МЕТОДИЧНІ ВКАЗІВКИ
ДО ВИКОНАННЯ КУРСОВОЇ РОБОТИ
З ДИСЦИПЛІНИ
«ЗАХИСТ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ»

для студентів напряму підготовки

6.170103 «Управління інформаційною безпекою»

Навчальне видання

МЕТОДИЧНІ ВКАЗІВКИ
ДО ВИКОНАННЯ КУРСОВОЇ РОБОТИ
З ДИСЦИПЛІНИ
«ЗАХИСТ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ»
для студентів напряму підготовки
6.170103 «Управління інформаційною безпекою»

Редактор В. Дружиніна
Коректор З. Поліщук

Укладачі: Карпінець Василь Васильович
Кец Дмитро Олександрович

Оригінал-макет підготовлено В. Карпінець

Підписано до друку 13.05.2017 р.
Формат 29,7×42 ¼. Папір офсетний.
Гарнітура Times New Roman.
Ум. друк. арк. 2,13.
Наклад 40 пр. Зам. № 2017-127.

Видавець та виготовлювач
Вінницький національний технічний університет,
інформаційний редакційно-видавничий центр.

ВНТУ, ГНК, к. 114.
Хмельницьке шосе, 95, м. Вінниця, 21021.
Тел. (0432) 59-85-32, 59-87-38.
press.vntu.edu.ua; *e-mail:* kivc.vntu@gmail.com
Свідоцтво суб'єкта видавничої справи
серія ДК № 3516 від 01.07.2009 р.

Міністерство освіти і науки України
Вінницький національний технічний університет

МЕТОДИЧНІ ВКАЗІВКИ ДО ВИКОНАННЯ КУРСОВОЇ РОБОТИ З ДИСЦИПЛІНИ «ЗАХИСТ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ»

для студентів напряму підготовки
6.170103 «Управління інформаційною безпекою»

Вінниця
ВНТУ
2017

Рекомендовано до друку Методичною радою Вінницького національного технічного університету Міністерства освіти і науки України (протокол № 6 від 18.02.2016 р.)

Рецензенти:
С. М. Захарченко, кандидат технічних наук, доцент
Д. І. Катєльніков, кандидат технічних наук, доцент
Ю. В. Булига, кандидат технічних наук, доцент

Методичні вказівки до виконання курсової роботи з дисципліни «Захист програмного забезпечення» для студентів напряму підготовки 6.170103 «Управління інформаційною безпекою» / Уклад.: В. В. Карпінець, Д. О. Кец. – Вінниця : ВНТУ, 2017. – 37 с.

У даних методичних вказівках наводяться основні рекомендації до виконання та оформлення курсової роботи з дисципліни «Захист програмного забезпечення».

ЗМІСТ

1 ОСНОВНІ ВИМОГИ ДО КУРСОВОЇ РОБОТИ

1.1 Мета і задачі курсової роботи

Курсова робота (КР) – навчальна самостійна робота з дисципліни, яка містить елементи (задачі) навчального, аналітично-розрахункового та науково-дослідницького характеру.

В курсовій роботі з дисципліни «Захист програмного забезпечення» студент повинен показати свої знання в галузі захисту програмних продуктів від нелегального копіювання і використання програм, несанкціонованого використання і дослідження, знання нових технологій захисту, вміння самостійно розробити систему захисту від атак певного виду, підібрати засоби для його реалізації та вміння показати розробку у вигляді, зручному для його використання стороннім користувачем.

Під час виконання курсової роботи студенти повинні використати всі знання, отримані ними під час вивчення дисциплін: «Технології програмування», «Захист операційних систем», «Основи інформаційної безпеки», «Основи криптографічного захисту інформації».

Під час виконання курсової роботи студенти повинні вміти:

– коректно сформулювати задачу для розв’язання її на певному типі обчислювальних машин, у певному операційному середовищі;

– правильно визначити механізми захисту (вбудовані або навісні) та виконувані функції;

– визначити конкретні методи захисту для можливості їх застосування відповідно до умов використання програмного забезпечення, які придатні для конкретного типу задач (захист інформації від нелегального копіювання, від несанкціонованого вивчення або дослідження програм);

– звести постановку задачі до розробки алгоритму і визначити структуру даних, яка дозволяє перейти від абстрактного формулювання алгоритму до конструювання структурної схеми;

– здійснювати програмну реалізацію винайденого типу захисту, використовуючи мови високого рівня;

– довести доцільність використання розробленого захисту, визначити коло програм і задач, для яких даний метод може бути застосований;

– довести ефективність вибраного способу захисту, використовуючи при цьому дизасемблери та налагоджувачі;

– подати власну розробку таким чином, щоб нею могли користуватись інші, підготувавши для цього відповідний методичний матеріал у вигляді детальних інструкцій та рекомендацій.

1.2 Тематика курсової роботи

Зміст курсової роботи відповідає навчальній програмі та робочому плану дисципліни «Захист програмного забезпечення» і повинен відображати суть обраної студентом теми. Зміст КР визначається завданням, яке видається не пізніше шести днів з початку семестру на консультації викладачем кожному студенту.

Виконання курсової роботи містить декілька послідовних етапів, які повинні бути пов'язані зі змістовною постановкою задачі, розробкою індивідуального та технічного завдання, варіантним аналізом методів захисту, вибором форми подання задачі, математичною моделлю розв'язання, вибором оптимального алгоритму для реалізації захисту, проведенням досліджень створеної програми та формулюванням обґрунтованих висновків щодо ефективності розробленого захисту. Орієнтовний графік виконання курсової роботи наведено у подальших розділах. Студентам бажано дотримуватись наведеного графіка, починаючи роботу над проектом з перших тижнів триместру. Під час проведення поточного контролю знань з дисципліни «Захист програмного забезпечення» якість, зацікавленість та ефективність роботи студентів буде оцінюватись відповідною кількістю балів, що суттєво впливатимуть на загальну оцінку з дисципліни за модуль.

Тематика курсової роботи повинна бути пов'язана з майбутньою спеціальністю студентів. Для програмної реалізації даного курсової роботи пропонується ряд основних методів захисту програм від копіювання, від статичного та динамічного дослідження.

Крім того, об'єктом для курсової роботи можуть бути дослідження та впровадження відомих систем захисту, реалізація тестових програм, розробка лабораторних практикумів для даної дисципліни та для інших дисциплін, пов'язаних із захистом програмного забезпечення тощо.

Індивідуальне завдання для курсових робіт визначається викладачем із загального списку завдань на курсову роботу (Додаток А). Заохочуються пропозиції студентів щодо самостійного, за узгодженням з викладачем, вибору теми курсової роботи поза межами запропонованого в методичних вказівках переліку.

Самостійний вибір предметної області, в якій доцільно використовувати сучасні методи захисту програм та оригінальні алгоритми, дозволяє зробити висновок щодо рівня творчої активності студента, його вміння самостійно здійснювати попередній аналіз предметної області, поставити перед собою конкретну задачу та ефективно її реалізувати.

2 КОРОТКІ ТЕОРЕТИЧНІ ВІДОМОСТІ

2.1 Мета і доцільність використання систем захисту

Перед початком проектування будь-якої системи захисту (СЗ) програмного забезпечення (ПЗ) треба цілком чітко собі уявляти, що саме і від якої загрози потрібно «захищати».

Системи захисту програмного забезпечення на сьогодні є досить поширеними і знаходяться у постійному розвитку завдяки збільшенню ринку ПЗ і телекомунікаційних технологій. Необхідність використання систем захисту програм обумовлена рядом проблем, серед яких варто виділити такі:

– промислове шпигунство – незаконне використання алгоритмів, що є інтелектуальною власністю автора, при написанні аналогів продукту;

– крадіжка і копіювання – несанкціоноване використання програмного забезпечення;

– несанкціонована модифікація програмного забезпечення з метою впровадження програмних зловживань;

– піратство – незаконне поширення і збут програмного забезпечення.

Не існує абсолютно надійних методів захисту. Кваліфіковані системні розробники, що користуються побітовими копіювальниками або сучасними засобами аналізу програмного забезпечення (налагоджувачі, дизасемблери, шістнадцяткові редактори, перехоплювачі переривань і т. п.), витративши певний час, зможуть подолати практично будь-який захист. І тому при проектуванні систем захисту слід передбачати те, що рано чи пізно цей захист буде знято.

