

Центральноукраїнський національний технічний університет
Механіко-технологічний факультет
Кафедра кібербезпеки та програмного забезпечення

”Допущено до захисту”
Завідувач кафедри кібербезпеки
та програмного забезпечення
д.т.н., професор
_____ Олексій СМІРНОВ
“___” _____ 2026 р.

ВИПУСКНА КВАЛІФІКАЦІЙНА РОБОТА
за першим (бакалаврським) рівнем вищої освіти
на тему
“Програмне забезпечення системи безпеки програмно
визначаємих мереж на базі технології SDNS”

Виконав здобувач вищої освіти
IV курсу, групи КІ-22-2
ОПП «Комп’ютерна інженерія»
спеціальності 123 «Комп’ютерна інженерія»
_____ Карпова Є.І.
«___» _____ 2026 р.

Керівник проекту
кандидат технічних наук, доцент
_____ Марченко К.М.
«___» _____ 2026 р.
Рецензент _____

Центральноукраїнський національний технічний університет
Факультет Механіко-технологічний
Кафедра Кібербезпеки та програмного забезпечення
Освітній ступінь бакалавр
Галузь знань_12 “Інформаційні технології”
Спеціальність 123 “Комп’ютерна інженерія”
Освітньо-професійна (освітньо-наукова) програма “Комп’ютерна інженерія”

ЗАТВЕРДЖУЮ
Завідувач кафедри
д.т.н., проф.
_____ Олексій СМІРНОВ
« 27 » лютого 2026 року

ЗАВДАННЯ НА ВИПУСКНУ КВАЛІФІКАЦІЙНУ РОБОТУ ЗА ПЕРШИМ (БАКАЛАВРСЬКИМ) РІВНЕМ ВИЩОЇ ОСВІТИ ЗДОБУВАЧА ВИЩОЇ ОСВІТИ

Карповій Єлизаветі Ігорівні

(прізвище, ім’я, по батькові)

1. Тема роботи *Програмне забезпечення системи безпеки програмно визначаємих мереж на базі технології SDNS*

2. Керівник роботи *Марченко Костянтин Миколайович, канд. техн. наук, доцент*
(прізвище, ім’я, по батькові, науковий ступінь, вчене звання)

затверджені наказом вищого навчального закладу № 136-02 від 27.02.2026 року

3. Строк подання студентом роботи до захисту *23.05.2026 р.*

4. Мета та завдання випускної кваліфікаційної роботи: *Метою роботи є розробка програмного забезпечення системи безпеки програмно визначаємих мереж на базі технології SDNS*

5. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити)

1. Призначення та область використання.

2. Перегляд аналогічних існуючих систем.

3. Опис і обґрунтування проектних рішень.

4. Етапи програмування системи.

5. Впровадження системи в промислову експлуатацію.

6. Висновки

6. Перелік графічного матеріалу (з точним зазначенням обов’язкових креслень)

Структурна схема системи *1 аркуш*

Функціональна схема системи *1 аркуш*

Діаграма процесів *1 аркуш*

Блок-схема алгоритму роботи додатку *2 аркуша*

7. Дата видачі завдання « 27 » лютого 2026 р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів випускної кваліфікаційної роботи за першим (бакалаврським) рівнем вищої освіти	Строк виконання етапів випускної кваліфікаційної роботи за першим (бакалаврським) рівнем вищої освіти	Примітка
1.	Аналіз існуючих систем	10.03.2026 р.	
2.	Постановка задачі, оформлення ТЗ	15.03.2026 р.	
3.	Розробка моделі компонента	20.03.2026 р.	
4.	Розробка структур даних	25.03.2026 р.	
5.	Розробка алгоритмів зв'язку та відображення	30.03.2026 р.	
6.	Програмування алгоритмів	10.04.2026 р.	
7.	Оформлення ПЗ	17.04.2026 р.	
8.	Попередній захист роботи	23.05.2026 р.	

Дата видачі завдання
« 27 » лютого 2026 р.

Підпис керівника

Марченко К.М.
(прізвище та ініціали)

Завдання прийнято до виконання
« 27 » лютого 2026 р.

Підпис здобувача

Карпова Є.І.
(прізвище та ініціали)

АНОТАЦІЯ

Карпова Є.І. Програмне забезпечення системи безпеки програмно визначаємих мереж на базі технології SDNS. 123 Комп'ютерна інженерія. Центральноукраїнський національний технічний університет. Кропивницький. 2026.

В даній випускній кваліфікаційній роботі за першим (бакалаврським) рівнем вищої освіти розроблено програмне забезпечення, яке призначено для системи безпеки програмно визначаємих мереж на базі технології SDNS.

Метою розробки є програмне забезпечення системи безпеки програмно визначаємих мереж на базі технології SDNS.

Результат роботи – програмна реалізація системи безпеки програмно визначаємих мереж на базі технології SDNS.

В процесі роботи над програмною моделлю виконано аналіз існуючих апаратних та програмних засобів. В повній мірі описані всі компоненти розробленого програмного забезпечення.

Розроблено зручний інтерфейс користувача. Наведені інструкції по роботі з програмними засобами.

Програма може використовуватися на ПЕОМ з ОС Windows 10/11.

Програму розроблено в середовищі Python.

Ключові слова: комп'ютерна інженерія, безпека програмно визначаємих мереж, SDNS

ABSTRACT

Karpova Ye.I. Software for the security system of software-defined networks based on SDNS technology. 123 Computer Engineering. Central Ukrainian National Technical University. Kropyvnytskyi. 2026.

In this final qualification work for the first (bachelor's) level of higher education, software has been developed, which is intended for the security system of software-defined networks based on SDNS technology.

The purpose of the development is software for the security system of software-defined networks based on SDNS technology.

The result of the work is the software implementation of the security system of software-defined networks based on SDNS technology.

In the process of working on the software model, an analysis of existing hardware and software was performed. All components of the developed software are fully described.

A convenient user interface has been developed. Instructions for working with software are provided.

The program can be used on a PC with OS Windows 10/11.

The program was developed in the Python environment.

