International Journal of Advances in Computer Networks and Its Security– IJCNS
Volume 5: Issue 2 [ISSN : 2250-3757]
Publication Date : 30 October, 2015

Secure PHP OpenSSL Crypto Online Tool

Radu Braniscan, Marius Constantin Popescu, Antoanela Naaji

***Abstract*—In this paper we studied the PHP's cryptographic extension OpenSSL and we did an implementation as a web application (online tool with demo) where the user can experiment and test for free symmetric operations (encryption and decryption), hashing, asymmetric encryption (generate public and private keys, encrypt and decrypt) using the most popular ciphers, along with their options to choose from: cipher mode and key size. This tool is important because it differentiates from the current online tools: it shows that the AES cipher is correctly implemented by successfully passing several test vectors; it uses the OpenSSL library; it allows the user to choose the desired cipher, cipher mode, key, padding and initialization vector if appropriate/necessary; it has an updated documentation (demo, examples, tutorial) with full source code and has the advantages later described in this paper when compared to other similar free online tools.**

***Keywords*—PHP, OpenSSL, cryptography, online, tool, demo, AES test vectors, AES, security, web application, ciphers, encrypt, decrypt, hash, symmetric encryption, asymmetric encryption**

I. Introduction

Studying this field we decided to make such an application based on an original design layout with a user-friendly interface [1], [2], [3]. After the application was developed we looked for other similar work. What we found out was that there was not one website that offered a complete and professional service to do crypto online. Even more, there is plenty of space for improvements to be me made and this web application comes to fill this gap and be a competitive tool to choose from when experimenting with online crypto tools. In this paper we will show the complete list of features of the tool, how to use it (short tutorial), and ~~I~~ demonstrate that the AES cipher is implemented correctly because it passes several test vectors. The application uses the following technologies in the front-end: HTML5 (forms) [4], CSS3 (Bootstrap, Font Awesome) [5], JavaScript (jQuery) [6], Bootstrap [7], Countable.js [8]). In the back-end it uses PHP (v5.4.7) and the OpenSSL (v1.0.1c) cryptographic extension [9], [10], [11]. The application is hosted at HostGator where it was tested and working in modern web browsers (Google Chrome v43 – recommended, Internet Explorer v9, Mozilla Firefox v36).

II. Features

A. Symmetric cryptography

Encipher or encrypt allows the user to make such operations (openssl_encrypt function) by writing in form inputs a plain text, key, selecting a cipher, the padding (OPENSSL_RAW_DATA – used or OPENSSL_ZERO_ PADDING) [12]; the IV is generated randomly and after clicking the encipher button it gets the result (cipher text) along with the other input texts. Also, the user can see the number of characters for each of these. If there is any error it will be displayed above the input forms (openssl_error_string function [13]). It allows using any of the 52 ciphers with their combined key size and mode (Tab.I).

Radu Braniscan, Marius Constantin Popescu, Antoanela Naaji
"Vasile Goldis" Western University of Arad
Romania

TABLE I. TYPES OF FILLING

aes-128-cfb	aes-256-cfb1	des-cfb	desx-cbc
aes-128-cfb1	aes-256-cfb8	des-cfb1	dea-cbc
aes-128-cfb8	aes-256-ecb	des-cfb8	idea-cfb
aes-128-ecb	aes-256-ofb	des-ecb	idea-ecb
aes-128-ofb	bf-cbc	des-ede	idea-ofb
aes-192-cbc	bf-cfb	des-ede-cbc	rc2-40-cbc
aes-192-cfb	bf-ecb	des-ede-cfb	rc2-64-cbc
aes-192-cfb1	bf-ofb	des-ede-ofb	rc2-cbc
aes-192-cfb8	cast5-cbc	des-ede3	rc2-cfb
aes-192-ecb	cast5-cfb	des-ede3-cbc	rc2-ecb
aes-192-ofb	cast5-ecb	des-ede3-cfb	rc2-ofb
aes-256-cbc	cast5-ofb	des-ede3-ofb	rc4
aes-256-cfb	des-cbc	des-ofb	rc4-40

Breakdown: aes-128-cfb (aes –cipher name, 128 – cipher key size in bits, cfb – cipher mode).

