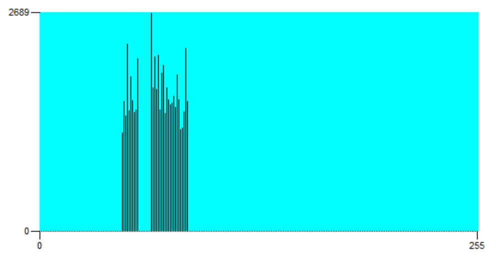
The encrypted file **G:/wavfile.wav.e32**
of size ifs = 88784 B
has been decrypted in 5.859 s
and saved as **G:/wavfile.wav**.
The size of the decrypted file: ofs = 55490 B.
Decrypting rate: 15153 B/s. ofs/ifs = .62500000.

Fig. 5. The effect of encoding and encrypting of the input file with the **Base 32** option



Byte frequency in the output encoded file
G:/wavfile.wav.pex of size 88784 B

The file **G:/wavfile.wav** of size ifs = 55490 B
has been encoded in 5.204 s
and saved as **G:/wavfile.wav.pex.**
The size of the encoded file: ofs = 88784 B.
Encoding rate: 10663 B/s. ofs/ifs = 1.6000000.



Byte frequency in the output encrypted file
G:/wavfile.wav.e32 of size 88784 B

The file **G:/wavfile.wav** of size ifs = 55490 B
has been encrypted in 5.203 s
and saved as **G:/wavfile.wav.e32**.
The size of the encrypted file: ofs = 88784 B.
Encrypting rate: 17064 B/s. ofs/ifs = 1.6000000.

The encoded file **G:/wavfile.wav.pex** of size ifs = 88784 B has been decoded in 5.859 s and saved as **G:/wavfile.wav.** The size of the decoded file: ofs = 55490 B. Decoding rate: 15153 B/s.ofs/ifs = .62500000.	The encrypted file **G:/wavfile.wav.e32** of size ifs = 88784 B has been decrypted in 5.860 s and saved as **G:/wavfile.wav**. The size of the decrypted file: ofs = 55490 B. Decrypting rate: 15151 B/s. ofs/ifs = .62500000.

Fig. 6. The effect of encoding and encrypting of the input file with the **Base 32 Extended Hex** option

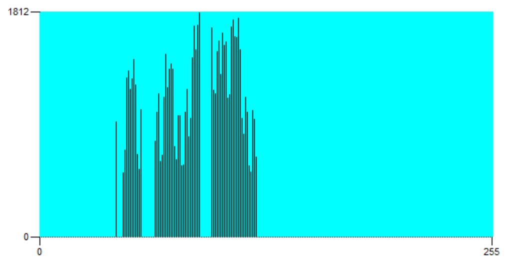
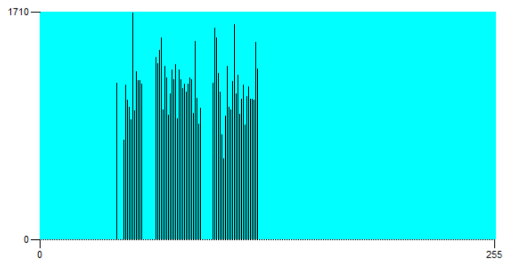
	
Byte frequency in the output encoded file **G:/wavfile.wav.p64** of size 73988 B The file **G:/wavfile.wav** of size ifs = 55490 B has been encoded in 4.781 s and saved as **G:/wavfile.wav.p64**. The size of the encoded file: ofs = 73988 B. Encoding rate: 11606 B/s. ofs/ifs = 1.3333574. The encoded file **G:/wavfile.wav.p64** of size ifs = 73988 B has been decoded in 5.281 s and saved as **G:/wavfile.wav**. The size of the decoded file: ofs = 55490 B. Decoding rate: 14010 B/s. ofs/ifs = .74998648.	Byte frequency in the output encrypted file **G:/wavfile.wav.e64** of size 73988 B The file **G:/wavfile.wav** of size ifs = 55490 B has been encrypted in 4.828 s and saved as **G:/wavfile.wav.e64**. The size of the encrypted file: ofs = 73988 B. Encrypting rate: 15325 B/s. ofs/ifs = 1.3333574. The encrypted file **G:/wavfile.wav.e64** of size ifs = 73988 B has been decrypted in 5.360 s and saved as **G:/wavfile.wav**. The size of the decrypted file: ofs = 55490 B. Decrypting rate: 13804 B/s. ofs/ifs = .74998648.