Метою проектування повинен бути вибір такого методу захисту, що забезпечить неможливість несанкціонованого копіювання для заздалегідь визначеного кола осіб на обмежений час. Наприклад, якщо потрібно захистити від копіювання комерційну версію програми, не обов’язково захищати цю версію «назавжди», оскільки вартість такого захисту перевищуватиме вартість самої програми. Цілком досить того, щоб спосіб захисту неможливо було «розгадати» до моменту появи наступної версії програми, а в новій версії змінити спосіб захисту.

Таким чином, для вибору способу організації захисту необхідним є індивідуальний підхід у кожному конкретному випадку.

Студент, вибравши собі завдання для курсової роботи, повинен чітко уявляти собі, що, як і від кого він збирається захищати, тобто визначитись з метою захисту, об’єктом захисту, механізмом, принципом і методом захисту.

Існуючі системи захисту програмного забезпечення можна класифікувати за рядом ознак, серед яких можна виділити:

– метод установлення;

– принцип функціонування.

2.2 Методи установлення захисних механізмів

За методом установлення системи захисту програмного забезпечення можна розділити на такі.

Навісний захист (або системи, що встановлюються на скомпільовані модулі ПЗ) полягає у вставленні фрагмента перевірного коду у виконуваний файл. Можливість вбудовування захисних фрагментів у виконуваний код обумовлена типовою архітектурою виконуваних модулів різних операційних середовищ, що містять, як правило, адресу точки входу у виконуваний модуль. У цьому випадку додавання захисного фрагмента відбувається таким чином. Захисний фрагмент додається до початку або кінця виконуваного файлу, точка входу корегується так, щоб при завантаженні управління передалося додатковому захисному фрагменту, а у складі захисного фрагмента передбачається процедура повернення до оригінальної точки входу. Досить часто оригінальний виконуваний файл піддається перетворенню. У цьому випадку перед поверненням управління оригінальній точці входу здійснюється перетворення образу оперативної пам’яті завантаженого виконуваного файлу до початкового вигляду.

Такі системи найбільш зручні для виробника програмного забезпечення, оскільки легко можна захистити вже цілком готові програми (зазвичай процес установлення захисту максимально автоматизований і зводиться до вказання імені файлу, що захищається, і натискання клавіші «Enter»), а тому і найбільш популярні. У той же час стійкість цих систем досить низька (у залежності від принципу дії СЗ), оскільки для обходу захисту досить визначити точку завершення роботи «конверта» захисту і передачі керування захищеній програмі, а потім примусово її зберегти в незахищеному вигляді.

Вбудований захист (або системи, що вбудовуються у початковий код ПЗ до компіляції) полягає у вставленні перевірного механізму в початковий код на етапі розробки і налагодження програмного продукту. Системи цього типу незручні для розробника ПЗ, оскільки виникає необхідність навчати персонал роботі з прикладним програмним інтерфейсом (ППІ) системи захисту, внаслідок чого збільшуються грошові і часові витрати. Крім того, ускладнюється процес тестування ПЗ і знижується його надійність, оскільки крім самого ПЗ помилки може містити ППІ системи захисту або процедури, що його використовують. Але такі системи є більш стійкими до атак, оскільки тут зникає чітка границя між системою захисту і ПЗ як таким.

Захист шляхом модифікації і перетворення виконуваного файлу до невиконуваного вигляду (шифрування, архівація з невідомим параметром і т. д.) і застосуванням для завантаження не засобів операційного

середовища, а деякої програми, в тілі якої і здійснюються необхідні перевірки.

Комбінований захист – це найбільш стійкі системи захисту. Зберігаючи переваги систем першого або другого типів і недоліки систем третього типу, вони максимально ускладнюють аналіз і дезактивацію своїх алгоритмів.

2.3 Принципи функціонування систем захисту програм

За принципом функціонування системи захисту можна розділити на:

– пакувальники/шифрувальники;

– системи захисту програмного забезпечення (СЗПЗ) від несанкціонованого копіювання (НСК);

– СЗПЗ від несанкціонованого доступу (НСД);

– СЗ від несанкціонованого статичного дослідження та модифікації;

– СЗ від несанкціонованого динамічного дослідження;

– СЗ від розпакування та дампінгу процесів.

2.3.1 Пакувальники/шифрувальники

Пакувальники/шифрувальники на початку їх існування мали на меті зменшення на диску обсягу модуля, що виконується, без нанесення шкоди для функціональності програми, але пізніше на перший план вийшла мета захисту ПЗ від аналізу його алгоритмів та несанкціонованої модифікації. Для досягнення цього вони використовують:

– алгоритми компресії даних;

– прийоми, пов’язані з використанням недокументованих особливостей операційних систем (ОС) і процесорів;

– шифрування даних, алгоритми мутації, заплутування логіки програми, приведення ОС у нестабільний стан на час роботи програми і ін.

Не зважаючи на те, що методи пакування і шифрування на сучасному етапі розвитку способів захисту комп’ютерної інформації є досить актуальними, вони не завжди зручні у використанні, оскільки для виконання захищеної програми необхідно спочатку виконати її розпакування або розшифрування, а отже, деякий час програма все одно перебуває у незахищеному вигляді, чим можуть скористатися злочинці.

2.3.2 Системи захисту від несанкціонованого копіювання

Системи захисту від несанкціонованого копіювання здійснюють, як правило, прив’язку ПЗ до дистрибутивного носія (CD, DVD, електронного ключа). Даний тип захистів ґрунтується на глибокому вивченні роботи контролерів накопичувачів, їх фізичних показників, нестандартних режимів розбиття при форматуванні, зчитуванні/записі і т. п. При цьому на

фізичному рівні створюється дистрибутивний носій, що, наприклад, має неповторні властивості (зазвичай це досягається за допомогою нестандартної розмітки носія інформації і запису на нього додаткової інформації, пароля або мітки), а на програмному рівні створюється модуль (контролююча частина програми), налаштований на ідентифікацію й автентифікацію носія за його унікальними властивостями. При цьому також можливе застосування прийомів, що їх використовують пакувальники/шифратори.

Методи захисту від несанкціонованого копіювання не потребують великих фінансових витрат при впровадженні, однак мають низьку стійкість до зламу. Внаслідок цього застосування такого захисту виправдано тільки для програмного забезпечення нижньої цінової категорії. Для подібних програм важливі популярність і великі тиражі (іноді і за рахунок піратських копій). Використання більш стійкої і дорогої системи захисту в даному випадку не буде мати сенсу. Серед методів захисту, пов'язаних з прив'язкою до дистрибутивних носіїв програм, можна виділити такі:

– використання всередині захищуваної програми нестандартних параметрів форматування (використання додаткових доріжок, «зайві сектори», нестандартні номери і розміри секторів, нестандартний Interleave, нестандартні заголовки секторів і таке інше);

– запис критичної інформації в проміжках;

– перевищення об'єму доріжки і його контроль;

– використання секторів з неправильною контрольною сумою (Bad CRC);

– вимірювання і перевірка довжини доріжки;

– використання фізичних дефектів на носії інформації;

– застосування технології «ослаблених бітів» (запис деякої ділянки дистрибутивного носія з невизначеним рівнем сигналу);

– пошкоджені або відсутні адресні маркери;

– різношвидкісні доріжки (збій синхронізації);

– зміна параметрів дискових накопичувачів (наприклад, зміна швидкості обертання диска).

Незважаючи на те, що захист від копіювання не завжди надійно захищає програми від нелегального розповсюдження, дані методи також мають право на існування, оскільки вони можуть досить ефективно використовуватись у комбінації з іншими методами захисту.

2.3.3 Системи захисту від несанкціонованого доступу

Системи захисту від несанкціонованого доступу здійснюють попередню чи періодичну аутентифікацію користувача ПЗ або його комп'ютерної системи шляхом опитування додаткової інформації. До цього типу СЗ можна віднести такі.

