УДК 004.891

д.т.н., проф. **Лєнков С.В.** (ВІКНУ)
д.т.н., проф. **Зубарєв В.В.** (ЦНДІ ОВТ)
к.т.н., доц. **Джулій В.М.** (ХмНУ)
к.т.н., доц. **Красильников С.Р.** (ХмНУ)

# АНАЛІЗ ЗАСОБІВ І МЕТОДІВ ПРОТИДІЇ АТАКАМ НА КОМП'ЮТЕРНІ СИСТЕМИ

***У статті проведено аналіз та запропоновано класифікацію існуючих вразливостей програмного забезпечення. Розглянуто існуючі на даний час методи і засоби протидії атакам. Проведений аналіз повідомлень, в яких йдеться про успішні хакерські атаки, дозволяє зробити висновок, що збільшується як кількість окремих атак, так і збиток, що наноситься кожною з них. Розглянуті випадки виявлення та експлуатації вразливостей в розповсюдженому по всьому світу програмних продуктах показують, що вони є серйозною загрозою безпеки комп'ютерних систем. Запропонована класифікацію досліджених вразливостей дозволяє усунути ряд вразливих функцій з вихідного коду і в результаті чого ймовірність, вразливості стає рівною нулю.***

***Ключові слова: інформаційна безпека, комп'ютерні системи, хакерські атаки, вразливість програмного коду.***

**Вступ**. В період бурхливого розвитку і впровадження у повсякденне життя нових комп'ютерних технологій провідну роль відіграють надійність систем доступу до інформації, і її зберігання. Останнім часом спостерігається тенденція до збільшення кількості та потужності комп'ютерних атак на інфраструктуру обчислювальних мереж, також постійно з'являється інформація про різні вірусні епідемії, які при поширенні генерують великі обсяги трафіку, внаслідок чого перевантажують канали зв'язку.

В даний час інформатизація проникла практично в усі сфери людської діяльності. Однак, приділяючи велику увагу новим функціональним можливостям засобів зберігання і обробки інформації, розробники часто випускають з уваги питання захищеності своїх програмних продуктів. Вразливість програмного коду, навмисно або частіше випадково залишена розробником, може стати причиною глобальних електронних «епідемій», що завдають відчутний фінансовий збиток, а за певних обставин здатна навіть призвести до знищення матеріальних об'єктів і загибелі людей. Поняття вразливості можна визначити як властивість комп'ютерної системи, наявність якої може дозволити зловмиснику завдати шкоди інтересам власника системи, інакше кажучи, привести до реалізації загрози інформаційної безпеки. Вразливість програмного коду - це особливість написання вихідного коду програми, в результаті наявності якої зловмисник, використовуючи спеціальним чином

підготовленні вхідні дані або інші параметри програмного оточення, може змусити вразливу програму працювати за алгоритмом, відмінним від закладеного програмістом.

**Постановка проблеми.** Наявні в даний час засоби протидії шкідливим програмам не завжди можуть впоратися з вирішенням свого завдання, головним чином тому, що націлені на виявлення і усунення наслідків функціонування вже відомих шкідливих програм і блокування попередньо знайдених вразливостей. Для підвищення захищеності додатків необхідно одночасно йти кількома шляхами: шукати універсальні ознаки груп вразливостей з метою їх ефективного виявлення, розробляти превентивні заходи протидії шкідливим програмам, а також розвивати системний підхід до навчання нових розробників, здатних вже на етапі написання програми виявляти і усувати в ній вразливості. Роботи, присвячені дослідженню вразливостей, містять перерахування існуючих проблем, але не дають рекомендацій зпідвищенню захищеності програмного забезпечення. Поділ вразливостей по класах могло б дати програмістам потужний інструмент виявлення вразливостей та запобігання їх використання. однак в даний час комплексних робіт по всіх видах вразливостей проведено не було.

