НТУУ «КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ»

ТЕПЛОЕНЕРГЕТИЧНИЙ ФАКУЛЬТЕТ

Кафедра АПЕПС

«Засоби захисту інформації»

Лабораторна робота № 4

ШИФРУВАННЯ ДАНИХ З ВИКОРИСТАННЯМ ЛІТЕРАТУРНОГО ФРАГМЕНТУ

*Варіант №12*

*Студента 5 курсу групи ТВ-31с*

Пєхота О.Е.

Перевірив:

кандидат фізико-математичних наук,

доцент Ю.А. Тарнавський

Київ –2014

# Варіант завдання

Ознайомитись з шифруванням даних методом літературного фрагменту і надати його програмну реалізацію

# Хід виконання завдання

1. Створюємо серверний додаток на мові програмування javascript.
2. Підключаємо необхідні бібліотеки для обробки запитів на сервер та методи для шифрування-дешифрування даних *(додаток1)*
3. Створюємо html сторінку для взяємодії з користувачем.
4. Додамо в сторінку візуальні елементи, а також код для обробки дій користувача та відправки запитів на сервер за допомогою мови програмування javascript. *(додаток 2)*
5. Запускаемо сервер та html сторінку користувача.
6. Ввводимо літературний фрагмент в якості ключа та поідомлення для шифрування.

# Висновки

Шифрування методом використання літературного фрагменту – цікавий алгоритм для розшірення кругозору в області алгоритмів шифрування. Але в ньому є багато недоліків. Це досить великий розмір ключа шифрування, а також низька крипто стійкість.

# Додатки

**Додаток 1**

// ciphers.js

, literaryCompositionCipher : function(text, key, inc) {

if(inc) {

var keyLines = key.split("\n").map(function(val, index) {

return { i : index, Value : val}

})/\*.sort(function() {

return .5 - Math.random();

})\*/;

var result = [];

for(var i = 0, n = text.length; i<n;i++) {

var c = text[i];

var found = false;

for(var j = 0; j< keyLines.length; j++) {

var line = keyLines[j];

var colIndex = line.Value.indexOf(c);

if(colIndex === -1) {

continue;

}

found = true;

result.push(j + ' ' + colIndex);

break;

}

if(found === false) {

return 'Symbol `' + c + '` is not present in your message';

}

}

return result.join();

} else {

var input = text;

var letters = input.split(',');

var keyLines = key.split("\n").map(function(val, index) {

return val;

});

var res = '';

for(var i = 0; i < letters.length; i++) {

var letter = letters[i];

var indexes = letter.split(' ');

var row = parseInt(indexes[0]);

var col = parseInt(indexes[1]);

res += keyLines[row][col];

}

return res;

}

}

// server.js

var express = require('express');

var path = require('path'); // модуль для парсинга пути

var log = require('./libs/log')(module);

var app = express();

app.use(express.favicon()); // отдаем стандартную фавиконку, можем здесь же свою задать

app.use(express.logger('dev')); // выводим все запросы со статусами в консоль

app.use(express.bodyParser()); // стандартный модуль, для парсинга JSON в запросах

app.use(express.methodOverride()); // поддержка put и delete

app.use(app.router); // модуль для простого задания обработчиков путей

app.use(express.static(path.join(\_\_dirname, "public"))); // запуск статического файлового сервера, который смотрит на папку public/ (в нашем случае отдает index.html)

app.use(function(req, res, next){

res.status(404);

log.debug('Not found URL: %s',req.url);

res.send({ error: 'Not found' });

return;

});

app.use(function(err, req, res, next){

res.status(err.status || 500);

log.error('Internal error(%d): %s',res.statusCode,err.message);

res.send({ error: err.message });

return;

});

require('./routes')(app);

app.listen(1337, function(){

log.info('Express server listening on port 1337');

});

// routes.js

module.exports = function (app) {

var symbols = require('./libs/symbols')('./../public/symbols.json').allSymbols;

var ciphers = require('./libs/ciphers')(symbols);

function handleData(req, res, flag) {

var message = req.body.message;

var cipher = req.body.cipher;

var result = '';

switch (cipher) {

case 'caesar' :

var key = parseInt(req.body.key);

result = ciphers.caesarCipher(message, key, flag);

break;

case 'trithemius' :

result = ciphers.trithemiusCipher(message, flag);

break;

case 'xor':

var seed = req.body.seed;

result = ciphers.xorCipher(message, seed, flag);

break;

case 'des':

var key = req.body.key;

result = ciphers.desCipher(message, key, flag);

console.log(result);

break;

case 'literaryCompositionCipher':

var key = req.body.key;

result = ciphers.literaryCompositionCipher(message, key, flag);

console.log(result);

break;

default :

result = 'You select wrong cipher!!!';