Fig. 7. The effect of encoding and encrypting of the input file with the **Base 64**

option

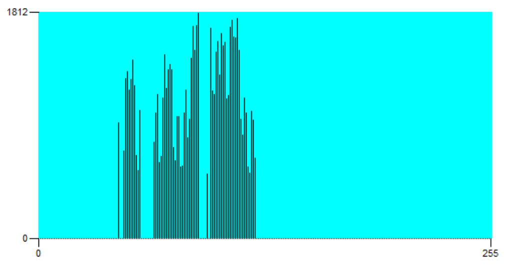
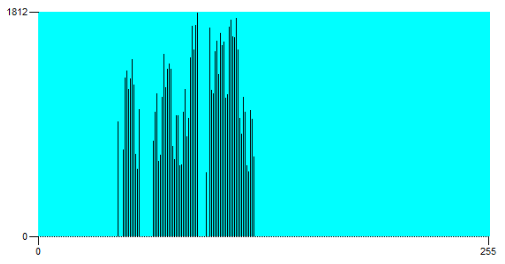
	
Byte frequency in the output encoded file **G:/wavfile.wav.p64** of size 73988 B	Byte frequency in the output encrypted file **G:/wavfile.wav.efe** of size 73988 B
The file **G:/wavfile.wav** of size ifs = 55490 B has been encoded in 4.859 s and saved as **G:/wavfile.wav.pfs**. The size of the encoded file: ofs = 73988 B. Encoding rate: 11420 B/s. ofs/ifs = 1.3333574	The file **G:/wavfile.wav** of size ifs = 55490 B has been encrypted in 4.859 s and saved as **G:/wavfile.wav.efe**. The size of the encrypted file: ofs = 73988 B. Encrypting rate: 15227 B/s. ofs/ifs = 1.3333574.
The encoded file **G:/wavfile.wav.pfs** of size ifs = 73988 B has been decoded in 5.343 s and saved as **G:/wavfile.wav**. The size of the decoded file: ofs = 55490 B. Decoding rate: 13848 B/s. ofs/ifs = .74998648.	The encrypted file **G:/wavfile.wav.efe** of size ifs = 73988 B has been decrypted in 5.266 s and saved as **G:/wavfile.wav**. The size of the decrypted file: ofs = 55490 B. Decrypting rate: 14050 B/s.ofs/ifs = .74998648.

Fig. 8. The effect of encoding and encrypting of the input file with the **Base 64 Filename Safe** option

Conclusions

A practical example of application of simple algorithms which converts the positive integers from the base 256 to bases 16, 32, and 64 as an effective cryptographic transformation has been shown. The worksheet **trafilencrp.mw** uses the secret keys of length 1729, 1802, and 1980 bits long. In the case of the key = 1729 bits the key space has 2^{1729} e.g. about $3.026\ 10^{520}$ elements. Assuming with optimism that it is possible to verify 10^{8} keys in one second, to verify all the secret keys would last $9.59\ 10^{504}$ years (the age of the Earth is

$4.54 \cdot 10^9$ years). For the time being a more effective cryptanalysis is not yet known. It is highly likely that if the input file is encrypted more than once, the generated cryptogram file is unbreakable.

Reference

[1] http://www.ietf.org/rfc/rfc4648.txt