ІНФОРМАЦІЙНІ ТЕХНОЛОГІЇ ТА КОМП'ЮТЕРНА ТЕХНІКА

УДК 621.391

О. С. Савенко[1]
С. М. Лисенко[1]
А. О. Нічепорук [1]

ІНФОРМАЦІЙНА ТЕХНОЛОГІЯ ДІАГНОСТУВАННЯ КОМП'ЮТЕРНИХ СИСТЕМ НА НАЯВНІСТЬ ПОЛІМОРФНОГО ПРОГРАМНОГО КОДУ

[1]Хмельницький національний університет

Запропоновано інформаційну технологію виявлення поліморфних вірусів на основі модифікованих емуляторів в корпоративній мережі, що дозволяє здійснювати виявлення нових поліморфних вірусів та копій вже існуючих. З метою підвищення ефективності діагностування в основу інформаційної технології закладено метод формування поведінки поліморфних вірусів та формування бази даних потенційно небезпечних поведінок.

Ключові слова: поліморфний вірус, потенційно небезпечна поведінка, модифіковані емулятори.

Вступ та постановка задачі

Сучасний розвиток технологій написання та впровадження вірусного коду не дозволяє здійснювати його виявлення простими сигнатурними методами [1]. Для виявлення такого типу вірусів більшість антивірусних сканерів використовують евристичні методи виявлення, за характеристичні ознаки яких виступають послідовності викликів API функцій, граф потоку управління програми, структурні особливості PE .EXE файлів, опкоди інструкцій та їх комбінації [2—3]. Проте, зазначені підходи мають низку недоліків: використання поліморфними та метаморфними вірусами обфускації програмного коду, із залученням декількох технік, що супроводжується наявністю в тілі вірусу комбінованих методів заплутування коду, складність алгоритмів аналізу графів (великий обсяг даних відносить задачу виявлення до класу NP повних задач), відсутність механізмів протидії антивідлагоджувальним та антиемуляційним технікам.

Одним із дієвих методів для виявлення поліморфних вірусів (ПВ) є емуляції виконання програмного коду. Проте, на сьогоднішній день шкідливе програмне забезпечення розширює та урізноманітнює набір інструментів та засобів для ухилення виявлення у віртуальному середовищі. Зокрема кількість обходів вірусними програмами віртуального середовища зросло на 2000 % у порівнянні із 2014 роком [4], що свідчить про загрозливу тенденцію поширення вірусних програм.

Таким чином, поширення та розвиток технік ухилення від емуляції ПВ потребує розробки нової інформаційної технології (ІТ), що дала б змогу здійснити виявлення нових ПВ та копій вже існуючих, які здійснюють інфікування виконуваних файлів PE .EXE формату в комп'ютерній системі (КС), що об'єднанні в корпоративну мережу.

З цією метою доцільним є залучення модифікованих емуляторів, розміщених на кожній КС. У зв'язку з нечіткістю та коливанням меж вхідних даних, для аналізу, необхідним є використання нечіткої логіки. Окрім того, з метою підвищення ефективності процесу виявлення необхідною є розробка методу, закладеного в ІТ, що дозволить здійснювати моніторинг дій в КС та формування бази потенційно небезпечних поведінок.

Діагностування КС на наявність ПВ та МВ

Процес діагностування КС на наявність ПВ та МВ складається з двох підпроцесів: процесу визначення підозрілого файлу, який проявляє поведінку вірусної програми, що належить до множини ПВ з формуванням бази потенційно небезпечної поведінки (ПНП) та процесу діагностування

КС процесу виявлення та локалізації ПВ або МВ. Позначимо процес визначення підозрілого файлу через $\Psi = \{\Psi_1, \Psi_2, \Psi_3, \Psi_4\}$, а процес виявлення та локалізації через $S = \{S_1, S_2, S_3, S_4, S_5, S_6\}$.