Завдання підвищення захищеності додатків вирішується за рахунок використання сканерів вразливостей і безпечних компіляторів, які в автоматичному режимі допомагають виявити багато з існуючих вразливостей і зменшити ризик їх виникнення за рахунок заміни небезпечних функцій на їх безпечні аналоги. Однак використання таких інструментів можливо тільки при наявності вихідного коду додатка, що захищається, а крім того при цьому збільшується обсяг програми та знижується її продуктивність. Безпека додатка повинна закладатися на етапі написання вихідного коду. Для цього необхідна підготовка нових розробників, що володіють не тільки навичками програмування, а й добре знайомий з питаннями розробки захищеного програмного забезпечення. Про необхідність підготовки таких фахівців йдеться в цільовій програмі розвитку освіти, де ставиться завдання підготовки кадрів з пріоритетних напрямів, в числі яких виділяються стратегічні комп'ютерні технології і програмне забезпечення, серед головних завдань виділяється створення єдиної системи підготовки кадрів у галузі інформаційної безпеки та інформаційних технологій.

Ризиком у сфері інформаційної безпеки називатимемо потенційну можливість понести збитки через порушення безпеки інформаційної системи. Найчастіше поняття ризику співставляють з поняттям загрози. Загрозою інформаційної безпеки називають потенційно можливу подію і неважливо, навмисну чи ні, яка може надати небажану дію на комп'ютерну систему, а також інформацію, що зберігається і обробляється в ній.

Вразливість інформаційної системи - це деяка невдала характеристика, яка робить можливим виникнення загрози. Вразливість є недостатня захищеність і / або деякі помилки в системі, а також наявність в системі потайних входів в ній, залишені розробниками цієї системи при її відлагодженні і налаштуванні.

Від загрози ризик відрізняється наявністю кількісної оцінки можливих втрат і (можливо) оцінки ймовірності реалізації загрози.

Для будь-якого проекту, що вимагає фінансових витрат на його реалізацію, дуже бажано вже на початковій стадії визначити, що ми будемо вважати ознакою завершення роботи і як будемо оцінювати результати проекту. Для задач, пов'язаних із забезпеченням інформаційної безпеки це більш ніж актуально. На практиці найбільше поширення одержали два підходи до обгрунтування проекту підсистеми забезпечення безпеки.

Перший з них заснований на перевірці відповідності рівня захищеності інформаційної системи вимогам одного з стандартів в області інформаційної безпеки. Це може бути клас захищеності відповідно до вимог керівних документів, профіль захисту, розроблений у відповідності зі стандартом ISO-15408, або який-небудь інший набір вимог. Тоді критерій досягнення мети в області безпеки - це виконання заданого набору вимог. Критерій ефективності - мінімальні сумарні витрати на виконання поставлених функціональних вимог: $\sum C_i \to \min$ де $C_i$ - витрати на і-й засіб захисту. Основний недолік цього підходу полягає в тому, що у випадку, коли необхідний рівень захищеності жорстко не заданий (наприклад,

через законодавчі вимоги) визначити "найбільш ефективний" рівень захищеності інформаційної системи (ІС) досить складно.

Другий підхід до побудови системи забезпечення інформаційної безпеки пов'язаний з оцінкою та управлінням ризиками. Спочатку він виник з принципу "розумної достатності" застосованого до сфери забезпечення інформаційної безпеки. Цей принцип може бути описаний таким набором тверджень:

- абсолютно неперебoрну систему захисту створити неможливо;
- необхідно дотримувати баланс між витратами на захист і отримуваним ефектом, в т.ч. і економічним, що полягає в зниженні втрат від порушень безпеки;
- вартість засобів захисту не повинна перевищувати вартості інформації, що захищається (або інших ресурсів - апаратних, програмних);
- витрати порушника на несанкціонований доступ (НСД) до інформації повинні перевищувати той ефект, який він отримає, здійснивши подібний доступ.

Але повернемося до ризиків. В даному випадку, розглядаючи інформаційну систему в її початковому стані, ми оцінюємо розмір очікуваних втрат від інцидентів, пов'язаних з інформаційною безпекою (як правило, береться певний період часу, наприклад - рік). Після цього, робиться оцінка того, як пропоновані засоби і заходи забезпечення безпеки впливають на зниження ризиків, і скільки вони коштують. Якщо представити деяку ідеальну ситуацію, то ідею підходу відображає наведений нижче графік (рис. 1).

