УДК 004.891.3: 004.3

КЛАСИФІКАЦІЯ ВІДМОВ ТА ВРАЗЛИВОСТЕЙ СИСТЕМНОГО ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

Т.О.ГОВОРУЩЕНКО, А.В.МЕВША, В.А.КРИСЬКОВ
Хмельницький національний університет

У даній статті розроблено класифікацію відмов та вразливостей системного програмного забезпечення, математичні моделі відмови та вразливості, критерії та продукційні правила класифікації відмов та вразливостей, які надають можливість полегшення процесу ідентифікації помилок у СПЗ.

In this article the authors developed the classification of failures and vulnerabilities of software system, the mathematical models of failure and vulnerability, the criteria and the production rules of classification of failures and vulnerabilities. The proposed solutions provide to facilitate the process of system software bugs identification.

Ключові слова: програмне забезпечення (ПЗ), системне програмне забезпечення (СПЗ), відмова СПЗ, вразливість СПЗ.

Вступ (сучасний стан галузі підвищення надійності та безпеки системного програмного забезпечення). СПЗ – це комплекс програм, які забезпечують роботу інших програм і керують апаратними ресурсами системи [1]. Чим досконалішим є СПЗ, тим комфортніше почувається користувач у системному середовищі [2].

З точки зору надійності та безпеки комп'ютера в цілому, найбільшу загрозу становлять саме помилки СПЗ. Так, наприклад, помилка переповнення буферу може викликати повну відмову системи (проблема надійності) або дозволити професійно написаному вірусу перехопити керування комп'ютером (проблема безпеки). Прикладні програми теж містять множину дефектів, однак ці помилки можуть викликати лише обмежену шкоду, якщо помилок не містить СПЗ.

Сучасне СПЗ має особливості, які роблять його ненадійним та небезпечним - воно величезне та має дуже слабку локалізацію помилок [3]. Наприклад, ядро операційної системи (ОС) Linux має більше 2,5 мільйонів рядків коду, а ядро ОС Windows XP – вдвічі більше. При цьому ОС містять сотні тисяч процедур, які скомпоновані разом як єдина двійкова програма, функціонуюча в режимі ядра.

Дослідження, проведене у [4], показує, що із зростанням розміру ПЗ збільшується щільність помилок у ПЗ, і для ПЗ розміром більше 512К складає 4-100 помилок на 1000 рядків коду. Інше дослідження [5] надає щільність від 2 до 75 помилок на 1000 рядків виконуваного коду в залежності від розміру модуля. Використовуючи помірну оцінку в 4 помилки на 1000 рядків коду, маємо 10000 можливих помилок у ядрі ОС Linux і 20000 можливих помилок у ядрі ОС Windows XP. Ще більшою проблемою є те, що зазвичай близько 70% операційної системи складається з драйверів, які мають помилок в 3-7 разів більше, ніж простий код [6], тому підрахована кількість помилок, ймовірно, дуже недооцінена. Зрозуміло, що знаходження і виправлення всіх цих помилок не є здійсненним. Аналіз мобільної ОС Android 2.0 показав наявність 359 програмних помилок, зокрема 88 помилок з високим ступенем ризику та 271 помилка з середнім ступенем ризику, що є серйозною загрозою для безпеки ОС [7].

Щодо слабкості локалізації помилок в ОС, то кожен з мільйонів рядків коду ядра може переписувати основні структури даних, які використовуються його незалежними компонентами, виводячи систему з ладу способами, які важко виявити. Крім того, якщо вірус інфікує одну процедуру ядра, то не існує способу втримати його від швидкого поширення на інші процедури та запобігти захопленню комп'ютера в цілому [3].

Наразі є різноманітні підходи, використовувані для підвищення надійності та безпеки ОС: 1) підхід Nooks [8], який захищає ядро від драйверів пристроїв шляхом розташування кожного драйвера в оболонку із захисного ПЗ; 2) підхід паравіртуалізації [9], який дозволяє одночасний запуск на одному комп'ютері декількох операційних систем за допомогою механізму віртуальних машин; 3) підхід створення мультисерверних операційних систем [10], при якому лише мікроядро запускається в режимі ядра, а вся інша частина ОС працює в режимі користувача як набір повністю ізольованих серверних процесів та драйверів; 4) підхід Microsoft Research [3], який відкидає концепцію ОС як єдиної програми, запущеної в режимі ядра, з деяким набором процесів користувача, запущених в режимі користувача, і замінює її на систему, написану мовами нового типу, які забезпечують безпеку типів. Всі ці підходи базуються на запобіганні повної відмови системи, спричиненої множиною помилок у драйверах пристроїв, але не спрямовані на прогнозування і усунення відмов та вразливостей системного програмного забезпечення.

