UDC 004.056

Samoilenko D.M.
Admiral Makarov National University of Shipbuilding

Sivko O.E.
Admiral Makarov National University of Shipbuilding

HTTP DATA CIPHERING ALGORITHM

Purpose. Modern tends of IT systems evolution are to fill all spheres of human life. Most of all, for connection and data transfer the HTTP protocol is used. Due to its oldness many modern problems find no simple solutions. It is possible to encounter such problems as insecurity of data transferring via POST method or troubles with sharing private data by e-mail. But it is possible to secure private data by using algorithm which is described. Methods. We used Shannon's principle by adding crypto "salt" to the source text and extending it to given length to prevent spoofing and substitution, DES-like algorithm (permutation – password applying – permutation) and feedback by ciphered text to give extra strength to the algorithm. Results. As the result an algorithm of HTTP data securing was described. PHP code fragments of functions which implement it were listed. Effectiveness of the method was proved by an algorithm tests. Discussion. Algorithm proposed in the work allows to securely transfer data via HTTP protocol. It suits for small pieces of data like passwords or personal data and should not be used to encrypt large sequences of text or other type of data.

Key words: *HTTP protocol, secure information, data protection, program code, cryptography.*

Introduction. Modern tends of IT systems evolution are to fill all spheres of human life. In some cases it dealt with interconnection of formally unrelated systems working with different encodings, protocols, algorithms, languages etc. Most of all, for connection and data transfer the HTTP protocol is used. Due to its oldness many modern problems find no simple solutions.

For example, the task is to send personal email proposition which should include some private data in a link inside the message. But email protocols allow no invisible POST data transfer. Only way is to include private data in GET parameters visible for all behind one's back. Another problem arises when user sends their private data via some e-form. Although POST method is available, data transferred could be spied with help of traffic sniffers.

Idea of the HTTP secure transferring algorithm. Such tasks as sending personal data by e-mails or acquiring passwords by Internet communication should be realized with data encryption. But there is another problem: symbols on client and server may have different encoding whereas encryption applies for symbol codes not for the symbols directly. Moreover, for a small data units like password or birthdate there are no necessity for complex ciphering. Follow Shannon's principles of secure communication we could easy use crypto "salt" with entropy not less than whole data. When entropy added equal or greater than data entropy we formally take a case similar to one-time pad ciphering that cannot be cracked.

This article is for develop a ciphering algorithm which operates with symbols directly (not with its codes) designed for small data units transfer via HTTP protocol especially for GET method.

In work [1] an idea for such ciphering was proposed. The base of the idea is involute transform in symbol set (alphabet). Mean *c* to be a ciphered symbol, *t* and *p* – text and password symbols respectively, the transform looks like

$$c = \overline{t + p}\,, \tag{1}$$

where overline means additive inverse in alphabet. In aforementioned work involute property of transform was proved, so deciphering use the same procedure (add password symbol and inverse):

$$t = \overline{c + p}\,. \tag{2}$$

Program code realization. Practical realization of data ciphering will be shown using PHP language as a popular tool for HTTP communications. For transform realization, it is necessary to declare an alphabet as a set of symbols allowed in channel

$alph = "abcdefgh…ABCDEFGH…0123…".

Usage of predefined symbols instead of built-in symbols tables, first, prevents discordance in different tables (on client and server side) and, second, allows additional control for unauthorized intervention in data.

Alphabet length will play role of calculation modulo

```
$mod = strlen($alph).
```

For calculation acceleration, we will use predefined array of symbols numbers

```
$sno = array (“a” => 0, “b” => 1,…).
```

Symbols numbers could be calculated at runtime calling corresponding function, but described way has advantage for expended time.

If $c variable used for cipher symbol, $t – for text (initial data) symbol, $p –for password and $d – for decrypted symbol, (1) and (2) transforms could be programmed as.

```
$c = $alph[$mod-($sno[$t] + $sno[$p])%$ mod-1];
$d = $alph[$mod-($sno[$c] + $sno[$p])%$mod-1].
```

For better invulnerability to impersonate, substitution and spoofing attack [3–4] the entropy of joint set MES (message – encryption – source) should be minimal. Therefore, ciphered blocks should be aligned – they must have the same size despite to different size of initial blocks. However, the data about initial block length should be preserved in cipher block.