У міру того, як витрати на захист ростуть, розмір очікуваних втрат падає, якщо обидві функції мають вигляд, представлений на рис. 1., то можна визначити мінімум функції "Очікувані сумарні витрати", який необхідний. На жаль, на практиці точні залежності між витратами і рівнем захищеності визначити не представляється можливим, тому аналітичний метод визначення мінімальних витрат в представленому вигляді непридатний.

Для того, щоб перейти до розгляду питань опису ризику, введемо ще одне визначення. Ресурсом або активом називатимемо іменований елемент інформаційної системи, що має (матеріальну) цінність і підлягає захисту. Тоді ризик може бути ідентифікований наступним набором параметрів:

- загроза, можливою реалізацією якої викликаний даний ризик;
- ресурс, відносно якого може бути реалізована дана загроза (ресурс може бути інформаційний, апаратний, програмний і т.д.);
- вразливість, через яку може бути реалізована дана загроза відносно даного ресурсу.

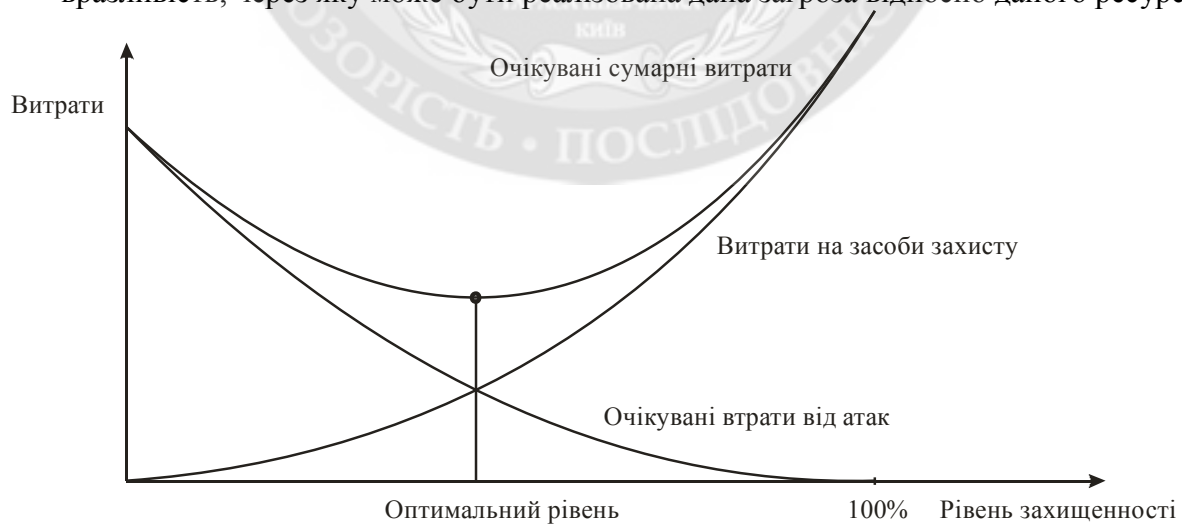


Рис. 1. Ідеалізований графік співвідношення "витрати на захист - очікувані втрати"

Важливо також визначити те, як дізнатися, що небажана подія сталася. Тому в процесі опису ризиків, зазвичай також вказують події-"тригери", що є ідентифікаторами ризиків, що сталися або очікуються незабаром (наприклад, збільшення часу відгуку web-сервера може

свідчити про виробленої на нього одного з різновидів атак на "відмова в обслуговуванні" ). Виходячи зі сказаного вище, в процесі оцінки ризику треба оцінити вартість збитку і частоту виникнення небажаних подій і ймовірність того, що подібна подія завдасть шкоди ресурсу.