При численних дослідженнях в галузі підвищення надійності та безпеки СПЗ, які ведуться роками, при побудові різноманітних підходів багатьма провідними вченими галузі, вразливості та відмови СПЗ, як і раніше, існують і проявляються у вигляді зависання ОС, неможливості завантаження, «втрати» певних периферійних пристроїв. Наслідки недостатньої безпеки СПЗ [11-15] представлені на рис.1.

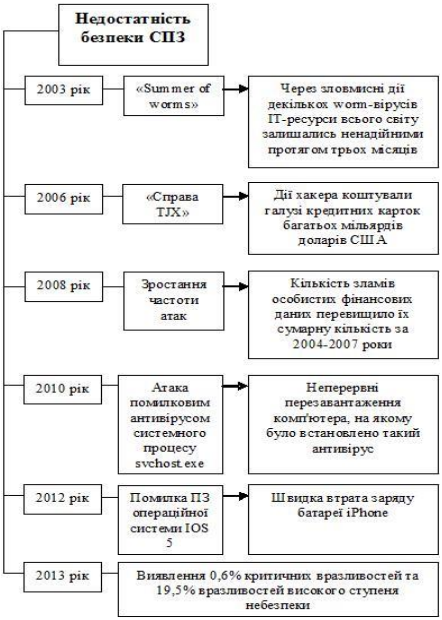

Рис. 1. Наслідки недостатньої безпеки СПЗ

Отже, поведінка СПЗ є недетермінованою через помилки, успішність спроб забезпечення безпеки ПЗ можливе лише за рахунок покращення якості ПЗ та скорочення кількості помилок, для чого слід полегшити процес ідентифікації помилок, тому *актуальною задачею* наразі є прогнозування відмов та вразливостей СПЗ, для якої спочатку слід вирішити задачу класифікації відмов та вразливостей СПЗ.

Постановка задачі. З результатів аналізу сучасного стану галузі слідує, що перспективними напрямками досліджень є: 1) побудова математичних моделей відмов та вразливостей СПЗ; 2) побудова критеріїв та продукційних правил класифікації відмов та вразливостей СПЗ.

1. Математичні моделі відмови та вразливості системного програмного забезпечення. З точки зору теорії надійності *відмова* (failure) – це подія, яка полягає у порушенні роботоздатного стану об'єкта, внаслідок чого система припиняє виконувати цілком або частково свої функції, тобто подія, яка полягає у переході об'єкту з одного рівня роботоздатності або функціювання на інший, більш низький, або в повністю нероботоздатний стан [16].

Визначення 1. Неістотною відмовою If з точки зору СПЗ називатимемо припинення функціювання програми на час, який перевищує заданий поріг, без втрати даних та вимоги перезавантаження комп'ютера.

Визначення 2. Істотною відмовою Sf з точки зору СПЗ називатимемо припинення функціювання програми на час, який перевищує заданий поріг, із втратою всіх або частини даних, але без вимоги перезавантаження комп'ютера.

Визначення 3. Критичною відмовою Cf з точки зору СПЗ називатимемо припинення функціювання програми, яке вимагає перезавантаження ЕОМ, на якій функціонує ПЗ.

Тоді математична модель відмови має наступний вигляд:

$$F = \begin{cases} If \;\mid\; \exists(s_i \in S) \wedge (D_i = D_{i-1}) \\ Sf \;\mid\; \exists(s_i \in S) \wedge (D_i < D_{i-1}), \\ Cf \;\mid\; \exists(s_i \notin S) \end{cases} \quad (1)$$

де i - момент часу після відмови СПЗ; $(i-1)$ - момент часу до відмови СПЗ; s_i - стан СПЗ після відмови СПЗ; $S = \{s_1,...,s_n\}$ - множина роботоздатних станів СПЗ (n - загальна кількість роботоздатних станів СПЗ); D_i - множина даних після відмови СПЗ; D_{i-1} - множина даних до відмови СПЗ.

В контексті функційної безпеки в центрі уваги опиняються функційні можливості ПЗ, використання яких може порушити його вірну роботу, а також цілісність, доступність або конфіденційність інформації.