Cipher modes: - Electronic Codebook (ECB), Cipher-Block Chaining (CBC), Cipher Feedback (CFB), Output Feedback (OFB). Almost all ciphers modes require an initialization vector (IV), except for the AES with ECB cipher mode. The application generates one for each operation and allows using any other by choice if desired. Decipher or decrypt is the reverse operation of encipher (openssl_decrypt function). Hashing can be done using any of the 43 algorithms (Tab.II).

TABLE II. ENCRYPTION ALGORITHM

md2	ripemd320	adler32	haval128,4
md4	whirlpool	crc32	haval160,4
md5	tiger128,3	crc32b	haval192,4
sha1	tiger160,3	fnv132	haval224,4
sha224	tiger192,3	fnv164	haval256,4
sha256	tiger128,4	joaat	haval128,5
sha384	tiger160,4	haval128,3	haval160,5
sha512	tiger192,4	haval160,3	haval192,5
ripemd128	snefru	haval192,3	haval224,5
ripemd160	snefru256	haval224,3	haval256,5
ripemd256	gost	haval256,3	

Hashing is a one-way operation (no decryption or dehashing), which can be done only by using a rainbow table.

International Journal of Advances in Computer Networks and Its Security– IJCNS
Volume 5: Issue 2 [ISSN : 2250-3757]
Publication Date : 30 October, 2015

B. *Asymmetric cryptography*

It can generate private and public keys based on the desired key bits size (512, 1024, 2048, 4096), key type (RSA) and digest algorithm (method): sha512, sha384, sha256, sha224, md2, md5, sha1, dss1, mdc2, ripemd160. The keys can be downloaded as txt files because they are temporarily saved on the server for 60 seconds (there is a cron job that deletes all txt files every minute).

Encrypt is done by inputting a plain text, the public key of the person to whom the message is sent and after clicking on the submit button it generates the cipher text that can be sent to the person that can decrypt (read) it only by using his private key. Decrypt is done by inputting the cipher text (encrypted with your own public key, e.g. a message received from somebody) and a private key of your own to be used in order to read the plain text.

III. Test Vectors

This tool passes the AES test vectors provided by InCON Team and available online here [14]. In order to pass the test some changes have to be made. The text of the test is in hex format. The original PHP function openssl_encrypt [15] transforms the text to hex format so, because of this, all the user's inputs (plain text, key, iv) are converted first with the php function hex2bin [16]. Exactly the reverse has to happen before: hex is transformed to text, which is then again transformed to hex. The output is then converted from bin to text format in order to obtain the same result [17]. The padding used through all the tests is OPENSSL_RAW_DATA.

A. *Test Vectors for AES/ECB Encryption Mode*

The test vectors for AES/ECB are presented in Tab.III, Tab.IV and Tab.V.

TABLE III. AES ECB 128-BIT ENCRYPTION MODE

Test vector	Cipher text
6bc1bee22e 409f96e93d 7e11739317 2a	3ad77bb40d7a3660a89ecaf32466ef97 (expected) 3ad77bb40d7a3660a89ecaf32466ef97a254be88e037 ddd9d79fb6411c3f9df8 (via SPOCOT – acceptable result)

Encryption key: 2b7e151628aed2a6abf7158809cf4f3c
Initialization vector: not necessary

TABLE IV. AES ECB 192-BIT ENCRYPTION MODE

Test vector	Cipher text
6bc1bee22e4 09f96e93d7e 117393172a	bd334f1d6e45f25ff712a214571fa5cc(expected) bd334f1d6e45f25ff712a214571fa5ccdaa0af074bd80 83c8a32d4fc563c55cc (via SPOCOT – acceptable result)

Encryption key:8e73b0f7da0e6452c810f32b809079e562f8ead2522c6b7b
Initialization vector: not necessary

TABLE V. AES ECB 256-BIT ENCRYPTION MODE

Test vector	Cipher text
6bc1bee22e 409f96e93d 7e11739317 2a	f3eed1bdb5d2a03c064b5a7e3db181f8 (expected) f3eed1bdb5d2a03c064b5a7e3db181f84c45dfb3b3b4 84ec35b0512dc8c1c4d6 (via SPOCOT – acceptable result)

Encryption key: 603deb1015ca71be2b73aef0857d77811 f352c073b6108d72d9810a30914dff4 Initialization vector: not necessary

B. *Test Vectors for AES/CBC Encryption Mode*

The test vectors for AES/ECB are presented in Tab.VI, Tab.VII and Tab.VIII [18].