**Виклад основного матеріалу досліджень.** Проведений аналіз повідомлень, в яких йдеться про успішні хакерські атаки або випущених програмних «латках» дозволяє зробити висновок, що збільшується як кількість окремих атак, так і збитка, що наноситься кожною з них. Так, відносно недавно заявив про собе мережевий хробак Stuxnet, функціонування якого призвело до фізичного руйнування обладнання, задіяного в іранській ядерній програмі. Якщо до Stuxnet можна було вважати, що розробкою вірусів і інших руйнуючих програмних впливів займалися, в основному, не професіонали, то цей хробак став без перебільшення справжньою кіберзброєю, що поклав початок новому етапу в розвитку руйнуючих програмних впливів. Розглянуті випадки виявлення та експлуатації вразливостей в розповсюджених по всьому світу програмних продуктах показують, що вони є серйозною загрозою безпеки комп'ютерних систем і в певному сенсі є перешкодою до розвитку суспільства в напрямку прискорення обміну інформацією та загальної комп'ютеризації. Хоча останнім часом і були впроваджені спеціальні підрозділи що займаються пошуком вразливостей, а також додаткові етапи при розробці програмного продукту, спрямовані на виявлення та усунення вразливостей, тим не менше, для кожної знайденої вразливості кожна фірма-розробник шукає свій спосіб протидії, в результаті заходи виявляються недостатньо ефективними або призводять навіть до породження нових вразливостей. Такий підхід не дозволяє виробити загальні методи усунення не окремих вразливостей, а їх класів, в внаслідок чого вразливості залишаються навіть у продуктах широко відомих фірм - розробників програмного забезпечення. Час, що проходить зазвичай між моментом виявлення вразливостей і моментом її закриття, може становити до півроку, тому завдання написання коду, спочатку позбавленого вразливостей, дуже актуальне. У зв'язку з цим необхідна система, яка дозволила б виявляти вразливості за певними ознаками і гарантовано їх закривати. В основу системи пошуку вразливостей має бути покладена класифікація вразливостей з чітко визначеними ознаками, що характеризують вразливий код. В даний час такої класифікації не запропоновано, тому кожен виробник намагається шукати вразливості за власними методиками, приділяючи більшу або меншу увагу окремим вразливостям відповідно до власних вподобань.

В даний час існує досить багато програмно-апаратних підходів до пошуку та усунення вразливостей, проте вони не дозволяють виявити і виправити всі вразливості, вимагають знання коду і високої кваліфікації програміста, що працює з результатами аналізу, а крім того, багато з пропонованих методів носять декларативний характер і не доведені до практичного втілення.

Для зниження числа успішних атак на програмні системи або зменшення їх наслідків використовуються різні програмно-апаратні засоби. До програмних засобів відносяться статичні сканери вихідного коду, динамічні аналізатори вразливих додатків, надбудови до компіляторів (або безпечні діалекти мов програмування), а також надбудови до ядра. Статичні аналізатори вихідного коду проводять синтаксичний аналіз коду програми, виявляють небезпечні функції, про що інформують програміста. До статичних сканерів відносяться такі додатки, як RATS, Flawfinder, Splint, ITS4 та ін. Недоліками сканерів є високі вимоги до кваліфікації програміста при виправленні знайдених вразливих місць, а також тривалий час роботи, тому процес ітераційний. Динамічні аналізатори працюють шляхом багаторазового запуску тестованого додатка з автоматично формованими вхідними параметрами, які вибираються за принципом близькості до граничних значень (велика довжина, максимально можливе ціле і т.п.). Така технологія отримала назву Fuzzing і частіше застосовується для web-додатків. До динамічних аналізаторів відносяться програми Sharefuzz, OWASP JBroFuzz, Bunny the Fuzzer та ін.. Для підвищення захищеності програм використовуються надбудови до компіляторів. При компіляції програми небезпечні функції можуть замінюватися на безпечні «обгортки», які перевіряють коректність вхідних