Combine statements about additional entropy, crypto “salt” and block alignment we can describe the first stage of data ciphering:

Length of initial data block calculates and includes in it.

Random symbols adds to initial block to extend it for predetermined size.

We propose to preserve initial data block length by including at the begin of data the symbol which order in alphabet is equal to the length. Since symbol set includes at least small, capital letters and digits its length quite enough for blocks with 70 symbols. As a rule, protected HTTP data blocks has significantly shorter length:

```
$text = “secure_text”; //HTTP data example;
$len = strlen ($text); //length of data;
$text = $alph [$len].$text; //length as a symbol in first position.
```

Next, we propose a schema, similar (in idea) to DES algorithm: A) initial permutation; B) password appliance; C) final permutation. Permutation tables from DES standard could be used for 64-symbol blocks. For smaller blocks and for simpler demonstration we will use 32-symbol version built on DES ideas [4].

```
$IP = array (30, 26, 22, …11, 7, 3);
$IP1 = array (24, 8, 32, …1, 25, 9);
$perm = “”;
For ($i = 0; $i < 32; $i ++);
$perm. = $text[$IP[$i]-1]; //-1 since origin 0.
```

Password appliance is produced in cipher-feedback mode. Since data block was expanded with “salt” there is not necessary to provide other non-linear transformations. This statement will be proved by examples analysis further. Cipher-feedback means that after password symbols finishing, previous symbols of cipher are used as password:

```
$pass = “password”;
$plen = strlen ($pass);
$ciph = “”;
for ($i = 0;$i < 32; $i ++){
$t = $perm [$i];
$p = ($i < $plen)?$pass [$i]: $ciph [$i-$plen]
//password or cipher
$ciph. = $alph [$mod-($sno [$t] + $sno [$p])% $mod-1];
}
```

Finally, the inverse IP^{-1} permutation applied to cipher text. Codes are the same to IP applica-

Table 1

Algorithm tests

<table>
<tr><th>text: _secure_text</th><th>password: p</th></tr>
<tr><td colspan="2">Y H b 9 b F S 0 8 m c 9 e G 5 R O p T g f x 5 R 1 z V 9 _ r 3 _</td></tr>
<tr><td colspan="2">6 O p r 5 y E Q e t q r 8 z R x W j g r m H H E v b b y J R T m</td></tr>
<tr><td colspan="2">j t q T Q T D o t 0 r T T 4 Q 7 _ d b b W D X N 9 n 9 s 8 G 1 A</td></tr>
<tr><td colspan="2">S D v g h J y 1 2 i w g k K L I I e 0 a z Q Z G h 7 4 k 2 R a w</td></tr>
<tr><td colspan="2">m b f Y N _ O j w S g Y Q a 1 2 c 6 e 2 I R O 4 c q f 0 Z x F 8</td></tr>
</table>

tion shown above, only name of array changes for $perm1.

Algorithm results test. For ciphering algorithm testing, first, produce many ciphers for the same initial text and password. Second, repeat the test for the data consists of the same symbols.

Tabulations in Tab.1 allows better visual analysis of cipher outcome: there are no vertical lines with the same symbol (analogue for fixed bits). Therefore, in practice algorithm shows appropriate results.

Second test (with similar symbol text) shows appropriate results too. There are no enlarged probability for basic symbol or it prevailing appearance in cipher-text.

Due to involute transform and mutually inverse permutations, deciphering algorithm is the same to ciphering, but cipher text should be used instead of initial data. In deciphered result first symbol order (in alphabet) tells us how much symbols are informative.

Conclusions. Adaptation of bitwise methods, crypto- and stegoalgorithms, and information theory theorems for symbol operations allows creating an algorithm for small-block data ciphering typical in network protocol like HTTP. Algorithm use no symbol code tables and shows no fixed symbols in subsequent operations. It could be useful in protected data transfer via open network.