В комп'ютерній безпеці термін «*вразливість*» (vulnerability) використовується для позначення недоліку системи, використовуючи який можна навмисне порушити її цілісність та викликати невірну роботу. Вразливість може бути результатом помилок програмування,

недоліків проектування системи, ненадійних паролів, вірусів та інших шкідливих програм [14].

Стандарт інформаційної безпеки [17] вводить спеціальне поняття для визначення вразливості СПЗ – недекларовані можливості. Недекларовані можливості – це функційні можливості ПЗ, які не описані або не відповідають описаним в документації, при використанні яких можливе порушення конфіденційності, доступності або цілісності оброблюваної інформації. Проте термін «вразливість» є більш широким та вживаним, тому надалі використовуватимемо саме цей термін.

Визначення 4. Вразливістю вірної роботи Vcw називатимемо функційну можливість, яка призводить до припинення функціювання програми на час, який перевищує заданий поріг, тобто до відмови СПЗ.

Визначення 5. Вразливістю цілісності інформації Vii називатимемо функційну можливість ПЗ, яка призводить до втрати повноти даних, до несанкціонованої (зловмисної або випадкової) зміни даних при виконанні певної операції, пов'язаної із даною функційною можливістю.

Визначення 6. Вразливістю конфіденційності інформації Vic називатимемо функційну можливість ПЗ, яка призводить до витоку, несанкціонованого розголошення, незаконного доступу або використання будь-якої інформації.

Визначення 7. Вразливістю доступності інформації Via називатимемо функційну можливість ПЗ, яка призводить до такого стану, при якому суб'єкти, які мають права доступу до інформації, не можуть реалізувати їх без перешкод.

Очевидно, що вразливості можуть бути комплексними – наприклад, вразливість, яка одночасно загрожує цілісності та конфіденційності інформації, або вразливість, яка одночасно загрожує вірній роботі СПЗ, цілісності, конфіденційності та доступності інформації і т.і.

Враховуючи вищевикладене, побудуємо математичну модель вразливості СПЗ з точки зору проявів вразливості – формула (2), де: $FT = \{ft_1 ... ft_m\}$ - множина всіх функційних можливостей СПЗ; m - загальна кількість всіх функційних можливостей СПЗ; j - момент часу після виконання функційної можливості із вразливістю; $(j-1)$-

момент часу до виконання функційної можливості із вразливістю; D_j - множина даних після виконання функційної можливості із вразливістю; D_{j-1} - множина даних до виконання функційної можливості із вразливістю; I_j - множина даних, які можуть бути використані після виконання функційної можливості із вразливістю; I_{j-1} - множина даних, які можуть бути використані до виконання функційної можливості із вразливістю; A_j - множина даних, які можуть бути безперешкодно використані суб'єктом, який має права доступу до них, після виконання функційної можливості із вразливістю; A_{j-1} - множина даних, які можуть бути безперешкодно використані суб'єктом, який має права доступу до них, до виконання функційної можливості із вразливістю.

$$V = \begin{cases} Vcw \quad \forall(Vcw \in FT) \wedge \exists(Vcw \mapsto F) \\ Vii \quad \forall(Vii \in FT) \wedge \exists(Vii \mapsto (D_j \neq D_{j-1})) \\ Vic \quad \forall(Vic \in FT) \wedge \exists(Vic \mapsto (I_j \neq I_{j-1}) \wedge (I_j > I_{j-1})) \\ Via \quad \forall(Via \in FT) \wedge \exists(Via \mapsto (A_j \neq A_{j-1}) \wedge (A_j < A_{j-1})) \\ Vcw \wedge Vii \\ Vcw \wedge Vic \\ Vcw \wedge Via \\ Vii \wedge Vic \\ Vii \wedge Via \\ Vic \wedge Via \\ Vcw \wedge Vii \wedge Vic \\ Vcw \wedge Vic \wedge Via \\ Vcw \wedge Vii \wedge Via \\ Vii \wedge Vic \wedge Via \\ Vcw \wedge Vii \wedge Vic \wedge Via \end{cases} . \quad (2)$$

2. Критерії та продукційні правила класифікації відмов і вразливостей системного програмного забезпечення. Враховуючи запропоновані визначення типів відмов, запропонуємо наступні *критерії класифікації відмов*:

1) втрата роботоздатності (можливості функціювання) СПЗ;
2) втрата даних (всіх або частини) СПЗ;
3) необхідність перезавантаження ЕОМ.

Враховуючи запропоновані визначення типів вразливостей, запропонуємо наступні *критерії класифікації вразливостей*:

1) виникнення відмови СПЗ;
2) втрата повноти даних;
3) витік недозволеної інформації;
4) неможливість одержання дозволеної інформації.