Процес визначення підозрілості включає етапи: Ψ_1 — відстеження викликів API функцій та зіставлення з правилами оцінки підозрілості; Ψ_2 — визначення кількості послідовних API викликів до множини легітимних або одного із класів нелегітимних API функцій; Ψ_3 — формування поведінки ОД на основі матриці API викликів; Ψ_4 — занесення поведінки до бази ПНП; процес діагностування та локалізації: S_1 — розсилання на інші КС мережі ОД до емуляції; S_2 — процес емуляції; S_3 — дизасемблювання та формування функціональних блоків; S_4 — формування вектора схожості ОД до емуляції та після; S_5 — нечітка класифікація; S_6 — визначення результату, блокування та занесення результату до бази ПНП. Для формалізації процесу діагностування розроблено модель процесу діагностування КС на наявність ПВ [6].

$$M_p = \left\langle \left\langle O, C, f_c, P \right\rangle, \left\langle F_S, F_P, f_e, f_d, f_{fb}^{Fp}, f_{fb}^{Fs}, \overline{V}, C_{pol}, f_{clss} \right\rangle \right\rangle, \quad (1)$$

де для етапів $\Psi_1 - \Psi_4$: $O = \{o_i\}_{i=1}^{N_f}$ — множина об'єктів діагностування, які є файлами формату PE EXE КС, N_f — кількість всіх файлів КС; $C = [0,1]$ — висновок, щодо підозрілості, що приймає два значення $\{syspicious,\ non-syspicious\}$; $f_c = \langle L, N, P \rangle$ функція зіставлення поведінки ОД з чорним та білим списками, L — множина поведінок, що входять до білого списку, N — множина поведінок, що входять до чорного списку; $P = \langle O, S, Q, H, \tau^T, \tau^W \rangle$ — поведінка ОД, де S — початковий стан ОД, $Q = \{q_i\}_{i=1}^{N_o}$ — стани ОД, де N_o — кількість API викликів ОД; H — множина API викликів, що здійснює ОД, причому $H = T \cup W$; $\tau^T : (Q,T) \to \left\{ k^T \middle| t_i^{q_i} = t_{i+1}^{q_{i+1}} \vee t_i^{q_i} \neq t_{i+1}^{q_{i+1}} \right\}$ — функція визначення кількості послідовних API функцій до множини T; $\tau^W : (Q,W) \to \left\{ k^W \middle| w_{i,j}^{q_i} = w_{i+1,j}^{q_{i+1}} \right\}$ — функція визначення кількості послідовних API викликів до j-го класу шкідливих API функцій.

Для етапів $S_1 - S_6$: $f_e(T_e, \Pi_e)$ — функція зміни параметрів емулятора, де $T_e = \{t_j\}_{j=1}^{6}$ — множина антиемуляційних та антивідлагоджувальних технік, $\Pi_e = \{\pi_r\}_{r=1}^{7}$ — множина параметрів емулятора, що підлягають зміні; $F_S = \left\{ f_{s_j} \right\}_{j=1}^{N_i}$ — лістинг дизасембльованих інструкцій ОД до емуляції, N_i — загальна кількість інструкцій; $F_P \left\{ f_{p_j} \right\}_{j=1}^{N_i}$ — лістинг дизасембльованих інструкцій ОД після емуляції, N_i — загальна кількість інструкцій; $f_d : O \to F_S \vee F_P$ — функція здійснення дизасемблювання ОД; $f_{fb}^{Fp} : (F_P) \to F_P^{'}$ — функція формування функціональних блоків ОД до емуляції, де $F_P^{'} = \left\{ f_p^{'} \right\}_{k=1}^{N_{FB}}$, $f_{fb}^{Fs} : (F_S) \to F_S^{'}$ — функція формування функціональних блоків ОД після емуляції, де $F_S^{'} = \left\{ f_s^{'} \right\}_{k=1}^{N_{FB}}$, N_{FB} — кількість функціональних блоків ОД до емуляції; $\overline{V} = \langle V_1, V_2, ..., V_6 \rangle$ — вектор схожості копій поліморфних вірусів; $f_{clss} : O \to \{R_b, C_{pol}\}$ — функція нечіткої класифікації, де R_b — довірений додаток, $C_{pol} = \{c_m\}_{m=1}^{12}$ — множина рівнів поліморфних вірусів.

Інформаційна технологія діагностування комп'ютерних систем на наявність поліморфного програмного коду

Розроблено нову ІТ діагностування КС на наявність поліморфних вірусів в корпоративній мережі. ІТ дозволяє здійснювати виявлення як нових так і копій вже існуючих поліморфних вірусів. В основу запропонованої ІТ входять два методи: метод формування поведінки поліморфних та

метаморфних вірусів та формування бази даних потенційно небезпечних поведінок, Метод виявлення метаморфних вірусів на основі модифікованих емуляторів. Узагальнену схему ІТ показано на рис. 1.