1. Системи парольного захисту ПЗ. Цей клас СЗПЗ на сьогоднішній день є найпоширенішим. Основний принцип роботи даних систем полягає в ідентифікації та аутентифікації користувача ПЗ шляхом запиту додаткових даних (це можуть бути назва фірми і/або ім'я і прізвище користувача, його пароль або реєстраційний код). Ця інформація може запитуватися в різних ситуаціях, наприклад, при старті програми, після закінчення терміну безкоштовного використання ПЗ, під час виклику процедури реєстрації, в процесі встановлення на ПК користувача. Процедури парольного захисту прості в реалізації і, тому вони дуже часто застосовуються виробниками ПЗ. Більшість парольних СЗПЗ використовують логічні механізми, що зводяться до перевірки правильності пароля (або реєстраційного коду) і запуску або незапуску ПЗ, у залежності від результатів перевірки.

Існують також системи, які шифрують ПЗ, що захищається, і використовують пароль або похідну від нього величину як ключ дешифрації. Більшість таких систем використовує слабкі або найпростіші алгоритми шифрування.

2. Системи прив'язки ПЗ до комп'ютера користувача. Системи цього типу при встановленні ПЗ на персональний комп'ютер користувача здійснюють пошук унікальних ознак комп'ютерної системи або ці унікальні ознаки встановлюються самою системою захисту. Після чого цей модуль захисту в самому ПЗ налаштовується на пошук і ідентифікацію даних ознак, за якими надалі визначається авторизоване чи неавторизоване використання ПЗ. При цьому можливе застосування методик оцінювання швидкісних і інших показників процесора, материнської плати, додаткових пристроїв, ОС, зчитування/запис у мікросхеми енергонезалежної пам'яті, запис прихованих файлів, налаштування на найбільш використовувану карту ОЗП і т. п.

3. Системи з ключовими дисками. В даний момент цей тип систем захисту малорозповсюджений через його моральне старіння. СЗПЗ цього типу багато в чому аналогічні системам з електронними ключами, але тут критична інформація зберігається на спеціальному ключовому носії. Так само багато спільного є і з системами захисту від копіювання, оскільки використовуються одні й ті ж методи роботи з ключовим носієм.

Основною загрозою для таких СЗПЗ є перехоплення зчитування критичної інформації, а також незаконне копіювання ключового носія. Але, використовуючи комбінування методів захисту, даний вид захисту можна зробити більш стійким.

4. Апаратно-програмні системи захисту з електронними ключами. Цей клас СЗПЗ останнім часом здобуває все більшу популярність серед виробників програмного забезпечення. Під програмно-апаратними засобами захисту у даному випадку маємо на увазі засоби, основані на використанні так званих апаратних (електронних) ключів. Електронний

ключ – це апаратна частина системи захисту, що являє собою плату з мікросхемами пам'яті і, у деяких випадках, мікропроцесором, розміщену в корпусі і призначену для установлення в один зі стандартних портів ПК (COM, LPT, USB ...) чи слот розширення материнської плати. Також, як такий пристрій, можуть використовуватися СМАРТ-карти. За результатами проведеного аналізу, програмно-апаратні засоби захисту в даний момент є одними із найстійкіших систем захисту ПЗ від НСД. Електронні ключі за архітектурою можна розділити на ключі з пам'яттю (без мікропроцесора) і ключі з мікропроцесором (і пам'яттю). Зазвичай більш стійкими є системи захисту з апаратною частиною другого типу. Такі комплекси містять в апаратній частині не тільки ключ дешифрації, але і блоки шифрації/дешифрації даних.

У першому випадку ключову інформацію вводить користувач, у другому – вона міститься в унікальних параметрах комп'ютерної системи користувача, у третьому – вона зберігається на диску, а у четвертому випадку ключова інформація зчитується з мікросхем електронного ключа.

2.3.4 Захист від статичного дослідження

Цілком зрозуміле бажання більшості програмістів працювати з твердою копією досліджуваної програми. Відсутність початкових текстів зовсім не є непереборною перешкодою для вивчення і модифікації коду додатка. Методики зворотного проектування дозволяють автоматично розпізнавати бібліотечні функції, локальні змінні, стекові аргументи, типи даних, цикли і т. д. Для цього зламниками використовуються такі інструменти, як декомпілятори, дизасемблери, шістнадцяткові редактори, редактори ресурсів. Першочерговою задачею зловмисника при зламі практично будь-якого захисту є дизасемблювання коду програми, що виконується, і одержання лістинга з мнемонічним зображенням асемблерних команд, тобто можливість статичного дослідження програми. Дизасемблер відновлює код програми, послідовно декодуючи команду за командою з того місця, куди програмі передається керування при її запуску. Він намагається додержуватися порядку виконання інструкцій, відрізняючи їх у такий спосіб від даних.

Будь-який, навіть найвитонченіший захист від копіювання, що забезпечує майже 100% гарантію легальності копії, є марним, якщо код програмного продукту доступний для вивчення й аналізу. У ньому завжди знайдуться місця, злегка змінивши які, можна якщо не цілком відключити захист, то, принаймні, прив'язати його ще до декількох комп'ютерів.

Можна виділити декілька найзагальніших підходів до захисту ПЗ від дизасемблювання.

1. Шифрування коду – один з найпоширеніших та надійних методів захисту від статичного дослідження.

2. Маніпулювання із заголовками EXE-файлів – метод, побудований на використанні властивостей структури виконуваних файлів і розробці навісного захисту.

3. Обман дизасемблера. Методи цієї групи полягають у тому, щоб заплутати дизасемблер, або підсунувши дані замість коду, або дезорієнтувати його логіку, повести його по помилковому сліду, підсунути зайві фрагменти коду і т. д. Всі ці способи обману можна поділити на такі групи:

– ускладнення (утруднення) логіки;

– перемішування коду – авангардний і досить перспективний метод захисту ПЗ не тільки від дизасемблювання, а й від налагодження;

– різноманітні методи обфускації коду.

4. Методи емуляції, серед яких виділяють:

– емуляція процесора;

– емуляція мультизадачності.

2.3.5 Захист від динамічного дослідження

Якщо дизасемблювання є першим кроком щодо статичного вивчення програми, то наступним кроком «дослідника» програми буде вивчення її роботи в динаміці, під контролем налагоджувачів. Налагоджувачі дозволяють аналізувати код в процесі його роботи, відстежувати і змінювати стан регістрів і стека, правити код «на льоту» – загалом, спостерігати за роботою програми і навіть активно в неї втручатися. Динамічні засоби дослідження вивчають програму, інтерпретуючи її в реальному або віртуальному обчислювальному середовищі. Ці засоби можуть будувати алгоритм програми тільки на підставі конкретної її траси, одержаної при певних вхідних даних. Тому задача отримання повного алгоритму програми в цьому випадку еквівалентна побудові вичерпного набору текстів для підтвердження правильності програми, що практично неможливо, і взагалі при динамічному дослідженні можна говорити лише про побудову деякої частини алгоритму. Більшість прийомів проти трасування програм взято з текстів вірусів і деяких захистів, що застосовувались в основному на програмному забезпеченні бухгалтерських програм. Основні заходи протидії несанкціонованому налагодженню такі:

– блокування налагоджувальних переривань клавіатури, екрана;

– робота напряму з контролерами пристроїв, прямі виклики BIOS;

– різноманітні перевірки в тілі програми;

– часовий контроль, «паралельні» процеси;

– використання стеків;

– захист від налагодження на процесорах типу i80386;

– використання апаратних особливостей процесора;

– використання особливостей роботи ОС і самих налагоджувачів.

2.3.6 Системи захисту від розпакування і дампінгу процесів

Дизасемблювати запаковану або зашифровану програму напряму неможливо. У такому випадку зламники часто використовують спеціальні засоби, за допомогою яких можна спробувати витягнути з пам'яті комп'ютера знімок (дамп) програми у момент її роботи. Цей дамп вже можна більш менш успішно дизасемблювати. Більш того, на основі дампа можна відтворити виконуваний файл програми, причому цей файл успішно завантажуватиметься, запускатиметься і працюватиме. Саме на цьому принципі і основана робота більшості сучасних розпакувальників: піддослідна програма запускається під управлінням розпакувальника. Розпакувальник чекає деякої події, яка говорить про те, що програма повністю розпакувалася, тут же «заморожує» програму і скидає її дамп на диск. Захисні системи нерідко намагаються протидіяти отриманню роботоздатного дампа за допомогою маніпуляцій зі сегментами і таблицями імпорту-експорту. У цих випадках застосовують PE-реконструктори, тобто утиліти, що знаходять в дампі некоректні посилання на функції і намагаються їх відновити.

