Ключові слова: інформаційна безпека, ідентифікація, системи контролю та управління доступом, біометрична ідентифікація, зображення особи.

THE SELECTION OF CHARACTERISTIC FRAGMENTS IN THE IMAGE OF A HUMAN FACE

The development of information technology is closely touches on the issues of information security part, which is the control system and access control objectives for information and activities. Control systems and access control for use biometric identification by the image of a human face. However, as practice shows, the control system and access control has one major drawback is the possibility of an attacker tampering with the image of a real person his portrait, that is, the attempt to present a portrait of a real person, which can lead to the penetration of the attacker on the object information activities. The article discusses the implementation of the detector of the human face in the image to the possibilities of its implementation directly by the controller of the camera and the selection of characteristic fragments in the software system to improve the reliability of the identification.

Index terms: information security, identification, access control systems and access control, biometric identification, a facial image.

Швец Валеріян Анатолійович, кандидат технічних наук, доцент, доцент кафедри засобів захисту інформації Національного авіаційного університету.
E-mail: hvan@nau.edu.ua

Швец Валериян Анатольевич, кандидат технических наук, доцент, доцент кафедры средств защиты информации Национального авиационного университета.
Valerian Shvets, PhD, Associate Professor of the Academic Department of of information security tools National Aviation University.

Німченко Тетяна Васильовна, кандидат технічних наук, доцент кафедри засобів захисту інформації Національного авіаційного університету.
E-mail: fiona54@ukr.net
Нимченко Татьяна Васильевна, кандидат технических наук, доцент кафедры средств защиты информации Национального авиационного университета.
Tatyana Nimchenko, PhD, Associate Professor of the Academic Department of information security tools National Aviation University.

Васянович Віталій Васильович аспірант кафедри засобів захисту інформації Національного авіаційного університету.
E-mail: vasianovichv@ukr.net
Васянович Виталий Васильевич аспирант кафедры средств защиты информации. Национального авиационного университета.
Vitaly Vasyanovych, postgraduate student of the Department of information security tools National aviation University.

УДК 003.26.09: 004.056.55

САМОТЕСТУВАННЯ ТА КОНТРОЛЬ ЦІЛІСНОСТІ КЛІЄНТСЬКОГО КОДУ МЕРЕЖНОГО ІНФОРМАЦІЙНОГО РЕСУРСУ

Денис Самойленко

Для убезпечення інформації від несанкціонованого доступу при мережній організації комунікації необхідно реалізовувати засоби перевірки якими програмними засобами було сформовано запит на її одержання. Існуючі засоби дозволяють реалізовувати захист та перевірку на справжність пасивних об'єктів шляхом включення до них цифрових «водяних» знаків. У статті розглянуто ряд способів автоматичного контролю цілісності клієнтського коду мережних інформаційних ресурсів. Показано низьку ефективність способів, побудованих на аналізі HTML коду ресурсу, рекомендовано реалізовувати розподілені засоби самотестування. Запропоновано методику одержання псевдо-поліморфного коду, використовуючи динамічну заміну елементів однакової семантики. Методика випробувана на ряді популярних браузерів, відзначено особливості та застереження щодо її використання. Реалізація запропонованих заходів дозволить покращити інформаційну безпеку мережних ресурсів.

***Ключові слова**: інформаційна безпека, мережні ресурси, захист даних, самотестування, цілісність.*

Постановка проблеми у загальному вигляді. Розвиток інформаційних технологій, трансформація способів спілкування, прискорення обмінних процесів призвели до формування нового типу комунікаційних відносин – інформаційного суспільства. Пріоритетність розвитку суспільства саме такого типу зазначається у відповідному Законі України [1]. Як правило, відображенням особи (фізичної чи юридичної) у інформаційному просторі виступає мережний інформаційний ре-

сурс (МІР), за яким складається «мережний образ» особи та формується відношення до неї. Тенденція переходу до МІР на державному рівні засвідчено початком робіт зі створення Єдиної інформаційно-комунікаційної платформи (ЄІКП) [2].