Використовуючи розроблену математичну модель відмови, а також запропоновані критерії класифікації відмов СПЗ, побудуємо *продукційні правила класифікації відмов*:

1) якщо стан s_i СПЗ після припинення функціювання є роботоздатним ($si \in S$) та за час припинення функціювання не відбулось втрати даних, тобто $D_i = D_{i-1}$, то відмова є неістотною;

2) якщо стан s_i СПЗ після припинення функціювання є роботоздатним ($si \in S$), але за час припинення функціювання відбулась втрата даних, тобто $D_i < D_{i-1}$, то відмова є істотною;

3) якщо стан s_i СПЗ після припинення функціювання не є роботоздатним ($si \notin S$), то відмова є критичною.

Використовуючи розроблену математичну модель вразливості та запропоновані критерії класифікації вразливостей, побудуємо *продукційні правила класифікації вразливостей*:

1) якщо під час виконання h-ї функційної можливості системне ПЗ припинило функціювання на час t_h, який перевищує заданий для СПЗ даного типу пороговий час $t_{\lim}$ ($t_h > t_{\lim}$), то h-а функційна можливість СПЗ є вразливістю вірної роботи;

2) якщо після виконання h-ї функційної можливості СПЗ відбулась втрата повноти даних, тобто $D_{j_h} \neq D_{j-1_h}$, то h-а функційна можливість СПЗ є вразливістю цілісності інформації;

3) якщо після виконання h-ї функційної можливості СПЗ відбувся витік даних, тобто $I_{j_h} \neq I_{j-1_h}$, причому $I_{j_h} > I_{j-1_h}$, то h-а функційна можливість СПЗ є вразливістю конфіденційності інформації;

4) якщо після виконання h-ї функційної можливості СПЗ виникла неможливість одержання дозволеної користувачу інформації, тобто $A_{j_h} \neq A_{j-1_h}$, причому $A_{j_h} < A_{j-1_h}$, то h-а функційна можливість СПЗ є вразливістю доступності інформації;

5) якщо під час виконання h-ї функційної можливості системне ПЗ припинило функціювання на час t_h, який перевищує заданий для СПЗ даного типу пороговий час $t_{\lim}$ ($t_h > t_{\lim}$), а також відбулась втрата повноти даних, тобто $D_{j_h} \neq D_{j-1_h}$, то h-а функційна можливість СПЗ є вразливістю вірної роботи та цілісності інформації;

6) якщо під час виконання h-ї функційної можливості системне ПЗ припинило функціювання на час t_h, який перевищує заданий для СПЗ даного типу пороговий час $t_{\lim}$ ($t_h > t_{\lim}$), а також відбувся витік даних, тобто $I_{j_h} \neq I_{j-1_h}$, причому $I_{j_h} > I_{j-1_h}$, то h-а функційна можливість СПЗ є вразливістю вірної роботи та конфіденційності інформації;

7) якщо під час виконання h-ї функційної можливості системне ПЗ припинило функціювання на час t_h, який перевищує заданий для СПЗ даного типу пороговий час $t_{\lim}$ ($t_h > t_{\lim}$), а також виникла неможливість одержання дозволеної користувачу інформації, тобто $A_{j_h} \neq A_{j-1_h}$, причому $A_{j_h} < A_{j-1_h}$, то h-а функційна можливість СПЗ є вразливістю вірної роботи та доступності інформації;

8) якщо після виконання h-ї функційної можливості СПЗ відбулась втрата повноти даних, тобто $D_{j_h} \neq D_{j-1_h}$, а також відбувся витік даних, тобто $I_{j_h} \neq I_{j-1_h}$, причому $I_{j_h} > I_{j-1_h}$, то h-а

функційна можливість СПЗ є вразливістю цілісності та конфіденційності інформації;

9) якщо після виконання h-ї функційної можливості СПЗ відбулась втрата повноти даних, тобто $D_{j_h} \neq D_{j-1_h}$, а також виникла неможливість одержання дозволеної користувачу інформації, тобто $A_{j_h} \neq A_{j-1_h}$, причому $A_{j_h} < A_{j-1_h}$, то h-а функційна можливість СПЗ є вразливістю цілісності та доступності інформації;