2.4 Основні алгоритми захисту програмного забезпечення

При розробці систем захисту програмного забезпечення використовуються такі алгоритми:

– алгоритми заплутування – використовуються хаотичні переходи в різні частини коду, впровадження помилкових процедур-«пустишок», холості цикли, спотворення кількості реальних параметрів процедур ПЗ, розкидання ділянок коду по різних областях оперативного запам'ятовувального пристрою (ОЗП) і т. п.;

– алгоритми мутації – створюються таблиці відповідності операндів-синонімів та їх взаємозамінність при кожному запуску програми за визначеною схемою або випадковим чином;

– алгоритми компресії даних – програма упаковується, а потім розпаковується по ходу виконання;

– алгоритми шифрування даних – програма шифрується, а потім розшифровується по ходу виконання;

– обчислення складних математичних виразів в процесі відпрацьовування механізму захисту – елементи логіки захисту залежать від результату обчислення значення будь-якої формули або групи формул;

– методи утруднення дизасемблювання – використовуються різні прийоми, спрямовані на запобігання дизасемблюванню в пакетному режимі;

– методи утруднення налагодження – використовуються різні прийоми, спрямовані на ускладнення налагодження програми;

– емуляція процесорів і операційних систем – створюється віртуальний процесор і/або операційна система (не обов'язково реально існуючі) і програма-перекладач із системи команд ІВМ у систему команд створеного процесора або ОС. Після такого перекладу ПЗ може виконуватися тільки за допомогою емулятора, що різко утруднює дослідження алгоритму ПЗ;

– нестандартні методи роботи з апаратним забезпеченням – модулі системи захисту звертаються до апаратури ЕОМ, минаючи процедури операційної системи, і використовують маловідомі або недокументовані її можливості.

Всі ці алгоритми або їх комбінація можуть бути використані при реалізації будь-якого виду захисту. Вибираючи певний алгоритм, певний вид і механізм захисту, студент повинен обґрунтувати свій вибір і довести доцільність його використання саме для захисту вибраної категорії програмних продуктів.

3 ВИМОГИ ДО РОЗРОБКИ ПРОГРАМНОГО ПРОДУКТУ

Результатом виконання курсової роботи повинен бути, як правило, повноцінний програмний додаток для конкретної операційної системи (сімейства Windows), призначений для здійснення захисту деякого об'єкта. Крім того, курсова робота може бути систематизацією наукових досягнень і програмних розробок у певній області захисту програмного забезпечення і поданням їх у вигляді методичної документації для проведення лабораторних робіт з дисципліни «Захист програмного забезпечення» або інших дисциплін. Розроблена програма повинна відповідати таким вимогам.

Розробка алгоритму захисту інформації для будь-якого відомого методу захисту або власного алгоритму захисту повинна бути здійснена на основі глибокого і ретельного аналізу літературних джерел з вибраної теми.

Перелік рекомендованої літератури наведено далі в окремому підрозділі даних методичних вказівок. Можливість використання та доцільність застосування вибраного методу захисту повинні бути всебічно обґрунтованими.

Розроблений алгоритм повинен бути порівняний з іншими методами захисту, наведено його переваги та недоліки.

А отже, перший розділ пояснювальної записки повинен бути присвячений саме теоретичним відомостям про вибрану проблему.

Об'єктом захисту може бути вихідний код програми певною мовою програмування, виконуваний код програми (exe-файл, функції DLL-бібліотеки). Тобто, захист може бути як вбудований (внутрішній), так і навісний (зовнішній). Механізм захисту повинен бути обумовлений заздалегідь і визначений в індивідуальному завданні.

Графічна частина розробки повинна подаватися у тексті пояснювальної записки до курсової роботи (або у додатках до неї) у вигляді відповідних рисунків, схем функціонування системи захисту, схем взаємодії програм, блок-схем конкретних алгоритмів, схем даних тощо. При цьому всі схеми, рисунки, блок-схеми повинні бути оформлені відповідно до прийнятих правил при оформленні технічної документації.

При програмній реалізації задачі допускається застосування будь-яких мов програмування (C++, C#, Delphi тощо) та будь-якого візуального середовища програмування, причому вибір тієї або іншої мови програмування і середовища програмування повинні бути також обґрунтовані.

Вважається перевагою для роботи застосування принципів об'єктно-орієнтованого програмування, використання API-функцій ядра операційної системи, системних функцій, функцій для роботи з файловою системою, з дисковою системою та конфігурацією комп'ютера на низькому рівні.

Реалізація дружнього інтерфейсу є обов'язковою належністю грамотно розробленої програми: використання багаторівневого меню, різноманітних елементів керування роботою програми, графічна інтерпретація результатів, попередження про можливі помилки при введенні інформації або під час інтерактивного режиму роботи і т. д. Обов'язковою є наявність вбудованої системи підказок і допомоги, розрахованих на звичайного користувача, необізнаного з тонкощами програмування.

Подання інформації (вхідної, проміжної, результуючої) повинно буди зрозумілим, мати необхідні пояснення. Всі наслідки вхідних, проміжних, результуючих дій повинні бути виведені на екран у вигляді, зручному для розуміння стороннього користувача і аналізу.

В результаті реалізації поставленої задачі необхідно виконати аналіз роботи розробленої програми, здійснити тестування її роботи у різних передбачених програмою режимах, *навести висновки* щодо ефективності розробленого захисту, можливих об'єктів захисту, умов використання вибраного методу захисту, його недоліків і переваг. Для формування висновків щодо ефективності захисту можна використовувати засоби статичного та динамічного аналізу.

Розробка інструктивних документів для роботи з програмою є обов'язковою складовою пояснювальної записки до курсової роботи.

Бажано, щоб ці інструкції фрагментарно супроводжували весь процес роботи програми у вигляді підказок та допомоги.

В будь-якому випадку курсова робота повинна містити конкретну реалізацію будь-якого методу захисту (або групи методів певного спрямування) і наявно демонструвати його роботу на конкретних прикладах, з реальними результатами. Курсова робота повинна бути реалізована і подана таким чином, щоб можна було користуватись нею сторонньому користувачу (звичайно, обізнаному з тематикою).

4 ОФОРМЛЕННЯ ПОЯСНЮВАЛЬНОЇ ЗАПИСКИ

4.1 Загальні правила оформлення

Курсова робота повинна бути виконана й оформлена з дотриманням вимог до наукових робіт. Оптимальний обсяг курсової роботи – 30–40 друкованих сторінок, список використаних джерел – близько 20.

Обов'язковою вимогою до курсової роботи є написання її державною мовою, за винятком списку використаних джерел, де використане джерело записується мовою видання. Цитати з цих джерел наводяться в тексті виключно українською мовою.

Текст курсової роботи набирається на комп'ютері на одному боці аркуша білого паперу формату А4 (210×297 мм). Сторінки обмежуються полями: ліве – 25 мм, верхнє та нижнє – 20 мм, праве – 10 мм. Вирівнювання по ширині обов'язкове. Відстань між заголовком і текстом – 15–20 мм. Шрифт – чорного кольору. Щільність тексту однакова по всій роботі. Рекомендований шрифт – Times New Roman, розмір – 14, інтервал між рядками – 1,5. Текст курсової роботи може ілюструватись рисунками, схемами, графіками, діаграмами та таблицями.

Курсова робота починається з титульної сторінки за формою, наведеною в додатку Б. Це перша сторінка курсової роботи, яка входить до загальної нумерації сторінок, але її не нумерують. Далі номер сторінки проставляють у правому верхньому куту аркуша. За титульною сторінкою наводяться послідовно анотація, зміст, вступ, розділи в порядку подання, висновки, перелік посилань, додатки.