Захист інформаційних та комунікаційних процесів у МІР в інформаційному суспільстві може бути порівняний із захистом «доброго імені» чи іміджу як окремої особи, так і компанії чи державної установи. Актуальність захисту інформації встановлюється положеннями Доктрини інформаційної безпеки України [3]. Аналіз сучасного стану настроїв у інформаційному суспільстві дозволив сформулювати задачу убезпечення нових інформаційно-комунікаційних систем як «найзначиміший пріоритет при забезпеченні національних інтересів» [4]. Більш того, відсутність державних кордонів для мережі Інтернет дозволила надати поставленій задачі міжнародного значення [5]. У той же час підвищення інтересу до інформаційних та смислових війн значно посилюють актуальність досліджень у зазначеному напрямі.

Аналіз останніх досліджень і публікацій. Особливості та порядок впровадження захисних рішень у інформаційні системи різного типу носять помітний національний відтінок та, у даному випадку, регулюються стандартами і законодавством України [зокр. 6-10], а також чинними міждержавними документами. З точки зору забезпечення безпеки інформації МІР, як інформаційну систему, можна розглядати як набір функціональних послуг безпеки [9, п.7.3]. Перелік необхідних для МІР послуг безпеки встановлюється нормативним документом [10].

За функціональною архітектурою МІР складається з двох активних частин – серверної та клієнтської. Сервер передає до кінцевого споживача код клієнтської частини, сприймає та обробляє запити, що нею сформовані. За відносної зрозумілості процесів захисту серверної частини, особливу небезпеку становить принципова відсутність контролю того, що запит від клієнта формується саме тим програмним кодом, який було передано від сервера, а не було навмисно створено користувачем із нелегальними намірами.

Наявні вимоги щодо контролю цілісності МІР безпосередньо стосуються захисту серверної частини, а також захисту від потенційних модифікацій інформації при її передачі каналами зв'язку [10 пп. 7.2.2-7.2.4]. У той же час, окрема вимога щодо контролю цілісності клієнтського коду у явному вигляді не формулюється.

Також у контексті особливостей клієнт-серверної архітектури програмного коду потребує окремої уваги послуга самотестування [10 п. 7.2.14] для клієнтської частини МІР, оскільки уся її активність відбувається на боці клієнта і може бути ним контрольована.

Метою даної роботи є розвинення та випробування методів реалізації послуг безпеки самотестування та контролю цілісності для клієнтської частини мережного інформаційного ресурсу.

Виклад основного матеріалу. Процес функціонування клієнтської частини МІР можна спрощено уявити як послідовність: завантаження та виконання HTML коду – оброблення дій користувача – формування запиту до сервера – перевантаження сторінки чи її окремих елементів після відповіді сервера. Перше, найлогічніше рішення, яке може бути запропоновано для впровадження перевірки цілісності, вбачається у безпосередньому контролі першої ланки послідовності, тобто HTML коду усієї сторінки (чи лише суттєвих для безпеки елементів), що сприймається браузером.

Можна бути відносно впевненими у тому, що усі клієнти (з різними браузерами) одержують однаковий текстовий документ з HTML кодом, оскільки він передається сервером від одного джерела. Однак, перевірка HTML коду на цілісність, очевидно, буде здійснюватись активними клієнтськими скриптами після завантаження сторінки та її оброблення браузером. Тобто певні активні інструкції повинні запросити у браузера HTML код, що потребує контролю, а браузер, у свою чергу, передає у скрипт код, що був ним оброблений, а не вихідний текст, що був переданий сервером.

З метою перевірки впливу браузера на вихідні коди було проведено випробування для однакових HTML документів та різних, найбільш популярних браузерів. Зміст активної частини полягав у запиті внутрішнього коду для статичних об'єктів (innerHTML) або тіла об'єкту виклику (arguments.callee) – для функцій. Результати засвідчили, що вплив браузера на початковий HTML код полягає у наступному:

1) Різні браузери по-різному сприймають символ розриву (переведення) рядка.

2) Відбувається модифікація текстового вираження стандартних тегів.

3) В окремих випадках спостерігається зміна форматування.

Результати випробувань, що ілюструють відзначений вплив для різних браузерів, зведені у табл. 1. У першому стовпчику таблиці відзначено назви та версії браузерів, у другому – статистика їх використання українськими користувачами (за даними [11]), у третьому – ASCII код (коди), які відповідали символу розриву рядка у початковому HTML документі. У четвертому стовпчику наведено результат відображення внутрішнього тегу розриву рядка (через innerHTML), який у вихідному документі задавався як
. Окремо слід відзначити, що браузер K-Meleon суттєво змінив форматування тіла функції, додавши пробіли та розриви рядків, відсутні у вихідному документі.