TABLE VI. AES CBC 128-BIT ENCRYPTION MODE

Initialization vector	Test vector	Cipher text
000102030405 060708090A0 B0C0D0E0F	6bc1bee22e 409f96e93d 7e1173931 72a	7649abac8119b246cee98e9b12e9 197d (expected) 7649abac8119b246cee98e9b12e9 197d8964e0b149c10b7b682e6e3 9aaeb731c (via SPOCOT – acceptable result)

Encryption key: 2b7e151628aed2a6abf7158809cf4f3c

TABLE VII. AES CBC 192-BIT ENCRYPTION MODE

Initialization vector	Test vector	Cipher text
000102030405 060708090A0 B0C0D0E0F	6bc1bee22e 409f96e93d 7e1173931 72a	4f021db243bc633d7178183a9fa0 71e8 (expected) 4f021db243bc633d7178183a9fa0 71e8a647f1643b94812a175a13c8 fa2014b2 (via SPOCOT – acceptable result)

Encryption key: 8e73b0f7da0e6452c810f32b809079e562f8 ead2522c6b7b

TABLE VIII. AES CB 256-BIT ENCRYPTION MODE

Initialization vector	Test vector	Cipher text
00010203040 5060708090A 0B0C0D0E0F	6bc1bee22e 409f96e93d 7e1173931 72a	f58c4c04d6e5f1ba779eabfb5f7bf bd6 (expected) f58c4c04d6e5f1ba779eabfb5f7bf bd6485a5c81519cf378fa36d42b 8547edc0 (via SPOCOT – acceptable result)

Encryption key: 603deb1015ca71be2b73aef0857d77811f35 2c073b6108d72d9810a30914dff4

C. *Test Vectors for AES/CFB Encryption Mode*

The test vectors for AES/ECB are presented in Tab.IX, Tab.X and Tab.XI.

TABLE IX. AES CFB 128-BIT ENCRYPTION MODE

Initialization vector	Test vector	Cipher text
00010203040 5060708090a 0b0c0d0e0f	6bc1bee22e 409f96e93d 7e1173931 72a	3b3fd92eb72dad20333449f8e83 cfb4a (expected) 3b3fd92eb72dad20333449f8e83 cfb4a (via SPOCOT – exact result)

Encryption key: 2b7e151628aed2a6abf7158809cf4f3c

TABLE X. AES CFB 192-BIT ENCRYPTION MODE

Initialization vector	Test vector	Cipher text
000102030405 060708090A0 B0C0D0E0F	6bc1bee22e 409f96e93d 7e1173931 72a	cdc80d6fddf18cab34c25909c99a 4174 (expected) cdc80d6fddf18cab34c25909c99a 4174 (via SPOCOT – exact result)

Encryption key: 8e73b0f7da0e6452c810f32b809079e562f8 ead2522c6b7b

TABLE XI. AES CFB 256-BIT ENCRYPTION MODE

Initialization vector	Test vector	Cipher text
000102030405 060708090A0 B0C0D0E0F	6bc1bee22e 409f96e93d 7e1173931 72a	dc7e84bfda79164b7ecd8486985 d3860 (expected) dc7e84bfda79164b7ecd8486985 d3860 (via SPOCOT – exact result)

D. *Test Vectors for AES/OFB Encryption Mode*

The test vectors for AES/ECB are presented in Tab.XII, Tab.XII and Tab.XIV.

TABLE XII. AES OFB 128-BIT ENCRYPTION MODE

Initialization vector	Test vector	Cipher text
00010203040 5060708090A 0B0C0D0E0F	6bc1bee22e 409f96e93d 7e1173931 72a	3b3fd92eb72dad20333449f8e83 cfb4a(expected) 3b3fd92eb72dad20333449f8e83 cfb4a (via SPOCOT – exact result)

Encryption key: 2b7e151628aed2a6abf7158809cf4f3c

TABLE XIII. AES OFB 192-BIT ENCRYPTION MODE

Initialization vector	Test vector	Cipher text
00010203040 5060708090A 0B0C0D0E0F	6bc1bee22e 409f96e93d 7e1173931 72a	cdc80d6fddf18cab34c25909c99a 4174 (expected) cdc80d6fddf18cab34c25909c99a 4174 (via SPOCOT – exact result)