параметрів, можуть використовуватися «сторожові байти» для виявлення факту запису за межі виділеної пам'яті, може бути використано перемішування даних вразливою функцією для більш безпечного їх розміщення. Найбільш поширеними надбудовами до компіляторів є додатки Stackguard, ProPolice, StackShield, PointGuard, FormatGuard, Libformat та ін Недоліками в роботі надбудов до компіляторів є вимога доступності вихідного коду для безпечної компіляції, зниження швидкодії програми, а також вимоги щодо розстановки спеціальних символів в програмі для можливості перевірки компілятором задуму розробника. Надбудови до ядра (наприклад, Raceguard, Systrace, Janus) дозволяють здійснювати моніторинг дій недовірених процесів і при необхідності припиняти їх. Істотними їх недоліками є труднощі адміністрування системи і низька сумісність різних виконуючих процесів. Існують також апаратні методи підвищення захищеності працюючої системи. Серед інших вузькоспеціалізованих методів обмеження та контролю в них пропонується використання рандомізації в окремих елементах виконання процесу. Під рандомізацією у даному випадку розуміється внесення невизначеності порівняно з класичною архітектурою, в якій даний параметр або заздалегідь визначений, або розраховується на підставі відомих атрибутів і властивостей системи. Основним недоліком таких підходів є протидія далеко не всім видам атак, заснованими на використанні вразливостей програмного забезпечення (в першу чергу, переповнення буфера), а крім того багато з них не дозволяють використовувати спільні бібліотеки.

ER-модель процесу експлуатації вразливостей показано на рис. 2.

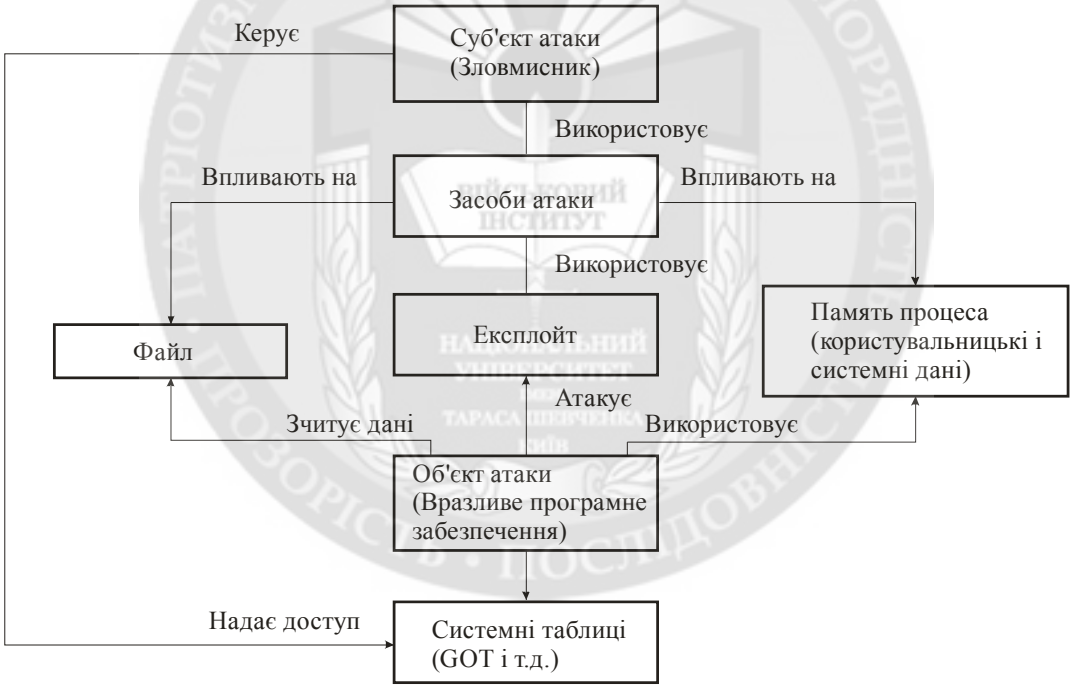


Рис. 2. ER-модель процесу експлуатації вразливостей

На підставі інформації про причини виникнення вразливостей, що зустрічаються в програмах, була складена їх класифікація за наступними параметрами (рис. 3 – 5): сутністю, що модифікується; наслідками атаки з використанням вразливостей, розміщенням шкідливої сутності.