Як видно з табл. 1, відмінності у оброблені вихідного коду наявні і відрізняються для різних браузерів. Відповідно, контроль цілісності через аналіз HTML коду елементів документу вимагає урахування специфіки конкретного браузера, що значно ускладнює його програмну реалізацію.

Інша можливість контролю цілісності клієнтського коду може бути запропонована з огляду на принципи, закладені у об'єктно-орієнтовану модель документа (DOM – document object model). У відповідності до DOM, виконання HTML коду браузером супроводжується побудовою об'єктного «документа». Наявність стандартів для процесів створення документів, встановлених консорціумом W3C (World Wide Web Consortium), спонукає дослідити сталість сформованої для однакових HTML кодів DOM структури для різних браузерів та можливість її використання для контролю цілісності.

Таблиця 1

Вплив різних браузерів на однаковий вихідний HTML код

Браузер та версія	Рейтинг,%	Розрив рядка	Тег 	DOM структура
Google Chrome 36.0.1985.125 m	30.90	#10	 	[0]: HEAD HEAD
Opera 12.14	29.97	#10	 	[1]: undefined #text
Mozila Firefox 20.0	27.17	#10	 	[2]: BODY BODY
Yandex 1.5.1106.241	0.93	#10	 	
Apple Safari 5.1.7	0.48	#32	 	
Maxthon 4.4.1.2000	0.24	#10	 	
CoolNovo 2.0.7.11	н/д	#10	 	
SeaMonkey 2.17	н/д	#10	 	
SRWare Iron 25.0.1400.0	н/д	#10	 	
IE 8.0.6001.18702	7.54	#13 #10	 	[0]: HEAD HEAD
K-Meleon 1.5.4	н/д	#32 *	 	[1]: BODY BODY

* з додаванням додаткових символів до тіла функції

Для вирішення поставленої задачі було створено простий HTML документ, який складається з заголовку (HEAD) та тіла (BODY), а також містить функцію аналізу дочірніх елементів об'єкта головного елементу DOM як скриптову функцію:

```
var c=document.documentElement.childNodes,d,str="";
for(var i=0;i<c.length;i++)
{d=c[i]; str+="["+i+"]: "+d.tagName; str+=" "+d.nodeName;}
alert(str);.
```

Результати виконання наведеного вище коду на різних браузерах включено до табл. 1. Як очевидно випливає з табл. 1 аналіз структури DOM також не придатний для контролю цілісності клієнтської частини коду, оскільки кількість дочірніх елементів головного об'єкту різна у різних браузерах. Слід відзначити, що подібні відмінності у побудові об'єктної структури присутні не лише елементу document, а й більшості його нащадків.

За відсутності простого підходу до контролю цілісності усього документу або його структурних частин через їх кодовий вміст чи об'єктне вираження, тобто певним «зовнішнім», окремим засобом перевірки, слід розглянути ідею включення засобів контролю до «внутрішнього» складу програмного коду елементів клієнтської частини МІР, тобто реалізацію розподілених засобів контролю. За такого підходу необхідно виділити елементи МІР, які є критичними з точки зору інформаційної безпеки і потребують контролю цілісності, та реалізовувати зазначену послугу безпеки у кожному з них, тобто, по-суті реалізувати у них засоби самотестування.

До таких критичних об'єктів клієнтської частини МІР слід віднести текстові поля та зображення, спотворення інформації у яких не допускається, а також усі функції, пов'язані з обробленням інформації, що вводиться користувачем.

Контроль цілісності текстових та графічних елементів, частіше за все, здійснюється шляхом включення до них певних унікальних ознак – цифрових «водяних» знаків (ЦВЗ) методами комп'ютерної стеганографії. У роботах [12-13] запропоновано методику динамічної стеганографії, тобто прихованої інтервенції ЦВЗ, що постійно оновлюються і дозволяють відрізнити поточний елемент від застарілого (узятого з попередньо завантаженої сторінки) чи підробленого.

Аналогічна методика може бути запропонована для впровадження у засоби самоконтролю виконавчого коду. Головною ідеєю є використання різних програмних реалізацій для тіла однієї і тієї ж функції при різних сеансах клієнт-серверної взаємодії, тобто впровадження псевдо поліморфного коду. Проте, для адаптації методів, випробуваних для текстових та графічних об'єктів, до активних скриптів необхідно урахувати ряд особливостей.