Encryption key: 8e73b0f7da0e6452c810f32b809079e562f8 ead2522c6b7b

TABLE XIV. AES OFB 256-BIT ENCRYPTION MODE

Initialization vector	Test vector	Cipher text
00010203040 5060708090 A0B0C0D0E 0F	6bc1bee22 e409f96e93 d7e117393 172a	dc7e84bfda79164b7ecd848698 5d3860 (expected) dc7e84bfda79164b7ecd848698 5d3860 (via SPOCOT – exact result)

Encryption key: 603deb1015ca71be2b73aef0857d77811f 352c073b6108d72d9810a30914dff4

IV. Comparison of Free Online Tools

While drawing up a review of 10 websites to be featured on the list, we established some minimum conditions (Tab.XV):

- no longer than 2 years have passed since the last update;
- it mentions at least the used cipher, if nothing else.

TABLE XV. COMPARISION BETWEEN ONLINE TOOLS

Criteria / Website	A	B	C	D	E	F	G	H	I	J
Documentation	⍓	⍓	⍓	⍓	⍓	☒	⍓	☑	⍓	⍓
A list of algorithms	☑	☑	☒	☑	☑	☑	☒	☒	☒	☒
Configure cipher options (mode, key size, iv, output and input encode type)	☑	⍓	☒	⍓	⍓	⍓	☒	⍓	⍓	☑
Symmetric cryptography	☑	☑	☑	☑	☑	☑	☒	☒	☒	☒
Asymmetric cryptography	☑	☒	☒	☒	☒	☒	☑	☑	☑	☑
Source code	☑	☒	☒	☒	☒	☒	☒	☑	☑	☑
Secure connection	☑	☒	☑	☒	☒	☒	☑	☒	☑	☒
Total:	6y 0n 1p	2y 3n 2p	2y 4n 1p	2y 3n 2p	2y 3n 2p	2y 4n 1p	2y 4n 1p	3y 3n 1p	3y 3n 2p	3y 3n 1p

Legend: ☑ - yes; ☒ - no; ⍓ - partial;

A. SPOCOT | http://proappsoft.com/spocot/
B. Online encrypt tool - Online tools - Tools 4 noobs | http://www.tools4noobs.com/online_tools/encrypt/
C. Encrypt free | https://www.encryptfree.com/
D. AES – Symmetric Ciphers Online | http://symmetric-ciphers.online-domain-tools.com/
E. Secure Cryptographic Online Tool | http://www.bierkandt.org/encryption/symmetric_encryption.php
F. Best Online Encrypt Decrypt Tool | http://codebeautify.org/encrypt-decrypt
G. PGP Encryption Tool – iGolder | https://www.igolder.com/PGP/generate-key/
H. JSEncrypt | http://travistidwell.com/jsencrypt/demo/
I. PGP Key Generator|https://wp2pgpmail.com/pgp-key-generator/
J. AS3 Crypto Demo page | http://crypto.hurlant.com/demo/

General Comparison - in our personal opinion the second best tool for *symmetric encryption* is "D" because it yields the closest result to the AES test vectors result, but with some catches: it has a daily limit based on IP, it can't select key size for AES and the encrypted text is shown in hex and binary format separated by a white space.

AES ECB 128-bit encryption mode
Encryption key: 2b7e151628aed2a6abf7158809cf4f3c
No initialization vector:
Test vector: 6bc1bee22e409f96e93d7e117393172a
Cipher text on InCON Team web site:
3ad77bb40d7a3660a89ecaf32466ef97
Cipher text on "D" web site:
3ac3977bc2b42e7a3660c2a82ec38ac3b32466c3af
For *asymmetric encryption* is "I" because of the overall total score.

V. Future Work

General (short term) - show execution time in seconds for each operation (transformation). General (medium-long term) - refactor and optimize the source code. Develop an online anonymous secure live chat service. Develop an online professional secure social network service. Develop the same application but using the Mcrypt cryptographic extension and make a comparative analysis. Add a feedback form to improve the service [19]. Limit the use of service or implement other techniques in order to avoid traffic overload and allow a large base of user access.

A. *Symmetric Cryptography*

Add options to select the code type of text for plain, key and cipher text such as: text, binary, hex, base64. Also make it so there is a default auto detection option which can be turn off if needed. Turn cipher mode and key size into options to choose for the cipher algorithm. Then they are included next to the algorithm name in the dropdown list. Add the feature so that once you write the plain text, key and initialization vector (if necessary), it generates cipher texts using all possible ciphers. Also allow adjustments in case of any error. Hash - add the option to use salts. Add completely new feature for dehashing by using rainbow tables.