Keywords: computer engineering, software-defined network security, SDNS

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ І ТЕРМІНІВ	2
ВСТУП.....	3
1 ПРИЗНАЧЕННЯ ТА ОБЛАСТЬ ВИКОРИСТАННЯ	6
1.1 Призначення системи.....	6
1.2 Область застосування.....	7
2 ПЕРЕГЛЯД АНАЛОГІЧНИХ ІСНУЮЧИХ СИСТЕМ	9
2.1 Огляд існуючих систем, технологій, архітектур та програмних рішень за профілем теми випускної кваліфікаційної роботи за першим (бакалаврським) рівнем вищої освіти.....	9
2.2 Обґрунтування вибору засобів для побудови системи та мови програмування.....	17
2.3 Розгорнута постановка завдання	22
3 ОПИС І ОБҐРУНТУВАННЯ ПРОЕКТНИХ РІШЕНЬ	24
3.1 Опис функціонування системи	24
3.2 Розробка структурної схеми.....	28
3.3 Розробка функціональної схеми	35
3.4 Розробка діаграми процесів.....	41
4 РЕАЛІЗАЦІЯ РОБОТИ. РОЗРАХУНКИ І ЕКСПЕРИМЕНТАЛЬНІ ДАНІ, ЩО ПІДТВЕРДЖУЮТЬ ВІРНІСТЬ ПРОЕКТНИХ ТА ПРОГРАМНИХ РІШЕНЬ.....	43
4.1 Розробка блок-схем та опис алгоритмів функціонування системи.....	43
4.2 Захист розробленого програмного забезпечення.....	70
5 ВПРОВАДЖЕННЯ СИСТЕМИ В ПРОМИСЛОВУ ЕКСПЛУАТАЦІЮ	72
6 ОСНОВНІ ВИСНОВКИ.....	78
СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ	80

					ВКРБ-123.26.0046.00.00.ПЗ			
Вим.	Арк.	№ докум.	Підп.	Дата	<i>Програмне забезпечення системи безпеки програмно визначаємих мереж на базі технології SDNS</i>	Лім.	Аркуш	Аркушів
<i>Розроб.</i>	Карпова Є.І.					Б	1	86
<i>Перев.</i>	Марченко К.М.							
<i>Н.контр.</i>	Коваленко А.С.					ЦНТУ КІ-22-2		
<i>Затв.</i>	Смірнов О.А.							

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ І ТЕРМІНІВ

АБС	—	автоматизована банківська система
AIC	—	автоматизованих інформаційних системах
АРМ	—	автоматизоване робоче місце
БЗ	—	база знань
ВМ	—	віртуальна машина
ЕС	—	експертна система
ЗК	—	загальні критерії
ІБ	—	інформаційна безпека
ІС	—	інформаційна система
ІТ	—	інформаційні технології
КБ	—	комерційний банк
КС	—	конкурентоспроможність
ММЕ	—	міжмережевий екран
НЛ	—	системи нечіткої логіки
НМ	—	нейронні мережі
ПЗ	—	профіль захисту
ПК	—	програмний комплекс
СЗІ	—	система захисту інформації
ЦОД	—	центр обробки даних
SDN	—	Software-Defined Network – програмно-визначаємі мережі
SDNS	—	Software-Defined Network Security

ВСТУП

Актуальність теми. SDNS (Software-Defined Network Security) – це технологія захисту програмно-визначаємих мереж (Software-Defined Network, SDN). SDN, що використовують керуючий протокол OpenFlow, володіють рядом властивостей, які добре підходять для побудови безпечного й керованого обчислювального середовища:

- Парадигма потоку даних більше зручна для забезпечення безпеки, оскільки вона пропонує наскрізну (end-to-end) модель зв'язків, орієнтовану на сервіси, не пов'язану із традиційними обмеженнями маршрутизації.

- Логічно централізоване керування дозволяє у всій мережі здійснювати ефективний захист і моніторинг появи погроз.

- Гранульоване керування політикою безпеки може ґрунтуватися на розв'язуваних завданнях, використовуваних сервісах, організаційних і географічних критеріях, а не на фізичній конфігурації.

- Перехід до керування через програмування дозволяє здійснювати динамічне й гнучке налаштування політики безпеки.

У рамках цього напрямку ведуться наступні роботи:

- Розробка системи контролю доступу застосунків до ресурсів контролера. Для того, щоб реалізовувати свою логіку, SDN застосунку одержують доступ до різного роду операціям над мережевими ресурсами й ресурсами контролера, у тому числі й критичним. Як і будь-яке програмне забезпечення, SDN застосунку можуть містити помилки й уразливості, які можуть привести до незапланованого поведіння застосунку. Тому для задоволення вимог інформаційної безпеки SDN контролери повинні забезпечувати політики доступу до інтерфейсів керування мережевими пристроями й ресурсами контролера.

					ВКРБ-123.26.0046.00.00.ПЗ	Арк.
Вим.	Арк.	№ докум.	Підпис	Дата		3

– Аналіз захищеності протоколів SDN. Протоколи SDN забезпечують прямий доступ і можливість маніпуляції площиною передачі даних всіх мережевих пристроїв. Експлуатація можливих уразливостей у цих протоколах може привести не тільки до захвата деяких пристроїв, але й до компрометації всієї мережі. Отже, щоб уникнути негативних наслідків комп'ютерних атак, необхідно робити аналіз захищеності протоколів SDN.

– Розробка систем виявлення вторгнень. Перехід на ідеологію SDN дозволяє перенести в керуючий контур SDN цілий ряд функцій, виконуваних у цей час окремими спеціалізованими пристроями. Причому використання таких рішень може бути більше ефективним і мати ряд переваг як технічних, так і економічних. Одним з видів таких функцій можуть бути функції системи виявлення вторгнень.

– Виявлення скомпрометованих комутаторів в SDN. Як і в будь-якому програмному або апаратному забезпеченні, в SDN комутаторах можуть перебувати деякі помилки, що ведуть до уразливостей, експлуатація яких може привести до захвата (компрометації) SDN комутатора зловмисником. Незважаючи на те, що функції керування мережею винесені на контролер, захвата зловмисником деякого комутатора може привести до порушення працездатності мережі або порушенню безпеки переданих по мережі даних. Тому необхідні методи, які дозволять виявляти скомпрометовані комутатори.

Мета й завдання дослідження. Метою роботи є програмне забезпечення системи безпеки програмно визначаємих мереж на базі технології SDNS.

Для досягнення поставленої мети визначена програма дослідження, що складається з наступних завдань:

– Огляд існуючих систем безпеки програмно визначаємих мереж на базі технології SDNS.

– Дослідження системи безпеки програмно визначаємих мереж на базі технології SDNS.

					ВКРБ-123.26.0046.00.00.ПЗ	Арк.
Вим.	Арк.	№ докум.	Підпис	Дата		4

– Програмна реалізація системи безпеки програмно визначаємих мереж на базі технології SDNS.

Практична цінність отриманих результатів полягає в тому, що розроблені алгоритми дозволяють успішно вирішувати задачі безпеки програмно визначаємих мереж на базі технології SDNS.

Таким чином, виходячи з вищеперерахованого, програмне забезпечення системи безпеки програмно визначаємих мереж на базі технології SDNS, є актуальною задачею, яка потребує вирішення у даній випускній кваліфікаційній роботі за першим (бакалаврським) рівнем вищої освіти.

К6ПЗ_2026

					ВКРБ-123.26.0046.00.00.ПЗ	Арк.
Вим.	Арк.	№ докум.	Підпис	Дата		5

1 ПРИЗНАЧЕННЯ ТА ОБЛАСТЬ ВИКОРИСТАННЯ

1.1 Призначення системи

Активна побудова приватних хмар кардинальним образом змінило інформаційні системи сучасних організацій. Віртуальні машини стали повністю програмно-конфігуруємими. Це забезпечило формування дуже динамічного середовища, у якій ЦОДи й хмарні сервери міняються щодня й щогодини. У таких умовах серверам і додаткам для нормального функціонування потрібні певні мережеві сервіси, такі як:

- маршрутизація;
- балансування навантаження;
- безпека.