По-перше, слід зазначити, що у мережному програмуванні прийнято використовувати код, оптимізований для найменшого розміру даних, що передаються від сервера до клієнта. При цьому не допускається використання у якості роздільників декількох пробілів, що унеможливлює відповідну методику ЦВЗ, популярну для текстів. Також застосовуються імена змінних мінімально можливої довжини, не вживаються засоби форматування та оформлення коду. За таких умов потенційна стегоємність коду скриптів є мінімальною.

По-друге, пряме використання популярних текстових та графічних методик ЦВЗ у програмних кодах неможливе, оскільки заміна латинських символів на кириличні у виконавчому коді неприпустима, так само, як не допускається модифікація найменш значущих інформаційних бітів.

Подібно до методу знаків однакового нарису, розвиненому для текстової стеганографії [напр. 12], для програмного коду можна запропонувати метод елементів однакової семантики. Закладений у суть вихідного методу факт того, що заміна знаків англійської абетки на знаки української не змінює сприйняття тексту «на око», проте дозволяє виявити їх програмно, цілком аналогічний твердженню, що заміна програмних елементів однакової семантики (імен змінних, функцій, об'єктів та їх методів чи властивостей, тощо, а також порядку їх оголошення) на інші не змінить функціональність коду, проте дозволить відрізнити один код від іншого. Слід лише зауважити, що зазначена заміна має бути синхронною у всьому коді в межах області видимості відповідних елементів. Асинхронно можливо здійснювати заміну роздільних символів, на зразок «пробіл» – «табуляція».

Змістом стегоповідомлення може бути як закладена послідовність замінених елементів (подібно до методики у роботі [12]), так і хеш-образ самого тіла функції. Другому варіанту слід надати перевагу, оскільки контроль послідовності елементів на клієнтському боці вимагатиме передавання у складі клієнтського коду масиву можливих замін, що розкриватиме ідею стегоалгоритму.

Ураховуючи наведені особливості, принцип контролю цілісності виконавчого коду клієнтської частини МІР за методом елементів однакової семантики можна описати у вигляді алгоритму та проілюструвати на прикладі у наступний спосіб:

1. Сервер формує певний шаблон для тіла функції, використовуючи спеціальні текстові конструкції, що у подальшому будуть замінені на однакові елементи коду. У тілі функції обов'язково повинен бути реалізований код обчислення хеш-образ власного коду, доступного через стандартний для усіх браузерів елемент arguments.callee.toString():

$placeholder='function_WM1_fun(){var_WM1_chk=CryptoJS.SHA1(arguments.callee.toString());alert(chk);}';.

У наведеному коді у шаблоні ($placeholder) передбачається підстановка замість конструкцій _WM1_ довільного роздільного знаку. Функціональність наведеного фрагменту полягає лише у виведені обчисленого хеш образу (CryptoJS.SHA1) у повідомленні (alert). Для реалізації хеш-функції SHA-1 використано зовнішню вільно поширювану бібліотеку CryptoJS v3.1.2 (code.google.com/p/crypto-js)

2. Для конкретного сеансу клієнт-серверної комунікації обирається зміст кожного з елементів для заміни та формується виконавчий код:

$code=str_replace("_WM1_","\t",$placeholder);.

У прикладі використано функцію заміни рядків (str_replace), замість конструкцій _WM1_ виконано підстановку символу табуляції, результат заміни збережено у змінній ($code).

3. Обчислюється хеш-образ сформованого коду і зберігається для кожного клієнта, що звертається до сервера

$hash=sha1($code);.

Для демонстрування використано стандартну хеш-функцію мови PHP (sha1), очевидно, що можливо впровадження довільного криптографічного перетворення. Процес збереження може бути реалізований у довільний спосіб – через механізм сесій, за допомогою файлової системи або сервера баз даних, тощо.

4. Сформований код передається до клієнта у складі активних скриптів

```
echo "<script> $code </script>";.
```

За наявності спеціального запиту від сервера на самотестування або безумовно, у кожному з запитів активні скрипти додають до даних, що передаються, власні контрольні образи.

5. Отримавши відповідь від клієнта сервер аналізує передане та збережене значення образу та приймає рішення щодо результатів контролю цілісності.

Слід зауважити, що у наведеному прикладі використано найпростіші коди, що лише ілюструють сутність алгоритму взаємодії.

