Introduction to Computer Security Module – G6077

-

OpenSSL in a web application (PHP)

**Objectives:**

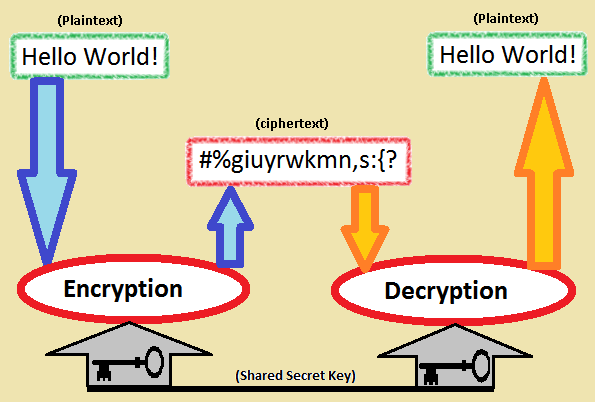
# Setting up Sussex ITS personal web space:

You might have set up personal web space for other modules. If you have, then just create another folder in public\_html, where you will save the work of this particular lab. If you have not, then follow the link to set up your personal web space: <http://www.sussex.ac.uk/its/help/faq?faqid=145>.

# Introuduction

In this topic, I will be talking about encryption and decryption using OpenSSL in PHP. I will be briefly discussing what “two way encryption” is and how to use it in your PHP application using OpenSSL encrypt and decrypt methods with a readily usable example. At the end of this page there will be a zip file where you can download and run the example.

Shared key encryption (symmetric)Two way encryption (encryption and decryption) is the simplest way of securely sending some sensitve information over the Internet using a shared secret key (symmetric). To make things understandble picture this scenario. A web user fills a form with some data and press the submit button. When the form is submitted, data will go from place A to place B. If we send data as it is, anyone (i.e. Spy) has access to data packets can open and see what’s inside. So, before sending data as it is we will use a convertion method (encryption) to transform human readable data (i.e. Plain text) into a human unreadable format (i.e. Cipher text). Then the receiving end will perform a data convertion exercise (decryption) to transform data back to readable format.



In PHP there is *[openssl\_encrypt](http://php.net/manual/en/function.openssl-encrypt.php" \t "_blank)* function available to encrypt a plain text using a hash key. Let’s take an example,

|  |
| --- |
| openssl\_encrypt($textToEncrypt, $encryptionMethod, $secretKey, $options, $iv); |

openssl\_encrypt takes five parameters which are,

1. *$textToEncrypt* – The plain text that needs to be encrypted
2. *$encryptionMethod* – Encryption method (Using openssl\_get\_cipher\_methods, [available methods](http://php.net/openssl_get_cipher_methods))
3. *$secretKey* – A key (needs to kept private)
4. *$options* – **OPENSSL\_RAW\_DATA**or **OPENSSL\_ZERO\_PADDING** (Default is 0)
5. *$iv* – An initialisation vector. (precisely 16 bytes)

The above will give you the encrypted text of the given plain text. When it comes to decryption you would only need to know the *$secretKey*and *$iv* which can be derived from the *$secretKey*(or can be unique on its own). The initialisation vector (*$iv*) is a random number which makes sure that the encrypted text is unique. It is important to learn about this *$iv*. (Refer to [this link](https://stackoverflow.com/a/12486940/767625) to learn more about $iv).

I have used the following to generate *$iv*,

|  |
| --- |
| $bytes = "";$last = ""; |
| while(strlen($bytes) &lt; 48) { |

|  |
| --- |
| $last = md5($last . $secretHash, true); |
| $bytes.= $last; |

|  |
| --- |
| } |
| $iv = substr($bytes, 32, 16); |

To decrypt above created cipher text we can use [openssl\_decrypt](http://php.net/manual/en/function.openssl-decrypt.php" \t "_blank)*function*. Let’s see an example,

|  |
| --- |
| openssl\_decrypt($cipherText, $encryptionMethod, $secretKey, 0, $iv); |

openssl\_decrypt takes five parameters which are,

1. *$cipherText* – The encrypted text that needs to be decrypted
2. *$encryptionMethod* – Encryption method (Using openssl\_get\_cipher\_methods, [available methods](http://php.net/openssl_get_cipher_methods))
3. *$secretKey* – A key (needs to kept private)
4. *$options* – **OPENSSL\_RAW\_DATA**or **OPENSSL\_ZERO\_PADDING** (Default is 0)
5. *$iv* – An initialisation vector.  (precisely 16 bytes)

Executing above will give you back the plain text data.

Link contain two sample programs one to encrypt and the other to decrypt. Visit “users.sussex.ac.uk/~YOURUSERNAME/OpenSSL/encrypt.html” to bring the encryption form. Type anything you like that needs to be encrypted and press the submit button.

In the follow up screen, copy the cipher text from the textarea. Next, visit “users.sussex.ac.uk/~YOURUSERNAME/OpenSSL/decrypt.html” and paste the copied cipher text into the textarea and press the submit button. In the next screen you will see the plain text that you have typed earlier.

In real life applications, you will not be copying encrypted or hash values from one web form to the other. They will be process according to the logic that is implemented in application. For example, if a hash value of password text is generated, it will be stored in a shadow file (password file). When the user log back in, the hash value will be generated and will be compare against the already stored value before the user is allowed to login.

Reference:

<http://anjanasilva.com/blog/encrypt-and-decrypt-using-openssl-in-php/>