Систематизація вразливостей дозволяє:

1) з високою достовірністю виявляти їх у вихідних кодах;

2) розробити методи безпечного програмування, виключаючи можливість появи вразливостей в програмах;

3) визначити механізми використання вразливостей, що дозволить боротися з ними у виконуваній програмі за відсутності вихідного коду.

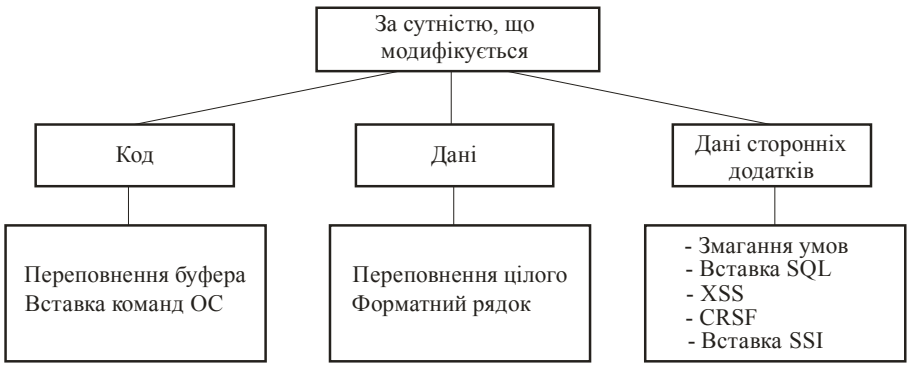


Рис. 3. Класифікація вразливостей за сутністю, що модифікується


Рис. 4. Класифікація вразливостей за наслідками атаки з використанням вразливостей

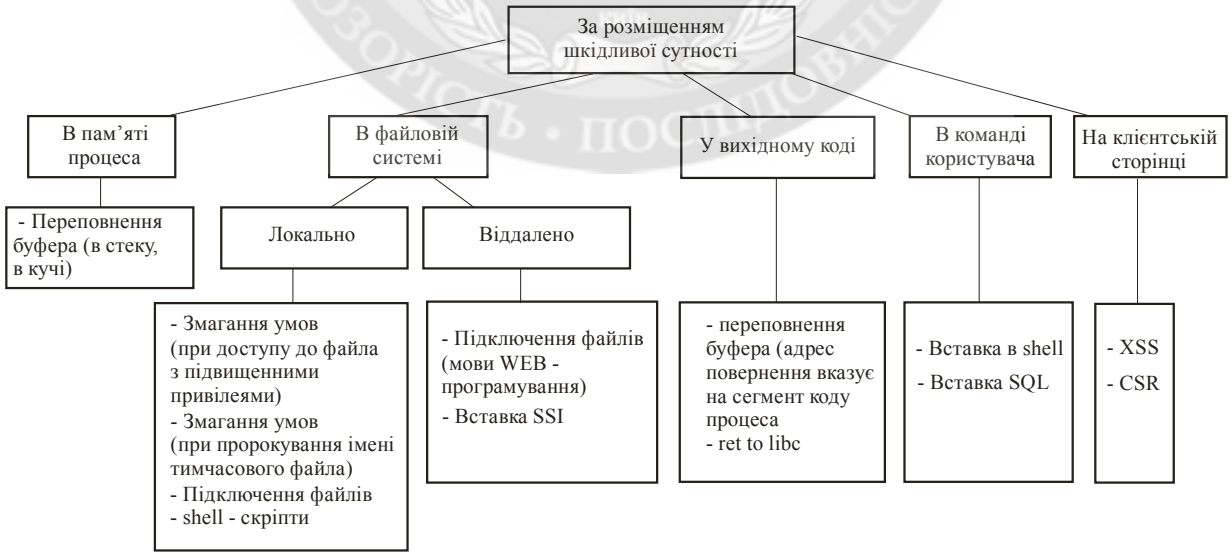


Рис. 5. Класифікація вразливостей за розміщенням шкідливої сутності

Якщо представити множину всіх вразливих функцій як $F=\{f_1, f_2, ..., f_n\}$, а всіх вразливостей, до яких може призвести використовування цих функцій як $V=\{v_1, v_2, ..., v_m\}$, то визначивши множину ймовірностей, з якими функція $f_i$ може призвести до появи вразливості $v_j$ як $P=\{p_i^j\}$ (i = 1, ..., n, j = 1, ..., m), можна визначити рівень небезпеки тієї чи іншої програми як $D=\sum_{i=1}^{n}\left(K_i \sum_{j=1}^{m} p_i^j r_j\right)$, де $K_i$ - число функцій *i*-го виду в програмі, а $r_j$ - небезпека *j*-ї вразливості в балах, визначена методом експертних оцінок.