Випробування алгоритму самотестування функцій для різних браузерів дозволило встановити наступні особливості:

1. Браузер K-Meleon, як зазначалось вище, здійснює суттєву зміну формату тіла функції, у наслідок чого усі хеш-образи кодів відрізняються від початкових. Слід відзначити, що подібні зміни властиві лише цьому браузеру.

2. У браузерах IE та Opera зберігається вихідний розривний символ між директивою оголошення функції function та її іменем fun (перша позиція підстановок _WM1_). В усіх інших браузерах було помічено примусову заміну цього символу на пробіл. У той же час, в тілі функції подібної заміни не спостерігалось (друга позиція підстановок _WM1_).

3. У браузері Apple Safari використовується примусове додавання пробілу між іменем оголошеної функції – fun() та початком її тіла – {. У інших браузерах подібного не виявлено.

Особливості 2 і 3 слід розглядати як певні обмеження на використання методики динамічної зміни кодів для роздільних символів. Так, заміна першого _WM1_ на постійний пробіл та додавання пробілу перед оголошенням тіла функції у вихідний код дозволило одержати повністю однакові хеш-образи функцій в усіх браузерах з табл. 1, окрім K-Meleon.

Саме K-Meleon продемонстрував принципову відмінність від усіх інших браузерів та необхідність корегування запропонованого алгоритму самотестування «під себе». Особливо дивує той факт, що в описі браузера відзначається використання технології Mozilla, при тому що у інших браузерах з цією технологією зазначених змін не спостерігається. За DOM структурою зазначений браузер також відходить від технології Mozilla, дублюючи структуру документа IE. Хоча алгоритм впливу браузера на код функції легко встановлюється, в силу незначної його популярності (див. табл. 1) модифікація алгоритму самотестування не вбачається за доцільне. Однак, виявлений факт може бути врахований для перевірки алгоритму на інші виключення щодо дієвості.

Аналізуючи отримані результати можна відзначити наступне. Одержуючи цілком однакові HTML коди та відповідаючи єдиним міжнародним мережним стандартам різні браузери порізному будують об'єктну ієрархічну структуру власного документа. При цьому також зазнають змін HTML інструкції, що передаються у створені елементи документа (innerHTML), у т.ч. елементи статичного типу. Зазначені відмінності не дозволяють створити уніфікований засіб контролю цілісності коду клієнтської частини MIP через самоаналіз його внутрішнього вмісту. Відмова від аналізу коду чи структури документа «в цілому» вимагає виділення критичних з позиції безпеки елементів та включення засобів контролю цілісності до їх внутрішнього складу, реалізуючи, по суті, розподілені засоби програмного самотестування.

Для статичних елементів у попередніх роботах запропоновано використання засобів динамічної стеганографії. Для активних елементів (скриптів) пропонується адаптована методика елементів однакової семантики, яка передбачає метаморфози кодів, що передаються клієнту, при кожному обмінному сеансі, без змін їх функціональності та розміру. Випробування методики зі змінами роздільних знаків (на прикладі пробілу та табуляції) засвідчили наявність «індивідуальностей» окремих браузерів та необхідності додержання певних обмежень при оформленні коду. У результаті було складено код, що дозволяв заміну пробіл-табуляція та мав однакові хеш-образи (для кожного варіанту заміни) для усіх браузерів, окрім K-Meleon.

Винятковість браузера K-Meleon засвідчує наявність альтернативних підходів до реалізації стандартизованої функціональності. Через невисоку популярність K-Meleon, без ризику значних втрат потенційних користувачів, може

бути виключений з переліку рекомендованих браузерів для робіт з підвищеними вимогами до інформаційної безпеки. Також можна рекомендувати включати програмну перевірку браузера та версії у коди скриптів, що вимагають контролю цілісності. У будь-якому разі доцільно забезпечувати потенційний моніторинг оновлень та виходів нових браузерів у контексті діездатності реалізованих засобів самотестування клієнтського коду.

Висновки. Досліджено особливості формування структури документа та внутрішніх HTML кодів елементів для різних браузерів, виявлено їх вплив на вихідні тексти. Показано недоцільність реалізовувати контроль цілісності за аналізом структури чи коду документа в цілому, рекомендовано впроваджувати засоби самотестування.