Текст основної частини курсової роботи поділяють на розділи та підрозділи. Кожну структурну частину роботи починають з нової сторінки.

Заголовки структурних частин роботи «ЗМІСТ», «ВСТУП», «РОЗДІЛ», «ВИСНОВКИ», «ПЕРЕЛІК ПОСИЛАНЬ», «ДОДАТКИ» друкують по центру великими літерами, використовуючи шрифт з підвищеною інтенсивністю (жирністю).

Заголовки підрозділів друкуються малими літерами (крім першої великої) з абзацу. В кінці заголовка крапки не ставлять. Якщо заголовок складається з двох або більше речень, їх розділяють крапкою.

4.1.1 Вимоги до оформлення розділів та підрозділів

Структурними елементами основної частини є розділи, підрозділи, пункти, підпункти, переліки.

Розділ – головний ступінь поділу тексту, позначений номером і має заголовок. Підрозділ – частина розділу, позначена номером і має заголовок. Пункт – частина розділу чи підрозділу, позначена номером і може мати заголовок. Підпункт – частина пункту, позначена номером і

може мати заголовок. Заголовки структурних елементів необхідно нумерувати тільки арабськими цифрами.

Допускається розміщувати текст між заголовками розділу і підрозділу, між заголовками підрозділу і пункту. Кожен розділ рекомендується починати з нової сторінки. Номер розділу ставиться перед назвою розділу, після номера крапка не ставиться. Підрозділи нумеруються в межах кожного розділу. Номер підрозділу складається з номера розділу і порядкового номера підрозділу, між ними ставиться крапка, наприклад: «2.3» (третій підрозділ другого розділу). У тому самому рядку після номера підрозділу оформлюється заголовок підрозділу. Цифри, які вказують номер, не повинні виступати за абзац.

В тексті документа може наводитись перелік, який рекомендується нумерувати малими літерами української абетки з дужкою або тире перед текстом. Для подальшої деталізації переліку використовують арабські цифри з дужкою.

Кожну частину переліку записують з абзацу, починаючи з малої букви і закінчуючи крапкою з комою, в кінці останньої ставлять крапку.

4.1.3 Оформлення формул

Формули і рівняння записують курсивом, вирівнюють по центру і позначають певним номером, написаним у дужках, який вирівнюється по правому краю.

Наприклад:

Корінь квадратного рівняння розраховується за формулою:

$$x = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}. \quad (4.1)$$

Формули (якщо їх більше, ніж одна) нумерують у межах розділу. Номер формули складається з номера розділу і порядкового номера формули в розділі, між якими ставлять крапку. Посилання на формули вказують порядковим номером формули в дужках, наприклад «... у формулі (3.1)».

4.1.4 Оформлення ілюстрацій

Для пояснення викладеного тексту рекомендується його ілюструвати графіками, кресленнями, фрагментами схем та ін., які можна виконувати чорною тушшю, простим олівцем середньої твердості та комп’ютерною графікою. Розміщують ілюстрації в тексті або в додатках.

В тексті ілюстрацію розміщують симетрично до тексту після першого посилання на неї або на наступній сторінці, якщо на даній вона не уміщується без повороту.

До ілюстрацій належать рисунки, схеми, графіки, діаграми. Ілюстрації і таблиці варто наводити в роботі безпосередньо після тексту за першою згадкою або на наступній сторінці. Ілюстрації і таблиці, розміщені на

окремих сторінках роботи, входять до загальної нумерації. Ілюстрацію, більшу за формат А4, враховують як одну сторінку і розміщують у додатках.

Ілюстрації позначають словом «Рисунок» і нумерують послідовно в межах розділу, за винятком ілюстрацій в додатках. Номер ілюстрації складається з номера розділу і порядкового номера ілюстрації, між ними ставиться крапка.

Наприклад:

Рисунок 1.2 (другий рисунок першого розділу).

Номер рисунка і його назва розміщуються послідовно під ним і вирівнюються по центру.

Наприклад:

Рисунок 3.3 – Вигляд головного вікна програми

Якщо частини ілюстрації не вміщуються на одній сторінці, то їх переносять на наступні сторінки. В цьому випадку, назва ілюстрації розміщується на першій сторінці, пояснювальні дані пишуться на кожній сторінці, а під ними позначають «*Рисунок _, аркуш –* ».

Якщо в тексті є посилання на складові частини зображеного засобу, то на відповідній ілюстрації вказують їх порядкові номери в межах ілюстрації.

Якщо ілюстрація є фрагментом повної розробленої схеми, то для всіх компонентів вказують ті позиційні позначення, які вказані на схемі.

Якщо ілюстраціями є фотографії, то останні повинні бути наклеєні на стандартні аркуші білого паперу і позначені як рисунки.

4.1.5 Оформлення таблиць

Таблиці у тексті пояснювальної записки набираються основним шрифтом, в деяких випадках розмір шрифту може бути зменшений до 10–12. Підписи таблиць розташовуються над таблицею з вказанням її номера і назви та вирівнюються по лівому краю. *Наприклад:*

Таблиця 1.1 – Основні алгоритми захисту

	Алгоритм захисту	Опис
	Алгоритм перемішування	Використання хаотичних переходів в різні частини коду, впровадження помилкових процедур, холостих циклів тощо.
	Алгоритми мутації	Створення таблиць відповідності операндів-синонімів і заміна їх між собою при кожному запуску програми за визначеною схемою чи випадковим чином.
…	**… … …**	**… … …**

На всі таблиці мають бути посилання за формою « ... в табл. 1.1 або в дужках по тексту (табл. 1.1). Посилання на раніше наведену таблицю да-

ють зі скороченим словом «дивись» (див. табл. 1.1) за ходом чи в кінці речення.

При перенесенні частин таблиці на інші сторінки, повторюють або продовжують найменування граф.

Допускається виконувати нумерацію граф на початку таблиці і при перенесенні частин таблиці на наступні сторінки повторювати тільки нумерацію граф.

У всіх випадках найменування (при його наявності) таблиці розміщують тільки над першою частиною, а над іншими частинами зліва пишуть «Продовження таблиці 1.1» без крапки в кінці, наприклад:

Продовження таблиці 1.1

	2	3
5	Алгоритми компресії даних	Програма упаковується, а потім розпаковується в міру виконання
6	Алгоритми шифрування даних	Програма шифрується, а потім розшифровується в міру виконання
...	**... ... ...**	**... ... ...**

4.1.6 Додатки

Додатки оформляють як продовження курсової роботи на прикінцевих її сторінках, зазначаючи їх у порядку посилань у тексті. За умови, що додатків більше, ніж один, їх перелік починається з окремого аркуша, по центру якого великими літерами друкується слово «ДОДАТКИ». Кожен додаток починається з нової сторінки і повинен мати заголовок, надрукований вгорі малими літерами з першої великої, симетрично щодо тексту сторінки. Посередині рядка над заголовком малими літерами з першої великої друкується слово «Додаток».

Додатки позначаються послідовно великими літерами українського алфавіту, за винятком літер Ґ, Є, З, І, Ї, Й, О, Ч, Ь, наприклад: Додаток А, Додаток Б і т. д. Один додаток позначається як додаток А.

4.2 Загальна структура пояснювальної записки

Кожен етап виконання курсової роботи обов’язково має знайти своє відображення в пояснювальній записці. Пояснювальна записка повинна відповідати індивідуальному завданню, а її оформлення – чинному стандарту ДСТУ 3008–95, який слід враховувати на момент виконання розробки з врахуванням всіх офіційних змін, введених в дію.

Пояснювальна записка до курсової роботи повинна мати таку структуру:

а) вступна частина, яка містить:

– титульний аркуш;

– індивідуальне завдання;

– анотацію;

– зміст.

б) основна частина, яка складається зі:

– вступу;

– аналізу сучасних систем захисту програмного забезпечення;

– розробки алгоритму системи захисту програмного забезпечення та вибору середовища для розробки;

– розробки програмної реалізації системи захисту ПЗ, її тестування та інструкції роботи програми для користувача;

– висновків;

– переліку використаних посилань.

в) додатки, які розміщуються після основної частини пояснювальної записки курсової роботи.