10) якщо після виконання h-ї функційної можливості СПЗ відбувся витік даних, тобто $I_{j_h} \neq I_{j-1_h}$, причому $I_{j_h} > I_{j-1_h}$, а також виникла неможливість одержання дозволеної користувачу інформації, тобто $A_{j_h} \neq A_{j-1_h}$, причому $A_{j_h} < A_{j-1_h}$, то h-а функційна можливість СПЗ є вразливістю конфіденційності та доступності інформації;

11) якщо під час виконання h-ї функційної можливості системне ПЗ припинило функціювання на час t_h, який перевищує заданий для СПЗ даного типу пороговий час $t_{\lim}$ ($t_h > t_{\lim}$), відбулась втрата повноти даних, тобто $D_{j_h} \neq D_{j-1_h}$, а також відбувся витік даних, тобто $I_{j_h} \neq I_{j-1_h}$, причому $I_{j_h} > I_{j-1_h}$, то h-а функційна можливість СПЗ є вразливістю вірної роботи, цілісності та конфіденційності інформації;

12) якщо під час виконання h-ї функційної можливості системне ПЗ припинило функціювання на час t_h, який перевищує заданий для СПЗ даного типу пороговий час $t_{\lim}$ ($t_h > t_{\lim}$), відбувся витік даних, тобто $I_{j_h} \neq I_{j-1_h}$, причому $I_{j_h} > I_{j-1_h}$, а також виникла неможливість одержання дозволеної користувачу інформації, тобто $A_{j_h} \neq A_{j-1_h}$, причому $A_{j_h} < A_{j-1_h}$, то h-а функційна можливість СПЗ є вразливістю вірної роботи, конфіденційності та доступності інформації;

13) якщо під час виконання h-ї функційної можливості системне ПЗ припинило функціювання на час t_h, який перевищує заданий для СПЗ даного типу пороговий час $t_{\lim}$ ($t_h > t_{\lim}$), відбулась

втрата повноти даних, тобто $D_{j_h} \neq D_{j-1_h}$, а також виникла неможливість одержання дозволеної користувачу інформації, тобто $A_{j_h} \neq A_{j-1_h}$, причому $A_{j_h} < A_{j-1_h}$, то h-а функційна можливість СПЗ є вразливістю вірної роботи, цілісності та доступності інформації;

14) якщо після виконання h-ї функційної можливості СПЗ відбулась втрата повноти даних, тобто $D_{j_h} \neq D_{j-1_h}$, відбувся витік даних, тобто $I_{j_h} \neq I_{j-1_h}$, причому $I_{j_h} > I_{j-1_h}$, а також виникла неможливість одержання дозволеної користувачу інформації, тобто $A_{j_h} \neq A_{j-1_h}$, причому $A_{j_h} < A_{j-1_h}$, то h-а функційна можливість СПЗ є вразливістю цілісності, конфіденційності та доступності інформації;

15) якщо під час виконання h-ї функційної можливості системне ПЗ припинило функціювання на час t_h, який перевищує заданий для СПЗ даного типу пороговий час $t_{\lim}$ ($t_h > t_{\lim}$), відбулась втрата повноти даних, тобто $D_{j_h} \neq D_{j-1_h}$, відбувся витік даних, тобто $I_{j_h} \neq I_{j-1_h}$, причому $I_{j_h} > I_{j-1_h}$, а також виникла неможливість одержання дозволеної користувачу інформації, тобто $A_{j_h} \neq A_{j-1_h}$, причому $A_{j_h} < A_{j-1_h}$, то h-а функційна можливість СПЗ є вразливістю вірної роботи, цілісності, конфіденційності та доступності інформації.

Висновки. У статті виконано аналіз сучасного стану галузі підвищення надійності та безпеки системного програмного забезпечення, в результаті якого визначено особливості СПЗ, які роблять його ненадійним та небезпечним. Проведено аналіз підходів, використовуваних наразі для підвищення надійності та безпеки СПЗ, який показав, що підходи базуються на запобіганні повної відмови системи, але не спрямовані на прогнозування і усунення відмов та вразливостей системного програмного забезпечення. В роботі доведено, що при численних дослідженнях в галузі підвищення надійності та безпеки СПЗ, які ведуться роками, при побудові різноманітних підходів багатьма провідними вченими галузі, вразливості та відмови СПЗ, як і раніше, існують і проявляються.

Розроблено класифікацію відмов та вразливостей СПЗ в залежності від їх прояву, математичні моделі відмови та вразливості СПЗ, а також критерії класифікації відмов та вразливостей СПЗ, які дають можливість сформулювати продукційні правила класифікації відмов та вразливостей СПЗ. Запропоновані авторами продукційні правила класифікації відмов та вразливостей СПЗ надають можливість полегшення процесу ідентифікації помилок у системному програмному забезпеченні.