Запропоновано методику елементів однакової семантики для самоконтролю активних скриптів клієнтської частини мережного інформаційного ресурсу, відзначені особливості та обмеження, які уніфікують її для переважної більшості популярних браузерів. У якості прикладу наведено фрагменти серверного коду мовою PHP.

Наявність поодинокого винятку з описаної методики обґрунтовує подальші дослідження зі створення єдиного алгоритму самотестування, прийнятного для довільного клієнтського програмного забезпечення.

ЛІТЕРАТУРА

[1]. Закон України Про Основні засади розвитку інформаційного суспільства в Україні на 2007-2015 роки. [Текст] / Відомості Верховної Ради України. – 2007, № 12, ст.102

[2]. Єдина інформаційно-комунікаційна платформа [Електронний ресурс] / Офіційний сайт Національної комісії, що здійснює державне регулювання у сфері зв'язку та інформатизації (НКРЗІ). – Режим доступу http://www.nkrz.gov.ua/uk/1324635473/1363191045/ (Дата звернення 26.06.14).

[3]. Доктрина інформаційної безпеки України, затверджена Указом Президента України від 8 липня 2009 року N 514/2009. [Текст] / Офіційний вісник Президента України. — 2009, № 20, стор. 18, стаття 677

[4]. Дубов Д.В. Забезпечення національних інтересів держави в інформаційному суспільстві: трансформація пріоритетів. [Текст] / Д.В. Дубов // Стратегічні пріоритети. – 2012. – №3(24). – С. 120-125

[5]. Пилипчук В.Г. Системні проблеми становлення і розвитку науки в інформаційній сфері [Текст] / В.Г. Пилипчук // Стратегічні пріоритети. – 2012. – № 2 (23). – С. 143-148.

[6]. ДСТУ 3396.1-96 Захист інформації. Технічний захист інформації. Порядок проведення робіт.

[7]. Про захист інформації в інформаційно-телекомунікаційних системах: Закон України від 05.07.1994 № 80/94-вр [Електронний ресурс]. – Режим доступу: http://zakon4.rada.gov.ua/laws/show/80/94-%D0%B2%D1%80

[8]. Про затвердження Правил забезпечення захисту інформації в інформаційних, телекомунікаційних та інформаційно-телекомунікаційних системах: Постанова Кабінету Міністрів України від 29.03.2006 № 373 [Електронний ресурс]. – Режим доступу: http://zakon2.rada.gov.ua/laws/show/373-2006-%D0%BF

[9]. НД ТЗІ 1.1-002-99 Загальні положення щодо захисту інформації в комп'ютерних системах від несанкціонованого доступу. [Текст] / НД ТЗІ, затверджений наказом ДСТСЗІ СБ України від 28.04.1999 № 22.

[10]. НД ТЗІ 2.5-010-2003 Вимоги до захисту інформації WEB - сторінки від несанкціонованого доступу. [Текст] / НД ТЗІ, затверджений наказом ДСТСЗІ СБ України від 02.04.2003 № 33.

[11]. Лучшие браузеры. рейтинг браузеров в Украине [Електронний ресурс]. – Режим доступу: www.rooom.com.ua/info-staty/39-raskrutka-seo-site/145-best-browsers-ukraina-statistika.html

[12]. Самойленко Д.М. Використання методів динамічної стеганографії для захисту інформаційних ресурсів [Електронний ресурс] / Самойленко Д.М., Новосьолова К.М. // Вісник НУК (електронне видання) 2013, № 1. – Режим доступу:http://ev.nuos.edu.ua/ru/material?publicationId=22518

[13]. Самойленко Д.М. Комплексна система захисту інформаційного ресурсу [Текст] / Самойленко Д.М. // Інформаційна безпека, 2013. – № 1 (9). с. 147-151 (ISSN 2224-9613)

REFERENCES

[1]. UKRAINE Law of Ukraine (2007) On main tends of information society evolution in 2007-2015. Kyiv: Verkhovna Rada

[2]. UKRAINE National commission for the state regulation of communications and information (2014). [Online] Official web-portal. Available from http://www.nkrz.gov.ua/uk/1324635473/1363191045/ [Accessed: 26th June 2014]

[3]. UKRAINE Law of Ukraine (2009) Doctrine of information security of Ukraine Kyiv: President of Ukraine

[4]. Dubov D.V. (2012) State national interests providing in information society: priorities transformation. Strategic Priority. 3 (24). – p. 120-125.