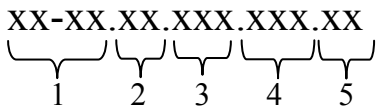
4.3 Вступна частина пояснювальної записки

4.3.1 Титульний аркуш

Титульний аркуш є першою сторінкою курсової роботи, яка не нумерується. Згідно з чинним стандартом титульний аркуш виконується за встановленим зразком. Зразок титульного аркуша пропонується у додатку Б.

На титульному аркуші для курсової роботи подаються: тема курсової роботи; запис „Пояснювальна записка ...» із зазначенням спеціальності, умовне позначення згідно з прийнятою системою (див. далі); перераховується науковий ступінь та звання керівника. Підписи керівника та студента із зазначенням термінів обов’язкові.

Для курсових робіт доцільною є предметна система умовних позначень, яка має таку структуру:



де 1 (ХХ-ХХ) – числовий шифр кафедри, прийнятий у ВНТУ (08–42);

2 (ХХ) – КП, якщо це курсовий проект або КР, якщо курсова робота;

3 (ХХХ) – умовне скорочення для дисципліни (ЗПЗ – захист програмного забезпечення);

4 (ХХХ) – варіант завдання (номер залікової книжки студента);

5 (ХХ) – код документа (ПЗ – пояснювальна записка, ТЗ – технічне завдання).

Слід зазначити, що робота, яка подається у вигляді копії, до захисту не береться.

4.3.2 Індивідуальне завдання

Конкретний зміст кожної курсової роботи, етапи її виконання визначає керівник на підставі індивідуального завдання, затвердженого завідувачем кафедри.

Попередньо керівник видає індивідуальне завдання до курсової роботи. Індивідуальне завдання в перелік змісту не вноситься і має бути другою сторінкою після титульного аркуша. Зразок індивідуального завдання до курсової роботи наведено в додатку Г.

Керівник курсової роботи пропонує зміст пояснювальної записки, як правило, в розроблених методичних вказівках, або в навчальних цілях зміст може висвітлюватись в індивідуальному завданні.

В залежності від специфіки дисципліни керівник КР може пропонувати тему, яка підлягає конкретному обґрунтуванню та розробці індивідуального завдання. Індивідуальне завдання до курсової роботи має містити термін видачі, підписи керівника та студента.

Завдання на курсову роботу повинно бути підготовлено студентом не пізніше другого тижня з початку навчального семестру, підписано викладачем, що видав завдання, і студентом, що прийняв його до виконання.

4.3.3 Анотація

Анотація призначена для ознайомлення з текстовим документом курсової роботи. Вона повинна коротко характеризувати мету роботи, засоби, використані для досягнення поставленої задачі, коротку інформацію про досягнуті результати. Розмір анотації повинен становити приблизно 1/3 частину сторінки.

Анотацію розміщують безпосередньо за аркушем з індивідуальним завданням, починаючи з нової сторінки (третьої), нумерація якої не зазначається. Заголовок (слово АНОТАЦІЯ) розміщується по центру сторінки, після нього пропускається 1 рядок.

4.3.4 Зміст

Зміст розташовують безпосередньо після анотації, починаючи з нової сторінки. До змісту входять: вступ; послідовно перелічені назви всіх розділів, підрозділів, пунктів і підпунктів (якщо вони мають заголовки) роботи; висновки; перелік використаних джерел; назви додатків і номери сторінок, які містять початок матеріалу. Зміст не містить титульний лист, індивідуальне завдання на курсову роботу та анотацію. Сам зміст за нумерацією пояснювальної записки є, як правило, четвертою сторінкою.

Нумерація у змісті починається зі вступу (відповідно до нумерації у пояснювальній записці). Нумерація сторінок по всій пояснювальній записці, включаючи додатки, повинна бути наскрізною.

Назви заголовків змісту повинні однозначно відповідати назвам заголовків пояснювальної записки за текстом. Формування змісту у текстовому документі бажано здійснювати автоматично, використовуючи засоби вибраного текстового редактора. Назви усіх розділів повинні бути розміщені по центру аркуша, а назви підрозділів, пунктів, підпунктів – з абзацу. Для запобігання плутанині з назвами розділів, підрозділів і нумерацією сторінок, а також для полегшення редагування і внесення змін в основний текст пояснювальної записки рекомендується використовувати можливості автоматичного формування змісту у документах Microsoft Office Word. Приклад оформлення змісту можна бачити у додатку В.

4.4 Вміст і оформлення основної частини

4.4.1 Вступ

Вступ пишуть з нової пронумерованої сторінки із заголовком „Вступ» посередині великими літерами.

Текст вступу повинен бути коротким і висвітлювати питання актуальності, значення, сучасний рівень і призначення курсової роботи. У вступі і далі за текстом не дозволяється використовувати скорочені слова, терміни, крім загальноприйнятих. Якщо ж в тексті є необхідність використовувати певні загальноприйняті скорочення (абревіатури), то при введенні їх вперше в дужках слід вказати скорочення. І лише після цього дане скорочення можна використовувати по тексту. Наприклад, несанкціоноване копіювання (НСК), або системи захисту програмного забезпечення (СЗПЗ). У назвах розділів, підрозділів, пунктів і підпунктів використовувати скорочення не рекомендується.

Вступ повинен стисло висвітлювати такі питання:

– стан розвитку проблеми в даній галузі, до якої має відношення розробка;

– галузь використання та призначення даної розробки;

– мету та загальну постановку задачі;

– актуальність, яка повинна подаватись в останньому абзаці вступу з метою стислого викладання суті вибраної розробки.

Обсяг вступу не повинен перевищувати 1–2 сторінок.

4.4.2 Розділ 1 – Аналіз сучасних систем захисту програмного забезпечення

Даний розділ є обов’язковим і передбачає посилання на відомі вітчизняні і зарубіжні аналоги, на існуючі програмні продукти відповідної спрямованості тощо. Даний розділ має містити в собі:

1) класифікацію систем захисту програмного забезпечення в цілому (загальні теоретичні відомості про сучасні системи захисту, їх

актуальність, надійність, види та особливості, переваги та недоліки);

2) аналіз існуючих методів захисту від конкретної загрози вказаної в індивідуальному завданні (переваги та недоліки окремих методів, поступове підведення до актуальності та доцільності використання конкретного методу захисту відповідно до варіанта);
3) аналіз конкретного методу захисту програмного забезпечення відповідно до варіанта (обґрунтування доцільності реалізації даного методу захисту програмного забезпечення, опис його особливостей та переваг).

Посилання на джерела по тексту пояснювальної записки є обов'язковими. Запропоновані можливі варіанти розв'язання основного питання повинні підкріплюватись техніко-економічним аналізом та визначення оптимального варіанта. Рекомендований обсяг розділу – 5–7 сторінок.

4.4.3 Розділ 2 – Розробка алгоритму системи захисту програмного забезпечення від конкретної загрози (вказаної в індивідуальному завданні)

Даний розділ присвячений розробці алгоритму системи захисту та вибору середовища для реалізації. Він містить детальний опис кроків, які необхідно виконати для досягнення мети роботи, вимоги до виконуваної розробки, очікувані результати. Таки чином, в цьому розділі рекомендується висвітлити такі питання:

1) основні принципи роботи даного методу (опис головних механізмів, що використовуватимуться в реалізації програмного забезпечення, та інформація про додаткові технології) ;
2) розробка алгоритму роботи системи захисту (інтегрованого модуля чи програми захисту) ПЗ від конкретної загрози, вказаної в індивідуальному завданні (детальний покроковий опис алгоритму роботи системи захисту ПЗ, побудова блок-схеми алгоритму роботи програми). Опис процесу розробки системи захисту проводиться відповідно до алгоритму, наприклад:

«..Розглянемо алгоритм програми, яка генерує ключовий файл.

Крок 1. Запуск програми-генератора ключових файлів. Користувач відкриває програму та вибирає функцію «Згенерувати файл».

Крок 2. Генерація глобального унікального ідентифікатора (GUID). На цьому етапі відбувається генерація глобального унікального ідентифікатора, який в подальшому буде використовуватись як одна зі складових ключової інформації у файлі.