References:

1. Самойленко Д.М. Комплексна система захисту інформаційного ресурсу. Інформаційна безпека. 2013. С. 147–151.
2. Самойленко Д.М. Web-орієнтована система шифрування URI параметрів. Проблеми кібербезпеки інформаційних та телекомунікаційних систем. 2017. С. 196–198.
3. Pei D.Y. Authentication Schemes. Singapore: Institute for Mathematical Sciences. 2001. 36 p. URL: www.ims.nus.edu.sg/Programs/coding/files/dypei.ps (дата звернення: 28.03.2018)
4. Simmons G.J. Authentication Theory / Coding Theory. Advances in Cryptology. 1985. P. 411–431.
5. DES supplementary material // Wikipedia, the free encyclopedia. URL: https://en.wikipedia.org/wiki/DES_supplementary_material (дата звернення: 28.03.2018)
6. FIPS Publication 46–3, Data Encryption Standard (DES). URL: https://csrc.nist.gov/csrc/media/publications/fips/46/3/archive/1999-10-25/documents/fips46-3.pdf (дата звернення: 28.03.2018)

АЛГОРИТМ ШИФРУВАННЯ ДАНИХ HTTP-ПРОТОКОЛУ

Мета. Сучасні течії еволюції IT-систем сприяють заповненню всіх сфер людського життя. Для встановлення з'єднань та передачі даних найчастіше використовується протокол HTTP. Через його застарілість багато нових проблем не знаходять простих рішень. Можна стикнутися із проблемами на зразок незахищеності передачі даних за допомогою POST або із проблемами під час передачі захищених даних електронною поштою. Але є можливість захистити приватні дані, якщо використити описаний алгоритм. Методи. Використано принцип Шенона шляхом додавання криптографічної «солі» до початкового тексту та розширення його для запобігання перестановкам і спуфінгу, DES-подібний алгоритм (перестановка – використання пароля – перестановка) та зворотний зв'язок за вже зашифрованим текстом із метою надання додаткової стійкості алгоритму. Результати. Описано прикладний алгоритм шифрування даних, що передаються за протоколом HTTP. Наведено фрагменти коду функцій мовою PHP. Ефективність методики доведено шляхом тесту алгоритму. Дискусія. Запропонований у роботі алгоритм дозволяє гарантувати безпечну передачу даних за допомогою протоколу HTTP. Він підходить для роботи з невеликими об'ємами даних, як-от паролі, персональні дані, та не може бути використаний для шифрування великих текстових послідовностей або ж інших типів даних.

***Ключові слова:** протокол HTTP, захищена інформація, захист даних, програмний код, криптографія.*

АЛГОРИТМ ШИФРОВАНИЯ ДАННЫХ HTTP-ПРОТОКОЛА

Цель. Современные течения эволюции ИТ-систем способствуют заполнению всех сфер человеческой жизни. Для установления соединений и передачи данных чаще всего используется протокол HTTP. Вследствие его устарелости многие новые проблемы не находят простых решений. Можно столкнуться с проблемами вроде незащищенности передачи данных с помощью POST или со сложностями при передаче защищенных данных по электронной почте. Но есть возможность защитить частные данные, используя описанный алгоритм. Методы. Использован принцип Шеннона путем добавления криптографической «соли» к исходному тексту и расширение его для предотвращения перестановок и спуфинга, DES-образный алгоритм (перестановка – применение пароля – перестановка) и обратная связь по уже зашифрованному тексту для придания алгоритму дополнительной устойчивости. Результаты. Описан прикладной алгоритм шифрования данных, передаваемых по протоколу HTTP. Приведены фрагменты кода функций на языке PHP. Эффективность методики доказана путем теста алгоритма. Дискуссия. Предложенный в работе алгоритм позволяет обеспечить безопасную передачу данных по протоколу HTTP. Он подходит для работы с небольшими объемами данных, например, паролями, персональными данными, и не может быть использован для шифрования больших текстовых последовательностей или других типов данных.

***Ключевые слова:** протокол HTTP, защищенная информация, защита данных, программный код, криптография.*