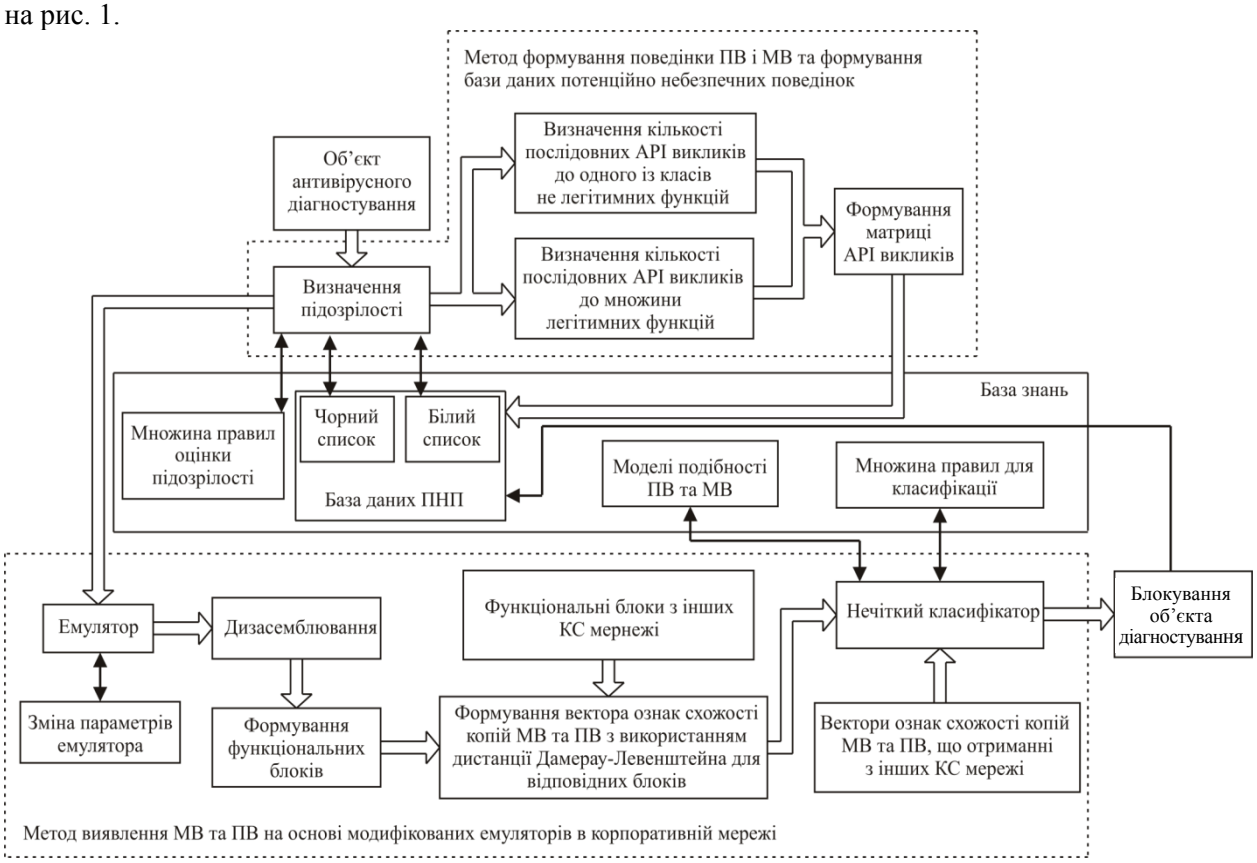

Рис. 1. Узагальнена схема ІТ діагностування КС на наявність ПВ та МВ на основі модифікованих емуляторів

Метод формування поведінки поліморфних та метаморфних вірусів та формування бази даних потенційно небезпечних поведінок

Розроблений новий метод формування поведінки поліморфних та метаморфних вірусів та формування бази даних потенційно небезпечних поведінок є складовою частиною методу виявлення поліморфних та метаморфних вірусів в корпоративній мережі на основі модифікованих емуляторів.