Крок 3. Шифрування GUID. На даному етапі вже згенерований глобальний унікальний ідентифікатор зашифровується алгоритмом асиметричного шифрування RSA.

Крок 4. Формування ключової інформації. Як ключова інформація використовуватиметься GUID згенерований в кроці 1 та його зашифрована форма, отримана у кроці 2. Таким чином, інформація, отримана з першого кроку, вписується між символами інформації, отриманої в другому кроці.

Крок 5. Запис ключової інформації. Інформація, отримана в попередньому кроці записується у файл.

Таким чином генерується ключовий файл. Схематичне зображення алгоритму зображено на рисунку 4.1..»;

3) вибір мови програмування (аналіз сучасних мов програмування та вибір найбільш доцільної для реалізації саме такого алгоритму програми).

4.4.4 Розділ 3 – Розробка системи захисту програмного забезпечення від конкретної загрози (вказаної в індивідуальному завданні)

Відповідно до проведених досліджень, на основі розробленого алгоритму, здійснюється програмна реалізація системи захисту та її тестування. Даний розділ містить в собі такі підрозділи:

1) розробка модуля захисту (відповідно до основних кроків алгоритму роботи програми відбувається детальне пояснення функціонування програми за допомогою опису окремих фрагментів програмного коду, наприклад:

«..Розглянемо детальніше метод, який зашифровує дані:

```
public string EncryptToRsa(string toEncrypt)
    { var rsa = new RSACryptoServiceProvider();
      rsa.FromXmlString(File.ReadAllText(«public.xml»));
      return
Convert.ToBase64String(rsa.Encrypt(Encoding.UTF8.GetBytes(toEncrypt), false));
}); ..»«
```

Рисунок 4.1 – Схематичне зображення алгоритму генерації ключового файлу

2) інструкція користувача роботи з програмою (опис роботи програми та зрозумілі вказівки для користувача за допомогою скрін-шотів та коротких пояснень);
3) тестування роботи програми (тестування розробленої програми на предмет коректності роботи програми, ефективності захисту, зручності подання матеріалів тощо. Також для формування інструкцій та рекомендацій по роботі з нею, тут необхідно продемонструвати весь хід виконання програми на всіх режимах її роботи).

4.4.5 Висновки

Висновки оформляють з нової пронумерованої сторінки, починаючи зі слова «ВИСНОВКИ» посередині великими літерами.

У висновках наводяться основні результати роботи над курсовою роботою. Коротко по основних розділах описуються етапи реалізації задачі курсової роботи.

На основі наведених досліджень надаються обґрунтовані висновки щодо переваг та недоліків застосування того чи іншого методу захисту, того чи іншого засобу при здійсненні реалізації задачі, недоліки та переваги розробки, труднощі при розробці програми та причини, що їх обумовили, і можливі шляхи їх подолання. Обов'язково слід навести можливі рекомендації прикладного застосування, об'єкти захисту, умови використання даного способу захисту програм та шляхи (перспективи) удосконалення розробленого ПЗ.

4.4.6 Перелік посилань

Перелік використаних посилань оформляють з нової пронумерованої сторінки, починаючи зі слова «ПЕРЕЛІК ПОСИЛАНЬ» посередині великими літерами більш високої насиченості, після чого пропускається один рядок.

Перелік містить список використаних джерел, які було використано в процесі виконання роботи, і на які повинні бути обов'язкові посилання в тексті пояснювальної записки. Література (книги, статті, патенти, журнали, інтернет-сторінки) в загальний список записується в порядку посилання на неї в тексті. Посилання на літературу наводять в квадратних дужках […], вказуючи порядковий номер за списком.

Кожне джерело повинно бути вказано разом з видавництвом, роком видання, кількістю сторінок. Літературу записують мовою оригіналу. У списку кожне джерело записують з абзацу, нумерують арабськими цифрами, починаючи з одиниці. Правильне оформлення певного джерела інформації можна переглянути у переліку літературних джерел у будь-якому навчальному посібнику. Якщо у списку використаних джерел є посилання на Інтернет-сторінки, слід наводити разом з назвою Інтернет-сторінки.

4.5 Оформлення додатків

В даній курсовій роботі в додатки має бути винесений повний лістинг програми. Він може бути суцільним в одному додатку, а може розділятися на декілька додатків, відповідно до структури програмного забезпечення, якщо воно складається з декількох окремих частин.

Програмний код в курсовій роботі має розмір 12 і шрифт Calibri.

У додатках можуть бути:

– додаткові ілюстрації, схеми або таблиці;

– матеріали, які через великий обсяг, специфіку викладення або форму подання не можуть бути внесені до основної частини (оригінали

фотографій, проміжні математичні доведення, лістинги дизасембльованого коду; протоколи випробувань; інструкції, методики, опис комп'ютерних програм, розроблених або використаних у процесі виконання роботи та ін.).

Додатки необхідно починати з нової сторінки, вказуючи зверху посередині рядка слово «Додаток» і через пропуск – його позначення. Додатки позначають послідовно великими українськими літерами, за винятком букв Ґ, Є, З, І, Ї, Й, О, Ч, Ь, наприклад, Додаток А, Додаток Б і т. д. Якщо додатків більше ніж літер, то продовжують позначати арабськими цифрами. Дозволяється позначати додатки латинськими літерами, за винятком букв *I* і *O*.

Кожен додаток повинен мати тематичний (змістовний) заголовок, який записують посередині рядка малими літерами, починаючи з великої.

Сторінки додатків нумеруються, продовжуючи загальну нумерацію у пояснювальній записці. Всі додатки входять у зміст, вказуючи номер, заголовок і сторінки, з яких вони починаються. Приклад оформлення додатків можна переглянути у додатках до даних методичних вказівок.

4.6 Графік виконання курсової роботи і порядок його захисту

Рекомендується такий графік виконання курсової роботи, який враховує самостійну роботу студентів під час 4-го триместру (16 тижнів) для денної форми навчання та 3-го семестру для заочної форми навчання.

Таблиця 1.2

Зміст розділу	Термін виконання
1	2
Отримання завдання на курсову роботу, розробка і оформлення індивідуального завдання	1 тижд.
Аналіз літературних джерел та сучасного стану проблеми, обґрунтування вибору методів захисту	2 тижд
Розробка структури програмного забезпечення: усвідомлення алгоритмів підзадач, виконання контрольних прикладів, розробка інтерфейсу, обґрунтування необхідності додаткових засобів, розробка структури вхідних і вихідних даних, підбір необхідних програмних засобів і т. д.	3–5 тижд.
Розробка програмного забезпечення і налагодження його: програмування та тестування основних процедур та функцій, програмна реалізація інтерфейсу, програмна реалізація роботи з файлами, з елементами керування, реалізація захисту та перевірки цілісності даних і т. д.	6–11 тижд.
Тестування розробки та виправлення виявлених недоліків. Підготовка контрольних прикладів, перевірка ефективності захисту	11 тижд.

Продовження таблиці 1.2

1	2
Оформлення пояснювальної записки до курсової роботи, розробка рекомендацій із експлуатації розробленої програми	12–13 тижд.
Здача курсової роботи на попередню перевірку: демонстрація роботи програми та чернетки пояснювальної записки	13 тижд.
Корегування і доповнення (при необхідності) програми згідно із зауваженнями керівника курсової роботи, врахування і виправлення пояснювальної записки	14–15 тижд.
Захист курсової роботи	15–16 тижні

Готовність до захисту курсової роботи визначає керівник за результатами попередньої перевірки якості пояснювальної записки та дієздатності програми. Записка повинна бути здана керівнику на перевірку не менше, як за тиждень до визначеного терміну захисту роботи. Якщо робота виконана в повному обсязі і не має принципових помилок, керівник допускає студента до захисту. В іншому випадку робота повертається студенту на доопрацювання. Після позитивного висновку про готовність курсової роботи студент повинен захистити його перед комісією у складі двох викладачів, які призначені кафедрою.