Перспективним напрямком досліджень є побудова методу та алгоритму прогнозування відмов та вразливостей системного програмного забезпечення на основі розроблених математичних моделей відмови та вразливості, критеріїв класифікації та продукційних правил відмов і вразливостей СПЗ.

Використані джерела:

1. Рощин А.В. Системное программное обеспечение: Учебное пособие – М.: МГУПИ, 2007 – 166 с.
2. Журавлева Т.Ю. Системное и прикладное программное обеспечение: Учебное пособие – М.: Издательство Московского государственного открытого университета, 2010 – 144 с.
3. Tanenbaum A.S., Herder J.N., Bos H. Can we make operating systems reliable and secure? // Computer, Volume 39, Issue 5 - IEEE Computer Society Press Los Alamitos, CA, USA, 2006 – pp. 44-51 // [Electronic resource] – Access mode: http://www.cs.vu.nl/~ast/publications/computer-2006a.pdf
4. С.Макконнелл. Совершенный код. Мастер-класс - М.: Издательство "Русская редакция", 2013 - 896 с.
5. Ostrand T.J., Weyuker E.J. The Distribution of Faults in a Large Industrial Software System // Proceedings of International Symposium on Software Testing and Analysis – ACM Press, 2002 – pp. 55-64
6. A. Chou , J. Yang , B. Chelf , S. Hallem , D. Engler, An Empirical Study of Operating Systems Errors // Proceedings of the 18-th ACM Symposium on Operating Systems Principles - Banff, Alberta, Canada, 2001 – pp.73-88 // [Electronic resource] – Access mode: http://www.stanford.edu/~engler/metrics-sosp-01.pdf
7. Study: 359 Android code flaws pose security risks // [Electronic resource] – Access mode: http://news.cnet.com/8301-30685_3-20021437-264.html?part=rss&subj=news&tag=2547-1_3-0-20

8. M.M. Swift , B.N. Bershad , H.M. Levy. Improving the Reliability of Commodity Operating Systems // ACM Transactions on Computer Systems, v.23, n.1 – ACM Press, 2005, pp.77-110 // [Electronic resource] – Access mode: http://pages.cs.wisc.edu/~swift/papers/nooks-tocs.pdf
9. J. LeVasseur, V. Uhlig, J. Stoess, S. Götz. Unmodified Device Driver Reuse and Improved System Dependability via Virtual Machines // Proceedings of the 6-th Symposium on Opearting Systems Design & Implementation, vol.6 - San Francisco, CA, 2004 - p.2 // [Electronic resource] – Access mode: http://static.usenix.org/event/osdi04/tech/full_papers/levasseur/levasseur.pdf
10. J.N. Herder, H. Bos, B. Gras, Ph. Homburg, A.S. Tanenbaum. Modular System Programming in MINIX 3 // Usenix ;login:, vol.30, n.4 – Usenix, 2005 // [Electronic resource] – Access mode: https://c59951.ssl.cf2.rackcdn.com/786-herder_1.pdf
11. М.Ховард, Д.Лебланк, Дж.Вьега. 24 смертных греха компьютерной безопасности: Как написать безопасный код. Библиотека программиста – СПб: Питер, 2010 – 400 с.
12. McAfee DAT 5958 Update Issues // [Electronic resource] – Access mode: http://isc.sans.edu/diary/McAfee+DAT+5958+Update+Issues/8656
13. How to fix battery life issues with iOS 6 or iPhone 5 // [Electronic resource] – Access mode: http://www.imore.com/how-fix-battery-life-problems-ios-6-or-iphone-5
14. М.Ховард, Д.Лебланк, Дж.Вьега. Уязвимости в программном коде и борьба с ними – М.: ДМК Пресс, 2011 – 288 с.
15. Статистические данные: Уязвимости по степени опасности в 2013 году // [Електронний ресурс] – Режим доступу: http://www.securitylab.ru/vulnerability/
16. ГОСТ 27.002-89. Надежность в технике: Основные понятия. Термины и определения // [Електронний ресурс] – Режим доступу: http://www.estateline.ru/legislation/1213/
17. ГОСТ Р 51275-2006. Защита информации. Объект информатизации. Факторы, воздействующие на информацию. Общие положення // [Електронний ресурс] – Режим доступу: http://www.altell.ru/legislation/standards/51275-2006.pdf