Традиційні процедури обслуговування мережі й керування нею стають стримуючим фактором у програмно-конфігуруємому середовищі. Все це привело до зародження концепції SDN, що покликана вирішити конфлікти між повільним налаштуванням традиційних мережевих сервісів вручну й необхідністю дуже швидкого задоволення потреб динамічного середовища програмно-конфігуруємих обчислень.

SDN формує середовище гнучкого й автоматизованого керування мережами й надання мережевих сервісів шляхом відділення в мережевому устаткуванні площини керування від площини пересилання даних і створення централізованого подання мережі. При цьому інтелектуальним центром стає SDN-контролер, що приймає рішення при виконанні мережевих операцій і наданні тих або інших сервісів. Таке централізоване середовище дозволяє швидко й ефективно пристосовуватися до постійних змін у хмарі й мінливих потребах організацій в ІТ.

					ВКРБ-123.26.0046.00.00.ПЗ	Арк.
Вим.	Арк.	№ докум.	Підпис	Дата		6

Почнемо з того, що SDN відокремлює друг від друга площини керування (Control Plane) і властиво передачі даних (Data Plane), які традиційно були тісно інтегровані в пропрієтарних мережевих елементах, таких як комутатори й маршрутизатори. У результаті утворюються три рівні:

- мережеві елементи SDN;
- SDN-контролер;
- застосунки SDN.

Через контролер мережеві адміністратори можуть задавати налаштування для всієї мережі програмним шляхом, не набиваючи вручну для тисяч пристроїв десятки тисяч рядків налаштовуючого коду. Завдяки інтелектуальним характеристикам SDN-контролера ІТ-служба має можливість міняти поведження мережі в реальному часі й розгортати нові застосунки й мережеві сервіси всього за кілька днів або навіть годин. Таким чином, за допомогою динамічних програм, що набудовуються автоматично, SDN дозволяє домогтися гнучкості конфігурування, керування, забезпечення безпеки й оптимізації мережевих ресурсів.

1.2 Область застосування

Безпека являє собою фундаментальний рівень ІТ-інфраструктури, не менш важливий, чим рівень зберігання, обчислювальний або мережевий рівні. Тому засобу безпеки теж повинні стати "програмно-конфігуруємими" – тобто такі ж гнучкі й еластичними, як і інші елементи інфраструктури ЦОДу.

Однією із ключових умов побудови гнучкої системи мережевої безпеки є інтеграція з контролером або основною платформою SDN, але настільки ж важлива й інтеграція з гіпервізорами, засобами керування хмарою й аналітичними інструментами. У деяких випадках елементи програмно-конфігуруємої системи безпеки можуть бути розгорнуті навіть раніше SDN-контролерів і комутуючих структур.

					ВКРБ-123.26.0046.00.00.ПЗ	Арк.
Вим.	Арк.	№ докум.	Підпис	Дата		7

Ми не думаємо, що буде існувати єдина стандартна платформа SDN, що будуть використовувати всі підприємства й постачальники послуг. Пристрою для забезпечення безпеки й продукти керування безпекою тепер можна не ізолювати від іншої частини інфраструктури, але вони повинні одержувати в реальному часі інформацію про зміни, що відбуваються в мережі й у ЦОДі. Тому рішення безпеки необхідно будувати на розширеній платформі, що підтримує інтеграцію й взаємодію з іншими компонентами інфраструктури через API і інші інтерфейси. Мова йде як про відкриті стандарти, так і про пропрієтарних інтерфейси: і ті, і інші мають свої добре відомі плюси й мінуси.

Таким чином, виходячи з вищеперерахованого, програмне забезпечення системи безпеки програмно визначаємих мереж на базі технології SDNS, є актуальною задачею, яка потребує вирішення у даній випускній кваліфікаційній роботі за першим (бакалаврським) рівнем вищої освіти.

К6ПЗ-2026

					ВКРБ-123.26.0046.00.00.ПЗ	Арк.
Вим.	Арк.	№ докум.	Підпис	Дата		8

2 ПЕРЕГЛЯД АНАЛОГІЧНИХ ІСНУЮЧИХ СИСТЕМ

2.1 Огляд існуючих систем, технологій, архітектур, програмних рішень за профілем теми випускної кваліфікаційної роботи за першим (бакалаврським) рівнем вищої освіти

Сьогодні рідкий ІТ-фахівець не чув про переваги SDN у порівнянні із традиційною реалізацією мереж. Завдяки видаленню інтелектуальної складової з мережевого устаткування й виводу її в окрему площину керування з'явилася можливість гнучко управляти транспортною інфраструктурою. Архітектура дозволяє також розширювати можливості мереж винятково програмними засобами, без додавання мережевих пристроїв. Зокрема, багатьом компаніям буде цікаво створити єдиний мережевий простір, що включає власний ЦОД і хмарні ресурси комерційного дата-центра, з можливістю переносу між ними застосунків зі збереженням політик безпеки й адресації. SDN-проекти в Україні поки численними не назвеш: вітчизняний ринок витримав традиційну паузу перед тим, як масово піти за світовими трендами. Одним з перешкод учасники ринку називають відсутність готовності замінити свою ІТ-інфраструктуру, що, на їхню думку, приведе до значних витрат. На витрати бізнес йти не готові, оскільки що існуючі мережі в більшості справно працюють і не вимагають радикального втручання. Однак всупереч побоюванням замовників багато світові вендори розробили гібридні рішення для «безшовного» впровадження SDN-продуктів в існуючі мережі, дозволяючи уникнути значних витрат на заміну устаткування. У деяких компаній не виправдалися надії на те, що вендори стануть випускати зверхдешеве устаткування (за рахунок виключення вартості інтелектуальної частини з кінцевої ціни) або що всі бажаючі зможуть побудувати мережі в технології SDN. Третина вендорів вважає, що їхні потенційні клієнти чекають, коли SDN і NFV стануть стандартом де-факто, і можна буде впроваджувати ці технології по обкачаних методиках з меншими ризиками.

					ВКРБ-123.26.0046.00.00.ПЗ	Арк.
Вим.	Арк.	№ докум.	Підпис	Дата		9

Забезпечення безпеки

Проте пауза закінчується. На українському ринку з'являється усе більше рішень, що дозволяють успішно впровадити SDN. Нові рішення для ЦОДів в області SDN (включаючи SDDC – Software-defined Data Center) з'явилися й в області безпеки. При забезпеченні безпеки ЦОДів необхідно виходити з аналізу можливих погроз, планування мережі й кінцевих цілей. Якщо захисту підлягає власний дата-центр комерційної організації, у першу чергу необхідно зосередитися на захисті внутрішніх ресурсів і забезпеченні захищеного середовища для підключення до цих ресурсів ззовні. Якщо ж захист впроваджується в операторський ЦОД, то на базі тих самих систем безпеки можна будувати як захист самого дата-центра, так і ІБ-сервіси для його клієнтів. Ця концепція активно розвивається в останні пару років, і накопичена маса прикладів її реалізації.