}

res.send(JSON.stringify({

result: result

})); //\*/

}

app.post('/api/encrypt', function (req, res) {

handleData(req, res, true);

});

app.post('/api/decrypt', function (req, res) {

handleData(req, res, false);

});

app.get('/api', function (req, res) {

res.send('API is running');

});

};

**Додаток 2**

// lab2.html

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="utf-8">

<meta http-equiv="X-UA-Compatible" content="IE=edge">

<meta name="viewport" content="width=device-width, initial-scale=1">

<title>ЛАБОРАТОРНА РОБОТА No 2: МЕТОД ШИФРУВАННЯ ДАНИХ «ШИФР ТРИТЕМІУСА»</title>

<!-- Bootstrap -->

<link href="/bower\_components/bootstrap/dist/css/bootstrap.min.css" rel="stylesheet">

<!-- HTML5 Shim and Respond.js IE8 support of HTML5 elements and media queries -->

<!-- WARNING: Respond.js doesn't work if you view the page via file:// -->

<!--[if lt IE 9]>

<script src="https://oss.maxcdn.com/libs/html5shiv/3.7.0/html5shiv.js"></script>

<script src="https://oss.maxcdn.com/libs/respond.js/1.4.2/respond.min.js"></script>

<![endif]-->

<style>

#progress\_bar {

margin: 10px 0;

padding: 3px;

border: 1px solid #000;

font-size: 14px;

clear: both;

opacity: 0;

-moz-transition: opacity 1s linear;

-o-transition: opacity 1s linear;

-webkit-transition: opacity 1s linear;

}

#progress\_bar.loading {

opacity: 1.0;

}

#progress\_bar .percent {

background-color: #99ccff;

height: auto;

width: 0;

}

</style>

<style type="text/css">

#copy.zeroclipboard-is-hover { background-color: #eee; }

#zeroclipboard-is-active { background-color: #aaa; }

</style>

</head>

<body>

<div class="container" style="padding-top: 10px">

<div class="row" style="margin-bottom: 10px; display: none">

<div class="col-xs-1">

<input id="kk" type="number" class="form-control" value="3">

</div>

</div>

<div class="row" >

<div class="col-md-6">

<textarea id="inputBox" style="width: 100%" class="form-control" rows="3" placeholder="Enter the text to encrypt"></textarea>

<br>

<div id="wrongSymbols" class="alert alert-danger" style="display: none">

</div>

<label>

Or select some text file: <br>

<input type="file" id="files" name="files" />

<br>

<button class="btn btn-default btn-sm" onclick="abortRead();">Cancel read</button>

<button class="btn btn-default btn-sm" onclick="clearFile()">Clear file</button>

<div id="progress\_bar"><div class="percent">0%</div></div>

</label>

<p id="fileDanger" class="bg-danger" style="display: none">Your browser isn't support files api</p>

</div>

<div class="col-md-6">

<textarea id="outputBox" class="form-control" rows="3" disabled placeholder="Here will be encrypted text"></textarea>

</div>

<div class="col-md-3">

<br>

<button data-type="1" class="btn btn-default btn-sm handleText">Encrypt</button>

<button data-type="0" class="btn btn-default btn-sm handleText">Decrypt</button>

<button id="copy" class="btn btn-default btn-sm">Copy result</button>

</div>

</div>

</div>

<!-- jQuery (necessary for Bootstrap's JavaScript plugins) -->

<script src="/bower\_components/jquery/dist/jquery.min.js"></script>

<!-- Include all compiled plugins (below), or include individual files as needed -->

<script src="/bower\_components/bootstrap/dist/js/bootstrap.min.js"></script>

<script src="/bower\_components/zeroclipboard/ZeroClipboard.min.js"></script>

<script>

// main form elements

var input = $('#inputBox');

var output = $('#outputBox');

var kk = $('#kk');

var fileButton = $('#files');

var progress = $('.percent');

var decodeBody = $('#decodeBody');

if (!(window.File && window.FileReader && window.FileList && window.Blob)) {

fileButton.prop('disabled', true);