B. *Asymmetric Cryptography*

Develop so it uses pass phrases (passwords), signatures and certificates. Find an alternative way to temporarily store the keys, maybe in the browser, instead of on the server as done currently, in order to avoid overload.

Demo, examples and Tutorial - online working link (HTTP) on nginx 1.8.0 [20]. Recommended - secure connection using a shared server certificate from HostGator (HTTPS) on an Apache server [21]. Menu (image – Fig.1) - Help contains about, updates log and link to this documentation. Source code links to the GitHub repository.

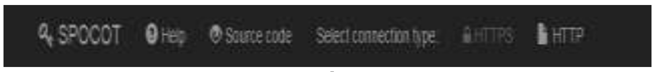
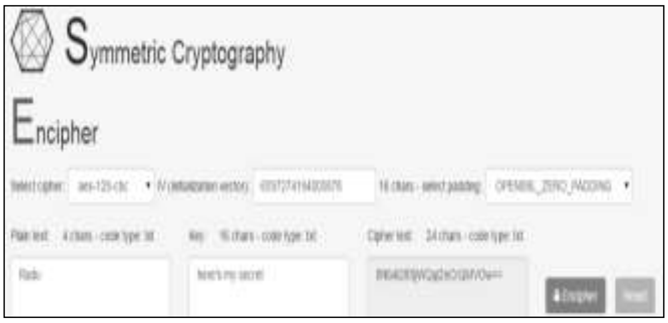


Figure 1. Applications menu.

C. *Symmetric cryptography - Encipher*

Change the cipher or padding [22]. The IV is generated automatically if empty and if the cipher requires it. Insert the text you want to encrypt into the plain text, the key size of the exactly required by the cipher, if not an error will show. After all of this you can click the blue button Encipher and the cipher text will be generated (Fig.2).



Select cipher: aes-128-cbc IV (initialization vector): 6597274194005676 16 chars - select padding: OPENSSL_ZERO_PADDING

Plain text: 4 chars - code type: txt Key: 16 chars - code type: txt Cipher text: 24 chars - code type: txt

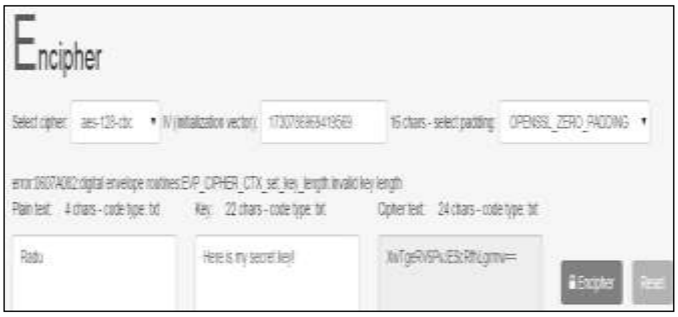
Radu

here's my secret

BN042B|WQd2nOGM/Ow==

Figure 2. Catch the fields conpletate and result for symmetric cryptography encryption operation.

Error example: The key size is here 22 chars, too long for the selected cipher that has a maximum acceptable key size of 16 chars (Fig.3).



Select cipher: aes-128-cbc IV (initialization vector): 1730789694195689 16 chars - select padding: OPENSSL_ZERO_PADDING

error:0607A082:digital envelope routines:EVP_CIPHER_CTX_set_key_length:invalid key length

Plain text: 4 chars - code type: txt Key: 22 chars - code type: txt Cipher text: 24 chars - code type: txt

Radu

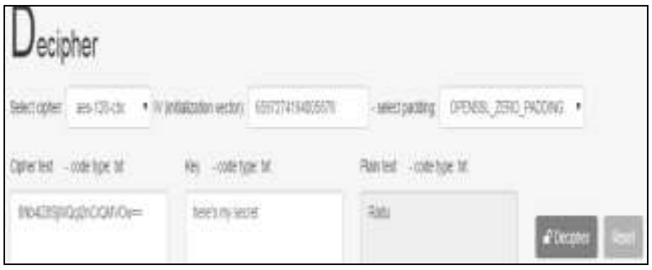
Here is my secret key!

XaTgeRVIPkJESlRfkUgmw==

Figure 3. Catch an error message conpletate fields and result for symmetric cryptography encryption operation.