Відповідні системи пропонує компанія Fortinet, добре відома в першу чергу завдяки високопродуктивним рішенням для комплексної безпеки. Практично із самого зародження концепції SDN компанія початку працювати над її захистом від кіберзагроз. Перші віртуальні пристрої FortiGate-VM були створені для захисту віртуалізованих і консолідованих ЦОДів.

У міру розширення можливостей програмно-визначаємих систем з'являються нові системи, призначені для роботи у зв'язуванні з партнерськими рішеннями. Мова йде про SDN-контролери, бази оркестровки, гіпервізорах, керуванні хмарними середовищами, керуванні безпекою й аналітику. Причому підтримуються практично всі існуючі гіпервізори, що дозволяє говорити про універсальність рішень. Fortinet також інвестує в технології SDN, реалізуючи інтеграцію з OpenFlow і OpenStack.

Оригінальне рішення, що дозволяє захистити всі сервери, а не тільки периметр і сегменти мережі, було реалізовано при інтеграції FortiGate-VMX із платформою VMware NSX. У класичних ЦОДах зловмисник може вибрати як мета один з «забутих» низькопріоритетних серверів. Якщо йому вдається

					ВКРБ-123.26.0046.00.00.ПЗ	Арк.
Вим.	Арк.	№ докум.	Підпис	Дата		10

проникнути за захист периметра й перехопити керування сервером, він тривалий час може відносно вільно почувати себе усередині периметра й збирати інформацію. Завдяки ж зв'язуванню технології VMware NSX і рішення FortiGate-VMX така можливість для нього закривається, причому без значного зростання навантаження на обчислювальні потужності.

Також рішення Fortigate забезпечують додаткову безпеку Cisco Application Centric Infrastructure (ACI) в умовах орієнтованої на застосунки інфраструктури. Cisco ACI придбало популярність у моделі надання хмарних сервісів, роблячи їх більше гнучкими. Традиційно обчислювальні мережі й мережева безпека жорстко прив'язані до устаткування, що робить складної налаштування й підвищує операційні витрати. Спільне рішення Fortigate і Cisco дозволяє автоматизувати відновлення політик і підвищити прозорість безпеки.

Недавно компанія випустила нову систему безпеки (Software-Defined Network Security Framework) для дата-центрів. Платформа надає можливості інтеграції з рішеннями таких компаній, як HP, Ixia, PLUMgrid, Pluribus Networks, Extreme Networks і NTT. За твердженням розроблювачів, система вносить інновації в усі основні шари мережевої архітектури. На рівні Data plane реалізоване об'єднання сервісів безпеки з апаратних модулів у логічні сутності, що надає додаткові можливості по масштабованості й тісній інтеграції в комутаційну фабрику й мережеві потоки. На рівні Control plane – керування й автоматизація обробки політик безпеки з адаптивним розподілом навантаження для виключення затримок при забезпеченні безпеки й відповідності вимогам у динамічних оточеннях. Management plane – централізоване керування політиками безпеки й подіями фізичних і віртуальних застосунків, приватних і загальнодоступних «хмар» по всій інфраструктурі для забезпечення комплексної безпеки.

Платформа протягом приблизно півтора років проходила тестування в компаніях, у її вносилися доробки, додаткові функції. Паралельно йшла сертифікація, і сьогодні цей продукт сертифікований практично в усіх напрямках,

					ВКРБ-123.26.0046.00.00.ПЗ	Арк.
Вим.	Арк.	№ докум.	Підпис	Дата		11

які пропонує VMware для інтеграції рішень в SDN. І, відповідно, він може застосовуватися в реальних дата-центрах. Зокрема, інтерес до SDN Security Framework виявив один з великих українських операторів зв'язку, що проводить його тестування.

Захист інфраструктури Microsoft Hyper-V для фінансових організацій

Регулярні проблеми з порушенням конфіденційності корпоративних даних, атаки на інфраструктуру й застосунки, запуснені у віртуальному середовищі, прямі багатомільйонні втрати грошей банками – це результат використання засобів захисту інформації (СЗІ) не відповідній сучасній архітектурі віртуалізованих ЦОД. Регулятори, що відслідковують численні інциденти ІБ намагаються врахувати широке використання віртуальних середовищ в ІС. Це привело до практики використання застарілих СЗІ, розроблених для фізичного середовища у віртуальному, що було на руку виробникам, але понизило рівень захисту ІС клієнтів, що втратили через використання старих технологій гроші, інформацію й клієнтів. Виробники платформ віртуалізації теж не встигали за розвитком технічних засобів і компетенцій зловмисників, пропонуючи перехідні архітектури своїх рішень, що не відповідали вимогам для віртуалізованих ЦОД. Однією із ключових характеристик сучасних СЗІ є безагентна архітектура. Але в самою популярною три роки тому платформі віртуалізації VMware vSphere v. 5, що анонсувала «безагентний» антивірус, насправді використовувався агент vShield, що знижував ефективність і захищеність клієнтських застосунків у віртуальних ВМ. Першим гіпервізором, що запропонували SDN архітектуру віртуального комутатора, став в Windows Server 2025R2 Extensible SwitchHyper-V, що дозволив стороннім розроблювачам: Cisco, NEC, Inmon і 5nine запропонувати розширення безпеки власної розробки й уперше реалізувати безагентний антивірус у віртуальному середовищі.

Розширюваність комутатора дозволила додаткам безпеки працювати безпосередньо на рівні віртуальних мереж. Прикладами таких розширень є

					ВКРБ-123.26.0046.00.00.ПЗ	Арк.
Вим.	Арк.	№ докум.	Підпис	Дата		12

міжмережеві екрани з SPI (Stateful Packet Inspection) і DPI (Deep Packet Inspection), антивірусні сканери, системи виявлення й запобігання вторгнення (IDS/IPS) захищаючи від атак на рівні застосунків (DDoS, spoofing й т.п.), засобу керування смугою пропускання, що забезпечують QoS у віртуальному середовищі. Створюючи спеціалізовані драйвери для фільтрації й зміни TCP/ IP- пакетів, контролю й авторизації з'єднань, фільтрації протоколів, продукти партнерів перевіряють, фільтрують або відхиляють шкідливі або підозрілі пакети, використовуючи сигнатурні або евристичні методи аналізу. На відміну від СЗІ попереднього покоління, які забезпечували безпека тільки VM або хосту, сучасні технічні засоби дозволяють контролювати й захищати все середовище віртуалізації: parent partition хосту, vNIC, мережевий трафік, диски й пам'ять VM, не залишаючи уразливостей або неконтрольованих областей.

У чому ж проблема використання застарілих агентних СЗІ?

1. Багато вірусів намагаються блокувати роботу агента у VM. Халатний працівник або зловмисник можуть видалити агент або відключити його. Безліч останніх інцидентів починалися із проникнення й відключення СЗІ на робочому місці, а потім зловмисник одержував доступ до керування всією мережею підприємства або банку.