З метою визначення підозрілості невідомої програми та її подальшої ідентифікації здійснюється відслідковування її поведінки, шляхом відстеження системних викликів API функцій. Виокремлено групи потенційно небезпечних поведінкових дій (ПНП) виконуваного файлу, що виконується в ОС. Групування небезпечних поведінкових дій здійснюється на основі впливу на будь-який компонент КС. Кожна група ПНП складається з множини ознак, що представляються у вигляді API функції: множина дії з файлами $\Phi = \{\phi_1, \phi_2, ..., \phi_n\}$, захист від емуляції $Z = \{\xi_1, \xi_2, ..., \xi_k\}$, встановлення $\Theta = \{\theta_1, \theta_2, ..., \theta_l\}$, доступ до мережі Інтернет $I = \{\iota_1, \iota_2, ..., \iota_b\}$, дії з процесами та потоками $\Pi = \{\pi_1, \pi_2, ..., \pi_u\}$, визначення інформації системи $\Xi = \{\zeta_1, \zeta_2, ..., \zeta_m\}$. Тоді, об'єднання множин ПНП API функцій визначимо через *W*, $W = \{\Phi \cup Z \cup \Theta \cup I \cup \Pi \cup \Xi\}$, причому

Поведінкова матриця викликів API функцій

	q1	q2	q3	q4	q5	q6	q7	q8	q9
S	-1	0	0	0	0	0	0	0	0
Z	0	0	0	0	0	+1	-1	0	0
Φ	0	0	0	+1	-1	0	+1	-1	0
П	0	+1	-1	0	0	0	0	0	0
Ξ	0	0	0	0	0	0	0	0	0
I	0	0	0	0	0	0	0	+1	-1
Θ	0	0	0	0	0	0	0	0	0
T	+1	-1	+1	-1	+1	-1	0	0	+1
τ^T	5	0	3	0	7	0	0	0	2
τ^{W_j}	0	1	0	3	0	4	1	2	0

$W \subseteq A^*_{dll}$, де A^*_{dll} — множина всіх API викликів. Представимо поведінку програми у вигляді орієнтованого графа $G = (V, Q)$, вершинами якого виступають групи ПНП, а дуги відображають стани ОД при виклику API функції (див. рис. 1).

Вершина S визначає початок виконання ОД, та у низькорівневому представлені відповідає точці входу у програму, вершина T — визначає виклик легітимної API функції. Здійснюючи послідовний виклик API функцій з цієї множини встановлюється значення τ^T, яке визначає кількість послідовних викликів однієї та/або різних API функції,

$$\tau^T = \begin{cases} p, \text{ якщо } a_2 \in T \wedge a_i \in T \wedge a_{i+1} \in T \wedge \ldots \wedge a_p \in T, \ i = \overline{2, p}; \\ 1, \ a_1 \in T \wedge a_2 \notin T; \\ 0, \text{ інакше,} \end{cases} \quad (2)$$

де p — кількість послідовних API викликів. З виразу (2) випливає, що за виконання першої умови значення τ^T буде більше одиниці і буде представляти кількість послідовних викликів API функцій в деякому стані ОД q_i. Перехід до стану ОД q_{i+1} буде відбуватись при виклику API функції з множини W. В обчисленні значення τ^T не враховується тип та призначення легітимних API функцій, оскільки вони однозначно визначають поведінку невідомої програми як нешкідливу.

Вершини $\Phi, Z, \Theta, I, \Pi, \Xi$ визначають групи ПНП API функцій. Якщо ребро входить у одну із вершин з множини W, то встановлюється значення τ^{W_j}, яке визначається таким чином:

$$\tau^{W_j} = \begin{cases} p_{W_j}, \text{ якщо } a_2 \in W_j \wedge a_i \in W_j \wedge a_{i+1} \in W_j \wedge \ldots \wedge a_{p_{W_j}} \in W_j, \ i = \overline{2, p_{W_j}}; \\ 1, \ a_1 \in W \wedge a_2 \notin W; \\ 0, \text{ інакше,} \end{cases} \quad (3)$$

де p_{W_j} — кількість послідовних API викликів до j-ї вершини з множини W, $j = \overline{1,6}$. Таким чином, отримана поведінка у вигляді поведінкової матриця викликів API функцій для графа $G = (V, Q)$ заноситься до бази ПНП, а ОД, якому притаманна ця поведінка надходить в систему виявлення поліморфних та метаморфних вірусів на основі модифікованих емуляторів.