**Висновки.** Запропонований підхід дозволяє усунути ряд вразливих функцій з вихідного коду, в результаті ймовірність появи вразливості стає рівною нулю. Тоді підвищення захищеності програми можна оцінити як відношення рівня небезпеки програми до ( D ) і після ( D ') застосування безпечного програмування.

Систематизація вразливостей дозволяє:

1) з високою достовірністю виявляти їх у вихідних кодах;

2) розробити методи безпечного програмування, виключаючи можливість появи вразливостей в програмах;

3) визначити механізми використання вразливостей, що дозволить боротися з ними у виконуваній програмі за відсутності вихідного коду.

ЛІТЕРАТУРА:

1. Духан Е.И. Применение программно-аппаратных средств защиты компьютерной информации / Е.И. Духан, Н.И. Синадский, Д.А. Хорьков // Федеральное агентство по образованию. – 2008. – С. 183

2. Ленков С.В.Методы и средства защиты информации /Ленков С.В. – К.:Арий, 2008. – 652 с.

3. Цирлов В.Л. Основы информационной безопасности автоматизированных систем / В.Л. Цирлов // Издание «Феникс». – 2008. – С. 473.

4. Щеглов А. Ю. Защита компьютерной информации от несанкцио-нированного доступа / А. Ю. Щеглов // Наука и Техника. – 2004. – С. 384с.

5. Згуровский М.З., Системный анализ (проблемы, методология, приложения) / М.З. Згуровский, Н.Д. Панкратова // – К.: Наук. думка, 2005. – 744 с.

**Без рецензії.**

**д.т.н., проф. Ленков С.В., д.т.н., проф. Зубарев В.В.,**
**к.т.н., доц. Джулий В.М., к.т.н., доц. Красильников С.Р.**

## АНАЛИЗ СРЕДСТВ И МЕТОДОВ ПРОТИВОДЕЙСТВИЯ АТАКАМ НА КОМПЬЮТЕРНЫЕ СИСТЕМЫ

***В статье проведен анализ и предложена классификация существующих уязвимостей программного обеспечения. Рассмотрены существующие в настоящее время методы и средства противодействия атакам.***

***Проведенный анализ сообщений, в которых говорится об успешных хакерских атаках, позволяет сделать вывод, что увеличивается как количество отдельных атак, так и ущерб, наносимый каждой из них. Рассмотренные случаи обнаружения и эксплуатации уязвимостей в распространенном по всему миру программных продуктах показывают, что они являются серьезной угрозой безопасности компьютерных систем. Предложенная классификация исследованных уязвимостей позволяет устранить ряд уязвимых функций из исходного кода и в результате чего вероятность, уязвимости становится равной нулю.***

***Ключевые слова: информационная безопасность, компьютерные системы, хакерские атаки, уязвимости программного кода.***

**Lenkov S., Zubarev V., Julie V., Krasilnikov S.**

# ANALYSIS TOOLS AND METHODS TO COUNTERACT ATTACKS ON COMPUTER SYSTEMS

***This paper analyzes and classification of existing software vulnerabilities. The existing at present the methods and means of countering attacks.***

***The analysis reports, which describe the successful hacking attack, allowing***

***conclude that increasing the number of individual attacks and damage to each of them. The cases detection and exploitation of vulnerabilities in distributed worldwide software products indicate that they are a serious threat to the security of computer systems. The classification of the studied vulnerabilities can eliminate a number of vulnerable functions from the source code and the resulting probability, vulnerability becomes zero.***

***Keywords: information security, computer systems, cyber attacks, vulnerability code.***