2. Бази сигнатур, їхні регулярні відновлення й сам антивірусний агент споживають великий обсяг ресурсів VM, завантажуючи мережу, приводячи до зниження продуктивності хосту або кількості VM, що зменшує економічний ефект від віртуалізації ресурсів.

3. При інфікуванні однієї або декількох VM на хості або одночасному скануванні підвищується кіл-у звертань до дисків, що приведе до антивірусного шторму й деградації продуктивності хосту.

4. Атаки на рівні віртуальної мережі або з однієї VM на іншу не визначаються апаратними COV контролююче фізичне середовище й атаки in/out. Цим активно користуються зловмисники, які за допомогою засобів соціальної інженерії або ненадійності персоналу одержують доступ до однієї VM і, потім,

					ВКРБ-123.26.0046.00.00.ПЗ	Арк.
Вим.	Арк.	№ докум.	Підпис	Дата		13

розвивають атаку на всю внутрішню мережу, обійшовши потужні й дорогі прикордонні міжмережеві екрани й COB.

Сучасні безагентні C3I, такі як Snipe Cloud Security для Hyper-V, у базовому режимі не встановлюють агенти у VM, скануючи VHD/VHDX VM за допомогою власного сервісу керування на хості. Продукт використовує сучасну технологію інкрементального сканування за допомогою драйвера CBT (Changed Block Tracking), обробляючи тільки змінені блоки даних файлів на диску VM. У результаті повторні сканування займають усього від 40 секунд до 3 минут, замість звичайного багатогодинного процесу, зберігаючи ресурси процесора й пам'яті. Сканування безагентного антивірусу виробляється до 70 разів швидше, ніж у застарілих рішень на базі агентів у VM. Також наявність вірусів сканується трафік HTTP віртуальної мережі, що взагалі недоступно технологіям попереднього покоління.

Отже, використовуючи сучасні безагентні технології захисту віртуального середовища компанії й організації можуть:

1. Захиститися від нових погроз у віртуальному середовищі й одержати надійний комплексний багаторівневий захист без «пробілів».
2. Зменшити вплив свого персоналу на безпеку.
3. Понизити трудомісткість забезпечення безпеки: немає необхідності перевіряти й обновляти сигнатури на кожній VM. Ці процедури автоматично виконуються на хості.
4. Уникнути конфлікту ресурсів, таких як антивірусні шторми

Інкрементальне сканування підвищує продуктивність віртуальної інфраструктури. На хості можна запустити більше VM, підвищити щільність віртуалізації й економічний ефект від її впровадження.

Розуміння серйозних погроз, пов'язаних з використанням застарілих технологій з'явилося й у банківській галузі, однієї із самих критичних до погроз і втрат.

					ВКРБ-123.26.0046.00.00.ПЗ	Арк.
Вим.	Арк.	№ докум.	Підпис	Дата		14

Один з лідерів ринку – компанія VMware придбала стартап Nicira, розроблювача комплексного рішення Software-defined networking (SDN). Доробивши й інтегрувавши їхнє рішення, VMware випустила свою платформу NSX, що є основою архітектури Software Defined Data Center (SDDC). Крім розширених функцій мережевого керування, платформа NSX використовується для побудови захищеної інфраструктури, що базується на міжмережевому екрані з SPI (Stateful Packet Inspection) і високої деталізації контролю по MAC, IP, портам групам active directory і т.д. NSX використовує рішення партнерів для реалізації розширених сценаріїв захисту інфраструктури: Palo Alto Networks, Checkpoint, Fortinet і McAfee. У складі сервісів безпеки, пропонуваніх партнерами VMware з'явився безагентний антивірус, міжмережевий екран рівня застосунків L7, IPS і моніторинг трафіку. Недоліком рішення є його ціна, що виробник анонсував на рівні 6000 доларів за 1 процесор.

Що ще з нових технологій безпеки очікує нас цього року? На другу половину року намічений реліз комерційної версії Windows Server 2025. Поряд з відновленнями перевірених безагентних розширень віртуального комутатора Hyper-V, ми чекає новий SDN світч – конкурент NSX, що крім традиційних завдань керування мережевим трафіком і його безпекою, забезпечить можливість побудови гібридної хмари й прозорого переміщення робочих навантажень в Microsoft Azure і назад.

Основні можливості Microsoft Network Controller аналогічні NSX:

1. Наявність RESTful API для керування SDN Host Agent, запущеними на кожному хості Hyper-v. SDN Host Agent управляє кожним Hyper-V Virtual switch.
2. Підтримка VxLAN або NVGRE для створення логічних мереж L2 у рамках уже існуючих мереж L3.
3. Наявність балансувальника навантаження Software Load Balancer, що транслює зовнішні IP адреси у віртуальні з використанням BGP

Нові технології дозволили створити багаторівневий керований міжмережевий екран Datacenter Firewall. Він багатокористувальницький,

					ВКРБ-123.26.0046.00.00.ПЗ	Арк.
Вим.	Арк.	№ докум.	Підпис	Дата		15

мережевого рівня, з фільтрацією за протоколом, типам порту й IP-адреси, Stateful Packet Inspection і захищає віртуальні мережі клієнта від небажаного трафіку з Інтернет і інтранет мереж. На рівні хосту безагентним захистом VM забезпечить антивірус і IDS від Snipe.

Крім того, в Windows Server 2025 з'явився ще ряд нововведень, що підвищують безпеку клієнтів у приватній, публічному або гібридній хмарі Microsoft.

Для захисту VM в Windows Server 2025 з'явиться технологія Shielded Virtual Machines, що дозволяє зашифрувати диски VM із боку гостьового адміністратора, надійно заклавши його вміст від адміністратора хосту за допомогою клієнтського Bitlocker.

З'явилися принципово нові інструменти безпеки, що використовують можливості великого Azure. Перша – це Microsoft Advanced Threat Analytics (ATA). Це платформа, що розвертається в приватній хмарі, що використовує трафік Active Directory і дані SIEM для виявлення й попередження про потенційні атаки за допомогою просунутих технологій, що використовують machine learning. ATA виявляє відхилення в поведінці, шкідливі атаки й «вузькі місця» у безпеці.

Друга платформа – це Microsoft Operations Management Suite (OMS). Він обробляє журнали безпеки й подій MME в приватній або публічній хмарі для аналізу й виявлення нестандартного поведінки. Одержавши дані аналізу, співробітник інформаційної безпеки може вирішити, потрібно чи подальше розслідування. потім використовувати можливості докладного пошуку, щоб відслідковувати й досліджувати інциденти. Це добре масштабований хмарний сервіс, що у стані обробляти декількома терабайт даних щодня. Ви можете використовувати можливості Microsoft Azure, щоб співвіднести дані, отримані у вашій мережі з базами IP адрес, отриманих від спеціалізованих фірм, урядових агентств і інших джерел.