$('#fileDanger').show();

}

ZeroClipboard.config( { moviePath: '/bower\_components/zeroclipboard/ZeroClipboard.swf' } );

var client = new ZeroClipboard( $('#copy') );

function clearFile() {

fileButton.replaceWith( fileButton = fileButton.clone( true ) );

}

client.on( 'load', function(client) {

// alert( "movie is loaded" );

client.on( 'datarequested', function(client) {

client.setText(output.val());

} );

client.on( 'complete', function(client, args) {

alert("Text copied to clipboard");//: " + args.text );

} );

} );

client.on( 'wrongflash noflash', function() {

ZeroClipboard.destroy();

} );

/\*

kk.click(function() {

input.val('');

output.val('');

fileButton.val('');

decodeBody.html('');

});\*/

var symbols = '';

// var symbols = 'абвгдеёжзийклмнопрстуфхцчшщъыьэюя ,.';

$.getJSON('/symbols.json', function(data) {

$.each( data, function( key, val ) {

console.log(key + ' symbols loaded.');

symbols += val;

});

console.log('Symbols successful loaded')

});

function getStrWithKK(text, kkVal, inc, callback) {

// encrypt

var url = 'http://localhost:1337/api/' + (inc ? 'encrypt' : 'decrypt');

var request = $.ajax({

url : url,

type: "POST",

crossDomain: true,

dataType: "json",

data : { message: text, key: kkVal, cipher: 'trithemius' },

success: function(data) {

callback(null, data);

}

});

}

$('.handleText').on( 'click', function() {

var flag = $(this).data('type') === 1;

var wrongSymbols = [];

var inputStr = input.val();

for(var j = 0; j< inputStr.length; j++) {

if(symbols.indexOf(inputStr.charAt(j)) === -1)

{

if(wrongSymbols.indexOf(inputStr.charAt(j)) === -1) {

wrongSymbols.push(inputStr.charAt(j));

}

}

}

if(inputStr.length > 0) {

if(wrongSymbols.length > 0) {

$('#wrongSymbols').show();

$('#wrongSymbols').html('You entered wrong symbols: ' + wrongSymbols.join(','));

return ;

} else {

$('#wrongSymbols').html('');

$('#wrongSymbols').hide();

}

} else {

$('#wrongSymbols').html('');

$('#wrongSymbols').hide();

}

getStrWithKK(input.val(), parseInt(kk.val()), flag, function(err, data) {

if(err) {

alert('Error with server');

return ;

}

output.val(data.result);

});

});

var reader;

function abortRead() {

reader.abort();

}

function errorHandler(evt) {

switch(evt.target.error.code) {

case evt.target.error.NOT\_FOUND\_ERR:

alert('File Not Found!');

break;

case evt.target.error.NOT\_READABLE\_ERR:

alert('File is not readable');

break;

case evt.target.error.ABORT\_ERR:

break; // noop

default:

alert('An error occurred reading this file.');

}

}

function updateProgress(evt) {

// evt is an ProgressEvent.

if (evt.lengthComputable) {

var percentLoaded = Math.round((evt.loaded / evt.total) \* 100);

// Increase the progress bar length.

if (percentLoaded < 100) {

progress.width(percentLoaded + '%');

progress.text(percentLoaded + '%');

//progress.style.width = percentLoaded + '%';

//progress.textContent = percentLoaded + '%';

}

}

}

function handleFileSelect(evt) {

// Reset progress indicator on new file selection.

progress.width('0%');

progress.text('0%');

reader = new FileReader();

reader.onerror = errorHandler;

reader.onprogress = updateProgress;

reader.onabort = function(e) {

alert('File read cancelled');

};

reader.onloadstart = function(e) {

$('#progress\_bar').addClass('loading');

//document.getElementById('progress\_bar').className = 'loading';

};

reader.onload = function(e) {

// Ensure that the progress bar displays 100% at the end.

progress.width('100%');

progress.text('100%');

setTimeout(function() {

$('#progress\_bar').removeClass('loading');

//"document.getElementById('progress\_bar').className='';"

}, 2000);

};

reader.onloadend = function(evt) {

if (evt.target.readyState == FileReader.DONE) { // DONE == 2

var text = evt.target.result;

input.val(text);

input.trigger('keyup');

}

};

// Read in the image file as a binary string.

reader.readAsText(evt.target.files[0]);

}

fileButton.change(handleFileSelect);

</script>

</body>

</html>