Метод виявлення метаморфних вірусів на основі модифікованих емуляторів

Розроблено новий метод виявлення поліморфних та метаморфних вірусів в корпоративній мережі на основі модифікованих емуляторів [5]. Розроблений метод використовує емуляцію виконання на кожному хості в мережі, використовуючи модифіковані емулятори. Хости — це мережні станції для обробки інформації, що поєднанні у локальну мережу. Основними функціями хостів є здійснення одноразової емуляції виконання невідомої програми та відправлення результатів на серверну частину.

Метод передбачає виконання таких кроків. На першому етапі здійснюється дизасемблювання ОД P та отримання зразка коду F_P з подальшим розбиттям на функціональні блоки (ФБ) $B_1, B_2, ..., B_g$ зразка коду F_P. З метою отримання зміненої ОД версії виконується емуляція та отримання зміненої версії ОД F_S. На наступному етапі здійснюється дизасемблювання зразка коду F_S та отримання ФБ $B_1, B_2, ..., B_h$. На основі попарного порівняння множини $B_1, B_2, ..., B_g$ та $B_1, B_2, ..., B_h$ з використанням алгоритму Вагнера–Фішера (4), формування векторів ознак схожості для зразків коду до емуляції F_P та після F_S (5)

$$\overline{S_{F_P F_S}} = \langle dL, T, D, I, R, M \rangle, \quad (4)$$

де dL — відстань Дамерау–Левенштейна для функціонального блоку між програмами F_p та F_s; T — кількість необхідних операцій обміну опкодів для приведення блоку програми F_p у F_s $(F_p = F_s)$;

D — кількість необхідних операцій видалення опкоду; I — кількість необхідних операцій вставки опкоду; R — кількість необхідних операцій заміни відповідних опкодів; M — кількість збігів між опкодами в функціональному блоці програми F_p та F_s.

$$OPT = \begin{cases} 0, & i = 0,\ j = 0; \\ i, & j = 0,\ i > 0; \\ j, & i = 0,\ j > 0; \\ \min \begin{cases} OPT(i, j-1) + w(a, \varepsilon) \\ OPT(i, -1j) + w(\varepsilon, b) \\ OPT(i, -1, j-1) + w(a, b) \\ OPT(i-2, j-2) + w(b, a) \end{cases} & j > 0,\ i > 0. \end{cases} \quad (5)$$

На наступному етапі здійснюється відправлення вектора $\overline{S_{F_P F_S}}$ на сервер для класифікації. Результатом роботи системи є ступінь належності кожної копії до одного із класів поліморфних вірусів. Далі здійснюється розсилка кожному хосту у мережі зразка коду F_P та ОД P, причому параметри налаштування модифікованих емуляторів будуть різні. Описані кроки алгоритму, здійснюються на кожному хості у мережі. Порівняльний аналіз ефективності діагностування відомих АЗ із розробленим ПЗ, що базується на основі запропонованої ІТ подано на рис. 2.

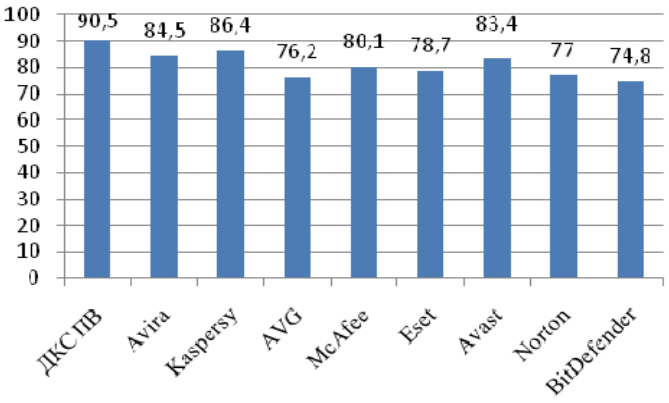

Рис. 2. Ефективність діагностуваня у порівнянні з відомими засобами

Висновки

Розроблено модель процесу діагностування комп'ютерних систем на наявність поліморфних вірусів, що базується на основі поведінкових характеристик поліморфних вірусів. Розроблено метод виявлення поліморфних вірусів на основі модифікованих емуляторів дозволяє здійснювати виявлення копій вже існуючих та нових поліморфних вірусів основною відмінністю від відомих підходів якого є урахування антиемуляційних та антивідлагоджувальних технологій, що використовуються в поліморфних та метаморфних вірусах. Розроблено інформаційну технологію діагностування комп'ютерних систем на наявність поліморфного вірусу на основі якої, реалізовано ПЗ. Отримані результати дослідження показали підвищення достовірності виявлення на рівні 5—15,7 %.

СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ

1. Lin D. Hunting for Undetectable Metamorphic Viruses / D. Lin, M. Stamp // Journal in Computer Virology. — 2011. — Vol. 7, issue 3. — P. 201—214.

2. Vinod P. Scattered Feature Space for Malware Analysis / P. Vinod, V. Laxmi, M. S. Gaur // Communications in Computer and Information Science. — 2011. — Vol. 190. — P. 562—571.

3. Lee J. Detecting Metamorphic Malwares Using Code Graphs / J. Lee, K. Jeong, H. Lee // In proc. ACM Symposium on Applied Computing, NY. — 2010. — P. 1970—1977.

4. Kruegel C. Evasive Malware Exposed and Deconstructed / C. Kruegel // RSA Conference, November, 2015. — P. 12—20.

5. Pomorova O. Metamorphic Viruses Detection Technique based on the Modified Emulators / O. Pomorova, O. Savenko, S. Lysenko, A. Nicheporuk // In Proc. ICT in Education, Research and Industrial Applications. Integration, Harmonization and Knowledge Transfer, Vol. 1614, Kyiv, June 2016. — P. 375—383.

6. Савенко О. С. Модель процесу діагностування комп'ютерних систем на наявність поліморфного та метаморфного програмного коду / О. С. Савенко, С. М. Лисенко, А. О. Нічепорук // Інформаційні технології та комп'ютерна інженерія. — 2014. — № 6. — С. 46—51.

Рекомендована Центром інформаційних технологій і захисту інформації ВНТУ

Стаття надійшла до редакції 4.10.2016

Савенко Олег Станіславович — канд. техн. наук, доцент, доцент кафедри системного програмування;
Лисенко Сергій Миколайович — канд. техн. наук, доцент, доцент кафедри системного програмування;
Нічепорук Андрій Олександрович — аспірант кафедри системного програмування e-mail: andrey.nicheporuk@gmail.com .
Хмельницький національний університет, Хмельницький

O. S. Savenko[1]
S. M. Lysenko[1]
A. O. Nicheporuk[1]

Information Technology of Diagnosing Computer Systems for the Polymorphic Code

[1]Khmelnytskyi National University

The paper presents information technology of detection of polymorphic viruses based of the modified emulators on a corporate network, allows to realize detection of the new polymorphic viruses and copies which are already existing. For the purpose of increase in efficiency of diagnostics in a basis of an information technology the method of forming of behavior of polymorphic viruses and formation of the database potentially of dangerous behavior has been presented.

Keywords: polymorphic virus, potentially dangerous behavior, modified emulators.

Savenko Oleg S. — Cand. Sc. (Eng.), Assistant Professor, Assistant Professor of the Chair of System Programming;
Lysenko Sergii M. — Cand. Sc. (Eng.), Assistant Professor, Assistant Professor of the Chair of System Programming;
Nicheporuk Andrii O. — Post-Graduate Student of the Chair of System Programming, e-mail: andrey.nicheporuk@gmail.com

О. С. Савенко[1]
С. М. Лысенко[1]
А. О. Ничепорук[1]

Информационная технология диагностики компьютерных систем на наличие полиморфного программного кода

[1]Хмельницкий национальный университет

Предложена информационная технология обнаружения полиморфных вирусов на основе модифицированных эмуляторов в корпоративной сети, позволяющая осуществлять обнаружение новых полиморфных вирусов и копий уже существующих. С целью повышения эффективности диагностики в основу информационной технологии заложен метод формирования поведения полиморфных вирусов и формирования базы данных потенциально опасных поведений.

Ключевые слова: полиморфный вирус, потенциально опасное поведение, модифицированные эмуляторы.

Савенко Олег Станиславович — канд. техн. наук, доцент, доцент кафедры системного программирования;
Лысенко Сергей Николаевич — канд. техн. наук, доцент, доцент кафедры системного программирования;
Ничепорук Андрей Александрович — аспирант кафедры системного программирования, e-mail: andrey.nicheporuk@gmail.com