D. *Symmetric cryptography - Decipher*

A catch with fields completed and the result for decryption operation of symmetric cryptography is shown in Figure 4.



Select cipher: aes-128-cbc IV (initialization vector): 6597274194005676 - select padding: OPENSSL_ZERO_PADDING

Cipher text: - code type: txt Key: - code type: txt Plain text: - code type: txt

BN042B|WQd2nOGM/Ow==

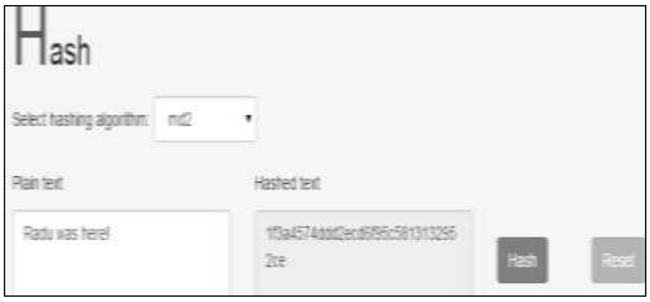
here's my secret

Radu

Figure 4. Symmetric cryptography-decipher.

E. *Hash*

A catch with fields completed and the result for symmetric cryptography operation of hash is illustrated in Figure 5.



Select hashing algorithm: md2

Plain text Hashed text

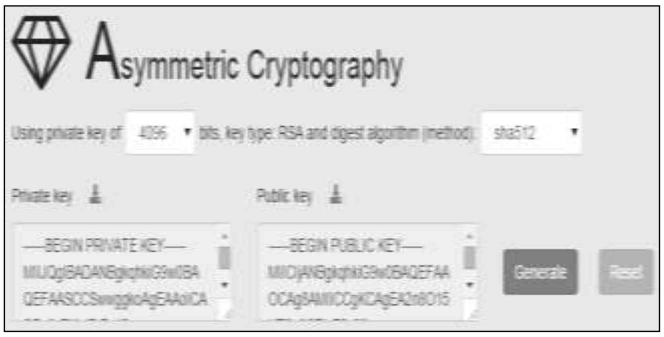
Radu was here!

1f3a4574ddd2ecd6f95c581313295 2ce

Figure 5. Hash result for the operation of symmetric cryptography.

F. *Hash Asymmetric Cryptography – First Generate Private and Public Key*

You can select the key size and the digest algorithm. After the "Generate" button is clicked the keys are automatically generated and you can download (save) the keys on your computer by clicking on the download icons (Fig.6). The keys are deleted from the server every minute so you don't have to worry about using them safely.


Asymmetric Cryptography

Using private key of 4096 bits, key type: RSA and digest algorithm (method): sha512

Private key

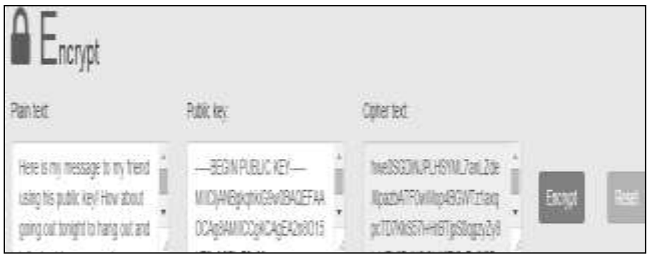
```
—–BEGIN PRIVATE KEY—–
MIIJQgIBADANBgkqhkiG9w0BA
QEFAASCCSwggkoAgEAAoICA
```

Public key

```
—–BEGIN PUBLIC KEY—–
MIICIjANBgkqhkiG9w0BAQEFAA
OCAg8AMIICCgKCAgEA2n8015
```

Figure 6. Catch the key pair generated from the generation of asymmetric cryptography.

G. *Encrypt using Public Key*


Encrypt

Plain text: Here is my message to my friend using his public key! How about going out tonight to hang out and

Public key:

```
—–BEGIN PUBLIC KEY—–
MIICIjANBgkqhkiG9w0BAQEFAA
OCAg8AMIICCgKCAgEA2n8015
```

Cipher text:

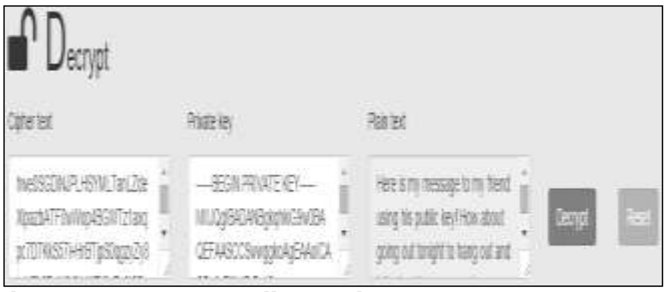
```
tw8SGDlNJPLHSYML7anLZde
XpazbATFlwWlbp4BGWTzIaxq
pcTDTKkS6T7H+HBTjpS8qgzyZy8
```

The plain text is limited to 245 characters (Fig.7).