Всі ці сучасні технології допоможуть вам створити багаторівневий захист, що поєднує обчислювальні потужності, що використовують як хмарні ресурси,

					ВКРБ-123.26.0046.00.00.ПЗ	Арк.
Вим.	Арк.	№ докум.	Підпис	Дата		16

так і власні ЦОД. Ви можете відмовитися від величезних разових витрат на малоефективні СЗІ попередніх поколінь, неефективних у віртуальному середовищі, комбінувати покупку сучасного безагентного ПЗ з оплатою сервісів на вимогу, у т.ч. по моделі SECaaS. Це дозволить вам істотно підвищити захист критичної для бізнесу інформації, виконати вимоги регуляторів і не допустити перевитрати бюджету. Помнете, що при покупці антивірусу, міжмережевого екрана або IDS для віртуального середовища вам потрібно перевірити, безагентні чи вони?

2.2 Обґрунтування вибору засобів для побудови системи та мови програмування

Python – динамічна інтерпретована об’єктно-орієнтована скриптова мова програмування із строгою динамічною типізацією. Офіційний сайт мови програмування Python <https://www.python.org/>. Python – багатоцільова мова програмування, яка дозволяє писати код, що добре читається. Відносний лаконізм мови Python дозволяє створити програму, яка буде набагато коротше свого аналога, написаного на іншій мові. Python – багатоплатформова мова програмування. Це означає, що програми на Python можна запускати в різних операційних системах без будь-яких змін.

Ще однією перевагою Python є його стандартна бібліотека, яка встановлюється разом з Python і містить готові інструменти для роботи з операційною системою, веб-сторінками, базами даних, різними форматами даних, для побудови графічного інтерфейсу програм тощо. Програми, написані на мові програмування Python, можуть бути як невеликими скриптами, так і складними системами. Python абсолютно безкоштовний.

Швидкість виконання коду Python

Один з можливих недоліків Python – швидкість виконання коду. Python не є компільованою мовою. Код на Python спочатку компілюється у внутрішній

					ВКРБ-123.26.0046.00.00.ПЗ	Арк.
Вим.	Арк.	№ докум.	Підпис	Дата		17

байт-код, який потім виконується інтерпретатором Python. У більшості випадків при використанні Python виходять програми повільніші в порівнянні з такими мовами, як С.

Втім, сучасні комп'ютери мають таку обчислювальну потужність, що для більшості застосунків швидкість розробки важливіша швидкості виконання, а програми на Python зазвичай пишуться набагато швидше.

Окрім того, Python легко розширюється модулями, написаними на С або С++. Такі модулі можуть використовуватися для виконання частин програми, що створюють інтенсивне навантаження на процесор.

Використання Python

Python використовується для різних цілей: для створення ігор і веб-застосунків, розробки внутрішніх інструментів для різноманітних проектів. Мова також широко застосовується в науковій області для досліджень і розв'язування прикладних завдань.

Застосування мови програмування Python:

1. BitTorrent – протокол для обміну даними.
2. Ubuntu Software Center – вільне програмне забезпечення для пошуку, установки і видалення пакунків в системі Ubuntu Linux.
3. Blender – програма для створення тривимірної комп'ютерної графіки, що включає засоби моделювання, анімації, вимальовування, пост-обробки відео, а також створення відеоігор.
4. GIMP – растровий графічний редактор, із підтримкою векторної графіки.
5. World of Tanks.
6. Вільна енциклопедія Вікіпедія.
7. Пошукова система Google.
8. DropBox – файловий хостинг, що включає персональне хмарне сховище, синхронізацію файлів і програму-клієнт.
9. YouTube – популярне відеосховище.

Версії Python

Мови програмування з часом змінюються – розробники додають в них нові можливості, а також виправляють помилки. Так з'являються різні версії мови. Наприклад, код написаний на Python 2 у більшості випадків не буде працювати у версії Python 3 без внесення додаткових змін.

Процесор є найважливішим компонентом в комп'ютері. Одна з основних функцій процесора – це обробка даних згідно комп'ютерної програми, яка є списком інструкцій, шляхом виконання арифметичних і логічних операцій над фрагментами даних.

Кожна інструкція в програмі – це команда, яка «повідомляє» процесору, яку операцію він повинен виконати. Процесор комп'ютера може розуміти лише ті інструкції, які написані на машинній мові. Машинна мова – це штучна мова, створена для передачі команд комп'ютеру. За допомогою машинної мови створюються ефективні програми, оскільки розробник отримує доступ до всіх можливостей процесора. Машинна мова – мова низького рівня.

Інструкція машинної мови існує для кожної операції, яку процесор здатний виконати – є інструкція для додавання чисел, є інструкція для віднімання чисел і т.д. Увесь набір інструкцій, який центральний процесор може виконати, відомий як набір інструкцій процесора.

Наприклад, у вас є певна програма, яка зберігається на диску вашого комп'ютера. Для виконання програми, ви здійснюєте подвійний клік на значку програми. Це змушує програму копіюватися з диска в оперативну пам'ять, після чого процесор комп'ютера виконує копію програми, яка знаходиться в оперативній пам'яті.

Коли процесор виконує інструкції програми, він бере участь у процесі, який є відомим як цикл `fetch – decode – execute` (отримати – декодувати – виконати). Цей цикл виконується для кожної інструкції у програмі і складається з трьох кроків:

Отримати

Програма – це послідовність інструкцій на машинній мові. Першим кроком циклу є завантаження (отримання) наступної інструкції з пам'яті в процесор.

Декодувати

Інструкція машинної мови – це двійкове число, яке представляє команду, що повідомляє процесору виконати певну операцію. На цьому кроці процесор декодує інструкцію, яку було «витягнуто» з пам'яті, для визначення того, яка операція повинна виконуватись.

Виконати

Останній крок циклу – виконати операцію.

Хоча процесор комп'ютера розуміє тільки машинну мову, людині непрактично писати програми на машинній мові. Така програма може мати тисячі або навіть мільйони бінарних інструкцій, і написання такої програми буде дуже обтяжливим процесом.

З цієї причини була створена мова асемблера як альтернатива машинній мові. Замість використання двійкових чисел для написання інструкцій, мова асемблера використовує короткі слова, відомі як мнемокоди.

Незважаючи на те, що мова асемблера не вимагає двійкових інструкцій, як у випадку машинної мови, проте вона вимагає високих знань про процесор. Використовуючи мову асемблера, навіть для найпростішої програми, необхідно написати велику кількість інструкцій.

Мова програмування високого рівня дозволяє створювати складні програми, не знаючи, як працює процесор, і не записуючи великої кількості інструкцій низького рівня. Крім того, більшість мов програмування високого рівня використовують слова, які легко зрозуміти.

Python – одна із популярних сучасних мов програмування високого рівня. Python – інтерпретована мова програмування. Python – це високорівнева інтерпретована мова програмування, на відміну від C++, яка є прикладом

					ВКРБ-123.26.0046.00.00.ПЗ	Арк.
Вим.	Арк.	№ докум.	Підпис	Дата		20

компільованої мови програмування. Назва Python відноситься як до мови програмування, так і до інтерпретатора – комп'ютерної програми, яка зчитує початковий код (написаний на Python) і виконує інструкції (команди).

Для перекладу мови високого рівня на машинну мову доступні два типи програм:

1. Компілятор.
2. Інтерпретатор.