ПЕРЕЛІК РЕКОМЕНДОВАНОЇ ЛІТЕРАТУРИ

1. Казарин О. В. Безопасность программного обеспечения компьютерных систем : монографія. / Казарин О. В. – М. : МГУЛ, 2003. – 212 с.
2. Казарин О. В. Теория и практика защиты программ. / Казарин О. В. – М. : МГУЛ, 2004. – 450 с.
3. Соколов А. Защита от компьютерного терроризма : справочное пособие. / А. Соколов, О. Степанюк – БХВ-Петербург : Арлит, 2002. – 496 с.
4. Щеглов А. Ю. Защита компьютерной информации от несанкционированного доступа. / Щеглов А. Ю. – Санкт-Петербург : Наука и Техника, 2004. – 384 с.
5. Швидченко И. В. Анализ криптостеганографических алгоритмов / И. В. Швидченко // Проблемы управления и информатики. – 2009. – № 4. – С. 149–155.
6. Румянцев П. В. Исследование программ Win32: до дизассемблера и отладчика /Румянцев П. В. – М. : Горячая линия-Телеком, 2004. – 367 с.
7. Румянцев П. В. Работа с файлами в Win 32 API. / Румянцев П. В. – М. : Горячая линия-Телеком, 2002. – 216 с.
8. Касперский К. Техника и философия хакерских атак. / Касперский К. – М. : Солон-Р, 1999. – 272 с.
9. Касперский К. Компьютерные вирусы изнутри и снаружи. / Касперский К. – СПб. : Питер, 2006. – 527 с.
10. Дудатьєв А. В. Захист програмного забезпечення : навчальний посібник. Частина 1. / Дудатьєв А. В., Каплун В. А., Семеренко В. П. – Вінниця : ВНТУ, 2005. – 140 с.

ДОДАТОК А

Варіанти завдань на курсову роботу

Розробка захисту програмного забезпечення від несанкціонованого копіювання:

1) шляхом прив'язки до унікальних параметрів вінчестера та використання ключа активації;
2) шляхом прив'язки до особливостей файлової системи NTFS та використання ключа активації;
3) шляхом прив'язки до параметрів логічних дисків з використанням ключа активації;
4) шляхом прив'язки до унікальних параметрів процесора та використання ключа активації;
5) шляхом прив'язки до унікальних параметрів системної плати та використання ключа активації;
6) шляхом прив'язки до системного реєстру та використання ключа активації;
7) шляхом прив'язки до унікальних параметрів ОС з використанням ключа активації;
8) шляхом прив'язки до фізичних адрес мережевих адаптерів з використанням ключа активації;
9) шляхом використання серверу активації на основі протоколу TCP;
10) шляхом використання серверу активації на основі протоколу UDP;
11) шляхом використання серверу активації на основі протоколу HTTP;
12) шляхом використання серверу активації на основі розробленого протоколу.

Розробка захисту програмного забезпечення від несанкціонованого дослідження:

13) шляхом перевірки контрольної суми виконуваного файлу;
14) шляхом виявлення та блокування роботи програмних шпигунів;
15) шляхом переплутування і клонування функцій;
16) шляхом внесення надлишкового коду;
17) шляхом використання символьної обфускації;
18) шляхом використання складних бульових виразів та комбінаторних тотожностей;

Розробка захисту програмного забезпечення від несанкціонованого доступу:

19) шляхом обмеження функціональних можливостей;
20) шляхом обмеження часу роботи програми;
21) шляхом обмеження кількості запусків програми;
22) шляхом використання ключових файлів;
23) шляхом використання ключа зі зворотним алгоритмом перевірки;
24) шляхом використання серверу авторизації на основі протоколу TCP;
25) шляхом використання серверу авторизації на основі протоколу UDP;
26) шляхом використання серверу авторизації на основі протоколу HTTP для соціальної мережі «Вконтакте»;
27) шляхом використання серверу авторизації на основі протоколу HTTP для соціальної мережі «LinkedIn»;
28) шляхом використання серверу авторизації на основі протоколу HTTP для соціальної мережі «Facebook»;
29) шляхом використання серверу авторизації на основі протоколу HTTP для соціальної мережі «Twitter»;
30) шляхом використання серверу авторизації на основі протоколу HTTP для соціальної мережі «Google+»;
31) шляхом використання серверу авторизації на основі розробленого протоколу;
32) шляхом контролю кількості виконуваних екземплярів в локальній мережі;
33) шляхом контролю кількості виконуваних екземплярів в глобальній мережі;
34) шляхом використання тесту Тьюрінга;
35) шляхом використання графічного тесту авторизації.

Розробка захисту програмного забезпечення від динамічного дослідження:

36) виявлення установлення прапорця трасування;
37) виявлення точок зупину налагоджувачів;
38) перекручування ходу виконання програми під налагоджувачем;

ДОДАТОК Б

Приклад оформлення титульного аркуша

Міністерство освіти і науки України

Вінницький національний технічний університет

Факультет менеджменту

Кафедра МБІС

КУРСОВА РОБОТА

з дисципліни «Захист програмного забезпечення»

на тему: «РОЗРОБКА ПРОГРАМНОГО ЗАХИСТУ ВІД ПЕВНОГО ВИДУ АТАКИ/КОНКРЕТНОЇ ЗАГРОЗИ»

08-42.КР.ЗПЗ.058.ПЗ

Студента 2 курсу УБ-16 групи

напряму підготовки 6.170103

спеціальності «Управління інформаційною безпекою»

Іванова І. І.

(прізвище та ініціали)

Керівник к.т.н., доц. Карпінець В. В.
(посада, вчене звання, науковий ступінь, прізвище та ініціали)

Національна шкала _____
Кількість балів: _____ Оцінка: ECTS _____

Члени комісії:

_____/_____

(підпис) (ПІБ)

_____/_____

(підпис) (ПІБ)

м. Вінниця – 2016

ДОДАТОК В

Приклад оформлення змісту

ЗМІСТ

ДОДАТОК Г

Приклад оформлення індивідуального завдання

Міністерство освіти і науки України

Вінницький національний технічний університет

Факультет менеджменту

ЗАТВЕРДЖУЮ

Зав. кафедри МБІС, проф., д.т.н.

_____О. М. Роїк

(підпис)

«___» вересня 20__ р.

ІНДИВІДУАЛЬНЕ ЗАВДАННЯ

на курсову роботу

з дисципліни «Захист програмного забезпечення»

студенту Іванову Івану Івановичу групи УБ-15

ТЕМА «Розробка захисту програмного забезпечення від

певного виду атаки / конкретної загрози «

Постановка задачі

1. Розробити загальну структурну схему функціонування системи захисту від певного виду загрози, з використанням конкретного методу захисту програмного забезпечення.

2. Здійснити програмну реалізацію системи захисту.

3. Провести тестування програми та розробити рекомендації по роботі з нею у вигляді інструкцій.

Вихідні дані

Вид загрози: несанкціоноване копіювання програмного забезпечення

Метод захисту: прив’язка до особливостей файлової системи

Механізм захисту: інтегрований

Об’єкт захисту: виконуваний модуль програми.

Дата видачі «___» вересня 20____р. Керівник _____ Карпінець В. В.

(підпис)

Завдання отримав_____

(підпис)

Навчальне видання

МЕТОДИЧНІ ВКАЗІВКИ ДО ВИКОНАННЯ КУРСОВОЇ РОБОТИ З ДИСЦИПЛІНИ «ЗАХИСТ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ»

для студентів напряму підготовки 6.170103 «Управління інформаційною безпекою»

Редактор В. Дружиніна
Коректор З. Поліщук

Укладачі: Карпінець Василь Васильович
Кец Дмитро Олександрович

Оригінал-макет підготовлено В. Карпінець

Підписано до друку 13.05.2017 р.
Формат 29,7×42 ¼. Папір офсетний.
Гарнітура Times New Roman.
Ум. друк. арк. 2,13.
Наклад 40 пр. Зам. № 2017-127.

Видавець та виготовлювач
Вінницький національний технічний університет,
інформаційний редакційно-видавничий центр.

ВНТУ, ГНК, к. 114.
Хмельницьке шосе, 95, м. Вінниця, 21021.
Тел. (0432) 59-85-32, 59-87-38.
press.vntu.edu.ua; *e-mail:* kivc.vntu@gmail.com
Свідоцтво суб'єкта видавничої справи
серія ДК № 3516 від 01.07.2009 р.