CA: PROJECT 2 David Kelly | C00193216 |March 10th 2018

# Logging

All actions including form submissions, database queries and state changes (e.g. logging out) are recorded in a log file [test.csv]. The following function takes: an event, the current action; a source; the view/file from which the action originated; a result; the outcome of the action (SUCCESS, DENY or EXCEPTION); and an optional payload, such as form data or user reference which triggered the action. The current server timestamp is prepended to the output. Using the \_crypt function (detailed below), each parameter of the output string is encrypted before writing to the log file.

Note: Building/rebuilding the database [index.php] clears previous log files, creates a new log file and appends the CSV headers. Initially, directory permissions may need to be set to ‘read & write’.

function logger($event, $user, $source, $result, $payload = "" ) {

$file = fopen('test.csv', 'a+') or die("Can't open file.");

$now = getTime();

$txt = \_crypt($now).",[".\_crypt($event)."],".\_crypt($user).",". \_crypt($source).",".\_crypt($result).",".\_crypt($payload)."\n";

fwrite($file, $txt);

fclose($file);

}

# Encryption/Decryption

All data located in database tables and log files are encrypted or decrypted using the following function. The function takes a string, the data to be encrypted or decrypted; and an action, ‘e’ or ‘d’. The function performs the specified action and returns a string or bool (false).

function \_crypt( $string, $action = 'e') {

First, I defined two secret pseudo-random string of bytes, a secret key and secret iv. These were generated using bin2hex(random\_bytes(32)). The secret key will be used in the encryption/decryption function. The second secret is used to create an initialization vector that helps to help strengthen the encryption.

$iv = substr( hash( 'sha256', $secret\_iv ), 0, 16 );

Using the openssl\_encrypt() function we encrypt/decrypt the data using AES 256 encryption in CBC mode using the secret key and initialization vector.

$output = base64\_encode( openssl\_encrypt( $string, $encrypt\_method, $key, 0, $iv ) );

$output = openssl\_decrypt( base64\_decode( $string ), $encrypt\_method, $key, 0, $iv );

The output is base64 encoded to be saved as a varchar or text file without the possibility of corruption.

# Token

Token for password reset is generated using random\_bytes().

string random\_bytes ( int $length )

Random\_bytes is a PHP 7 function that generates cryptographically secure pseudo-random bytes [1]. The function returns a string of 32 cryptographically secure bytes; 32 being the specified $length value. The resulting string is passed into bin2Hex() to convert the binary data into hexadecimal representation.

The token is encrypted then stored in the database with an associated timestamp. The token expires after 5 minutes. The token string is immediately displayed to the user to be submitted in a password reset request.

# Reference

1: PHP (2016). *Documentation*. Available: <http://php.net/manual/en/function.random-bytes.php>. Accessed: [10th March 2018].