Завантаження Python

Версії інтерпретатора Python для різних операційних систем доступні для безкоштовного завантаження за адресою <https://www.python.org/downloads>.

Середовище програмування для Python

Для написання програм використовують текстові редактори або інтегровані середовища розробки, які включають в себе різні інструменти для роботи з кодом: засіб для написання коду (текстовий редактор), інтерактивний інтерпретатор, відлагоджувач тощо.

Текстові редактори та інтегровані середовища програмування для Python:

- IDLE – стандартний редактор Python. Встановлюється разом з Python для користувачів Windows, окремим пакунком для користувачів Linux.
- Notepad++ – безкоштовний текстовий редактор початкового коду, який підтримує велику кількість мов, в тому числі і Python. Лише для користувачів Windows.
- Visual Studio Code – це легкий, але потужний редактор початкового коду, який розповсюджується безкоштовно і доступний у версіях для платформ Linux, Windows і macOS.
- PyScripter – інтегроване середовище розробки для мови програмування Python. Для користувачів Windows. Поширюється безкоштовно.
- Wing IDE 101 – вільне інтегроване середовище для Python, розроблене для навчання програмістів-початківців. Для користувачів Linux, Windows і macOS. Поширюється безкоштовно.

– Geany – вільний текстовий редактор з базовими елементами інтегрованого середовища розробки, доступний для операційних систем Linux, Windows і macOS.

– PyCharm – інтегроване середовище розробки для мови програмування Python. PyCharm є власницьким програмним забезпеченням. Наявна безкоштовна версія Community з усіченим набором можливостей. Для користувачів Linux, Windows і macOS.

– Thonny – IDE для вивчення програмування мовою Python. Для користувачів Linux, Windows і macOS.

– Mu – редактор коду Python для програмістів-початківців. Для користувачів Linux, Windows і macOS.

2.3 Розгорнута постановка завдання

Згідно з технічним завданням на випускню кваліфікаційну роботу за першим (бакалаврським) рівнем вищої освіти, реалізації підлягає програмне забезпечення, яке призначено для системи безпеки програмно визначаємих мереж на базі технології SDNS.

В процесі розробки випускної кваліфікаційної роботи за першим (бакалаврським) рівнем вищої освіти необхідно виконати наступний обсяг роботи:

а) провести аналіз існуючих систем-аналогів для виявлення їх позитивних і негативних якостей. Результати аналізу врахувати в подальших розробках;

б) вибрати та обґрунтувати методику побудови системи контролю роботи технологічного обладнання на виробництві в автоматизованому режимі. Розробити функціональну та структурну схеми системи;

в) розробити програмне забезпечення системи, що дозволить реалізувати поставлену технічним завданням задачу. Побудувати блок-схеми алгоритмів програми та підпрограми;

					ВКРБ-123.26.0046.00.00.ПЗ	Арк.
Вим.	Арк.	№ докум.	Підпис	Дата		22

г) організувати інтерфейс користувача з метою формування та виводу на екран ЕОМ повідомлень про некоректні дії користувача та нестандартні ситуації в роботі технологічного обладнання;

д) розробити рекомендації по організаційних та методичних заходах, які забезпечать впровадження системи в промислову експлуатацію та її подальшу успішну експлуатацію;

е) провести розрахунки по визначенню економічної ефективності розробленої системи;

ж) розробити заходи по охороні праці при впровадженні та експлуатації системи, а також розробити заходи з цивільного захисту;

з) сформулювати висновки про виконаний обсяг робіт та одержані результати.

К6ПЗ-2026

					ВКРБ-123.26.0046.00.00.ПЗ	Арк.
Вим.	Арк.	№ докум.	Підпис	Дата		23

3 ОПИС І ОБҐРУНТУВАННЯ ПРОЕКТНИХ РІШЕНЬ

3.1 Опис функціонування системи

Здається, немає проблеми в нових технологіях, що залучала б увагу більше, ніж безпека. У будь-якій технології практично завжди вважається (і часто небезпідставно), що безпеки в ній недостатньо. Завжди можна знайти привід для критики будь-якої технології, рішення, продукту, вендора через «недостатню безпеку» і одержати визнання й увагу до власної персони. Те ж саме можна сказати про SDN/NFV, які багатьма експертами оголошуються «чорними дірами» у безпеці.

Чи не так це насправді? Можна сказати лише те, що ще рано робити виводи. Подивимося, як насправді обстоять справи з безпекою в SDN/NFV.

Безпека – досить велике поняття, і щоб конкретизувати його щодо технологій програмувальних мереж і віртуалізації, насамперед «відокремимо мук від котлет». Почнемо з того, що розділимо поняття безпека на дві основних категорії – безпека контенту (шкідливе ПЗ, віруси, різні файли, які при завантаженні завдають шкоди комп'ютеру користувача) і безпека з'єднання, під яким розуміється здатність мережі підключати тих, кого треба, і не підключати тих, кого не треба.

Проблеми з безпекою контенту можуть виникати звідусіль, але найбільше їхнє джерело – Інтернет. Оскільки Інтернет ніколи не був, і, видимо, ніколи не буде безпечним джерелом, будь-який контент, отриманий через Інтернет, за замовчуванням вважається небезпечним. І технології SDN/NFV із цим нічого поробити не можуть.

Деякі вендори просувають ідею, що віртуальні функції інтелектуального аналізу здатні «на лету» розпізнавати погрозу в контенті, однак навряд чи це буде коли-або реалізовано на практиці. Інструменти, які можуть відслідковувати

					ВКРБ-123.26.0046.00.00.ПЗ	Арк.
Вим.	Арк.	№ докум.	Підпис	Дата		24

шкідливі сайти, дійсно існують, але навряд чи вони коли-або зможуть «на льоту» розпізнавати шкідливий контент, від джерела, що звичайно вважався довірчим.

Таким чином, залишається розглянути безпека з'єднань, що полягає в наданні послуг авторизованим користувачам, і блокуванню всіх інших. У безпеці з'єднань можна розглянути три основних області:

1. Керування доступом до мережі, що полягає в тім, щоб тільки авторизовані користувачі могли підключатися до мережі.

2. Запобігання впливу на мережу, що означає здатність зупиняти тих, хто намагається зробити щось деструктивне відносно послуг даної мережі, зсередини її, або ззовні її.

3. Запобігання несанкціонованого перехоплення даних, тобто здатність запобігати пасивне перехоплення мережевого трафіку.

Керування доступом до мережі – це, у першу чергу, процес адміністрування послуг у сучасних мережах. При замовленні або зміні послуг, вони можуть додаватися на сайт (або віддалятися із сайту), або, у деяких випадках, до реквізитів користувача. Якщо послуга дозволяє підключення plug-and-play, то цей процес може бути автоматизований і заснований на підтверджених даних довіреного користувача (credentials). Ризики, які SDN або NFV можуть тут привнести, ставляться до порталів самообслуговування, де процес керування доступом (admission-control) вважається менш безпечним.