Figure 7. Catch filled with fields (text in clear, public key) and the result (ciphertext) for asymmetric cryptography encryption operation.

H. *Decrypt using Private Key*

A catch with fields filled (text encrypted public key) and the result (plain text) to the asymmetric cryptography decryption operation is illustrated in Figure 8.


Decrypt

Cipher text

```
tw8SGDlNJPLHSYML7anLZde
XpazbATFlwWlbp4BGWTzIaxq
pcTDTKkS6T7H+HBTjpS8qgzyZy8
```

Private key

```
—–BEGIN PRIVATE KEY—–
MIIJQgIBADANBgkqhkiG9w0BA
QEFAASCCSwggkoAgEAAoICA
```

Plain text: Here is my message to my friend using his public key! How about going out tonight to hang out and

Figure 8. Decrypt using private key.

VI. Conclusions

This paper sheds light over a very specific field of online cryptographic using PHP’s OpenSSL that lacks a very good documentation and examples. As a result, the web application developed here is undoubtedly a tool that is very beneficial for all the users seeking to experiment with online crypto tools. If you are reading this paper 6 months after it was initially published, we suggest you visit the online documentation of the mentioned used technology because during this time some things may have changed and this paper, if not updated, will by then be obsolete. We had to write this because things are constantly changing in this field. As such, when using an outdated paper as a basis for developing an application, there is a high possibility that it may contain security vulnerabilities or issues.

References

[1] N. Ferguson, B. Schneier, T. Kohno, “Cryptography Engineering: Design Principles and Practical Applications”, Eds. Wiley, Indianapolis, 2010.

[2] C. Kaufman, R. Perlman, M. Speciner, “Network Security: Private Communication in a Public World”, Eds. Prentice Hall, New Jerssey, 2003.

[3] B. Groza, “Introducere în Criptografie Funcții Criptografice, Fundamente Matematice şi Computaționale”, Ed. Politehnica, Timişoara, 2012.

[4] HTML5 - Web developer guide | MDN: https://developer.mozilla.org/en-US/docs/Web/Guide/HTML/HTML5

[5] CSS3-CSS|MDN: https://developer.mozilla.org/en/docs/Web/CSS/CSS3

[6] jQuery: https://jquery.com/

[7] Bootstrap - The world's most popular mobile-first and responsive front-end framework: http://getbootstrap.com/

[8] Countable.js — Live word-counting in JavaScript: http://sacha.me/Countable/

[9] http://php.net/manual/en/book.openssl.php

[10] PHP: OpenSSL: http://php.net/manual/en/book.openssl.php

[11] OpenSSL: The Open Source toolkit for SSL/TLS: https://www.openssl.org/

[12] http://thefsb.tumblr.com/post/110749271235/using-openssl-en-decrypt-in-php-instead-of

[13] PHP: openssl_error_string - manually http://php.net/manual/en/function.openssl-error-string.php

[14] http://www.inconteam.com/software-development/41-encryption/55-aes-test-vectors#aes-ecb-128

[15] PHP: openssl_encrypt - manually http://php.net/manual/en/function.openssl-encrypt.php

[16] PHP: hex2bin - manually: http://php.net/hex2bin

[17] PHP: bin2hex – manually: http://php.net/bin2hex

[18] AES Test vectors: http://www.inconteam.com/software-development/41-encryption/55-aes-test-vectors

[19] Using openssl_en/decrypt() in PHP instead of Mcrypt: http://thefsb.tumblr.com/post/110749271235/using-openssl-en-decrypt-in-php-instead-of

[20] http://proappsoft.com/spocot/

[21] https://gator1575.hostgator.com/~appsoft/spocot

[22] The Rijndael Block Cipher. http://csrc.nist.gov/archive/aes/rijndael/Rijndael-ammended.pdf