[5]. Pylypchuk V.G. (2012) System problems of science becoming and evolution in information sphere. Strategic Priority. 2 (23). – p. 143-148.
[6]. NATIONAL STANDARD OF UKRAINE (1997) 3396.1-96 Information protection. Technical protection of information. Order of carrying out the works.
[7]. UKRAINE Law of Ukraine (1994) On information protection in information-telecommunication systems [Online] Available from http://zakon4.rada.gov.ua/laws/show/80/94-%D0%B2%D1%80 [Accessed: 26th June 2014]
[8]. UKRAINE Law of Ukraine (1994) On rules for information protection providing in information, telecommunication and information-telecommunication systems [Online] Available from http://zakon2.rada.gov.ua/laws/show/373-2006-%D0%BF [Accessed: 26th June 2014]
[9]. UKRAINE Security Service of Ukraine (1999) General statements on information security in computer systems from unauthorized access (1.1-002-99). Kyiv: SSU
[10]. UKRAINE Security Service of Ukraine (2004) Requirements for information security of WEB-page from unauthorized access (2.5-010-2003). Kyiv: SSU
[11]. ROOOM DESIGN (2014) The best browsers: rating in Ukraine [Online] Available from www.rooom.com.ua/info-staty/39-raskrutka-seo-site/145-best-browsers-ukraina-statistika.html [Accessed: 26th June 2014]
[12]. Samoilenko D.M., Novoselova K.M. (2013) Dynamic steganographics method usage for information resource protection. [Online] Works of NUoS 1(1) Available from http://ev.nuos.edu.ua/ru/material?publicationId=22518 [Accessed: 26th June 2014]
[13]. Samoilenko D.M (2013) Complex protection system of information resource. Information security, 1 (9). p. 147-151 (ISSN 2224-9613)

САМОТЕСТИРОВАНИЕ И КОНТРОЛЬ ЦЕЛОСТНОСТИ КЛИЕНТСКОГО КОДА СЕТЕВОГО ИНФОРМАЦИОННОГО РЕСУРСА

Для защиты информации от несанкционированного доступа при сетевой организации коммуникаций необходима реализация средств проверки того, какими программными средствами был сформирован запрос на ее получение. Существующие средства позволяют реализацию защиты и проверки подлинности пассивных объектов путем внедрения в них цифровых «водяных» знаков. В статье рассмотрен ряд способов автоматического контроля целостности клиентского кода сетевых информационных ресурсов. Показана низкая эффективность способов, построенных на анализе HTML коду ресурса, приведены рекомендации реализации распределенных средств самотестирования. Предложена методика получения псевдополиморфного кода, используя динамическую замену элементов одинаковой семантики. Методика испытана на ряде популярных браузеров, отмечены особенности и ограничения на ее применение. Реализация предложенных методик позволит улучшить информационную безопасность сетевых ресурсов.

Ключевые слова: информационная безопасность, сетевые ресурсы, защита данных, самотестирование, целостность.

SELF-TESTING AND INTEGRITY CONTROL OF INFORMATION RESOURCE CLIENT CODE

For the information protection from unauthorized access it is necessary to realize methods of checking the authenticity of program from which the request has been formed, especially in case of network communication. Existing methods allow authenticity checking of passive objects (text areas, images, etc) by digital watermarks introduction. In the present work there are shown the series of authenticity checking methods based on HTML code analysis. It was recommended using of distributed methods of program code self-testing in a way close to watermarks. It was proposed a method of pseudo-polymorphic code forming by dynamic replacing of similar semantic elements. The method were tested on several popular browsers, the same hash codes were obtained, there were shown some restrictions of usage. Realization of proposed methods will increase the information security of network resources.

Index terms: information security, network resource, data protection, self-testing, integrity.

Самойленко Денис Миколайович, кандидат фізико-математичних наук, доцент, доцент кафедри електрообладнання суден та інформаційної безпеки. Національний університет кораблебудування імені адмірала Макарова.
E-mail: DenNikSam@gmail.com
Самойленко Денис Николаевич, кандидат физико-математических наук, доцент, доцент кафедры электрооборудования суден и информационной безопасности. Национальный университет кораблестроения имени адмирала Макарова.
Denys Samoilenko, PhD, docent of Ship Electrical Equipment and Information Security Department, National University of Shipbuilding after Admiral Makarov.