Запобігання впливу на мережу також розпадається на дві категорії. Перша (і найбільше часто обговорювана) – це протидія атакам типу «відмова в обслуговуванні» (denial-of-service). Друга – відкликання доступу в т.зв. «кінцевих крапок, що відхилилися» (maverick endpoints), наприклад, у тих користувачів, які роблять у мережі ризиковані дії, або чимсь порушують установлені в даній мережі правила безпеки. Для мереж, крім публічних, таких як Інтернет, або мереж з поділюваної між різними підприємствами інфраструктурою, обидві ці завдання зводяться до використання механізму визначення джерела ризику (“finger-pointing”) і механізму відключення. SDN і NFV можуть грати тут як позитивну, так і негативну роль.

					ВКРБ-123.26.0046.00.00.ПЗ	Арк.
Вим.	Арк.	№ докум.	Підпис	Дата		25

Запобігання несанкціонованого перехоплення даних, якщо не розглядати завдання перехоплення на фізичних лініях у мережі доступу (ні SDN, ні NFV тут ні на що вплинути не можуть), то завдання також зводиться або до адміністрування, або до оперативного керування й контролю. Існують деякі способи перехоплення за допомогою пасивних знімачів (passive taps), які використовують деякі можливості SDN або NFV, здатні створити додаткові ризики.

Як і всі ризики, ризики безпеки необхідно усувати поетапно. Якщо SDN або NFV містять ті ж фактори ризиків, якими вже існують на традиційних мережах, то ці нові технології навряд чи що-небудь міняють у наявних методах забезпечення безпеки. Якщо вони привносять нові ризики, їх потрібно розглядати окремо.

SDN – це, по суті, централізовано керована структура передачі пакетів по мережі, у якій виключений процес адаптивної маршрутизації на кожному апаратному мережевому вузлі. Якщо центральний SDN-контролер і система адміністрування мережі в цілому скомпрометовані, то скомпрометованої виявляється вся мережа. Із цим важко сперечатися, але такий сценарій досить малоймовірний.

Сама більша уразливість у мережі SDN криється в лінці від мережевого елемента, позбавленого маршрутизації, до контролера SDN. Це головна проблема всіх систем віртуалізації взагалі: повинен бути дійсно внеполосний канал керування для площини передачі даних. Просто зашифрувати його – недостатньо, оскільки він може бути піддадуть атаці «відмова в обслуговуванні» (DoS).

З іншого боку, централізоване керування мережею передачі пакетів, що не знає топології з'єднання мережевих пристроїв, істотно ускладнює завдання захвата всієї мережі зі скомпрометованого пристрою. Це означає, що лінки між доменами SDN для комутаторів, орієнтованих на з'єднання, є безпечними. Тобто, безпека легше забезпечити чим захищати мережі від «пристроїв, що відхилилися» (maverick devices).

					ВКРБ-123.26.0046.00.00.ПЗ	Арк.
Вим.	Арк.	№ докум.	Підпис	Дата		26

Істотною перевагою SDN є її здатність розділяти мережі різних орендарів так, що вони зовсім невидимі друг для друга. Правила передачі пакетів, установлені централізовано, створюють явно певні маршрути й змінюються вони теж явно. Самі мережеві пристрої (без внутрішньої маршрутизації) не можуть змінювати маршрути, і, внаслідок того, що мережі орендарів не передають усередині себе ніяких загальних команд керування (тобто стосовно керування, вони є «позаполосними» – “out of band”), вони не можуть робити ніякого впливу на ресурси один одного. Тому, мережі SDN можуть бути істотно більше безпечними.

Для NFV всі набагато складніше. Те, що, було сконцентровано усередині одного апаратного пристрою в традиційній мережі, тепер розподілено по пулі ресурсів спільного використання. Віртуальні функції, що входять до складу віртуального «пристрою» тепер можуть бути розташовані на різних хостах і взаємоз'єднані через мережу. Це створює ризики у двох вимірах.

Один вимір – це горизонтальні мережеві з'єднання між віртуальними функціями. Що являє собою мережа, через яку вони проходять? Якщо припустити самий забавний варіант, що це – мережа загального користування, то стає можливим зламати кожний функціональний компонент. Навіть у тому випадку, якщо це – сама мережа передачі даних сервісів, те й тут окремі функції відкриті для зовнішніх з'єднань, для яких потрібно створювати захист. Навіть якщо вона є, однаково залишається ризик атак типу «відмова в обслуговуванні» з боку площини передачі даних.

Інший вимір – вертикальні зв'язки між функціями й ресурсами, на яких вони працюють. Сама більша складність в NFV – це забезпечення SLA, і в моделі ETSI цим займається “VNF Manager”, у якому по крайній сірці один компонентів повинен бути входити до складу VNF і працювати разом з нею. Оскільки менеджер VNF у принципі повинен працювати з урахуванням стану виділених для VNF ресурсів, те повинен бути канал від ПЗ орендаря (VNF/VNFM) до загальних ресурсів. Отут можливі значні ризики для безпеки й стабільності.

					ВКРБ-123.26.0046.00.00.ПЗ	Арк.
Вим.	Арк.	№ докум.	Підпис	Дата		27

Обоє ці виміри ризиків можуть збільшуватися тим фактом, що NFV привносить ризик вбудовування зловмисного ПЗ в мережу. Тобто, джерелом атаки може бути сама VNF, що може впливати на застосунки й даних користувачів у загальному пулі ресурсів. Будь-яка уразливість в VNF, якщо її можна скористатися, може бути справжньою катастрофою.

В NFV може бути трохи різний площин керування або мережі на різних незалежних пулах ресурсів. При цьому буде краще, якщо VNF буде мати власну IP-підмережа (subnet), у якій будуть «жити» її компоненти.

Загалом, можна сказати, що реальні проблеми безпеки в SDN і NFV перебувають поза рамками традиційних структур мережевої й інформаційної безпеки, оскільки тут потрібно забезпечувати безпека площини (площин) керування, які управляють віртуальними ресурсами і їхніми відносинами з додатками й сервісами. Ризики тут нарастають, оскільки ростуть і можливості «експонування» взаємозв'язків функцій і елементів через мережу. Проте, «ризики» SDN/NFV, дотепер представляються так, начебто вони є «розширенням» уже існуючих ризиків у традиційних мережах. Насправді це не так, і ризики SDN/NFV не можна усунути простим масштабуванням уже існуючих стратегій безпеки. Старі ризики залишаються, але процеси віртуальної безпеки зовсім інші, і необхідно розуміти, що потрібно забезпечувати, насамперед, безпека самої віртуалізації. Якщо цього не зробити, то традиційні ризики мультиплікуються й до них додадуться нові.

3.2 Розробка структурної схеми

Сьогодні, використання інформаційних технологій є одним із ключових моментів при створенні повністю автоматизованих підприємств. При цьому для мережі приділяється дуже важлива роль. Такі мережі повинні бути повністю захищені від непередбачених збоїв або погроз.[2] SDN мережа – мережа передачі даних, у якій рівень керування мережею відділений від пристроїв передачі даних і реалізується програмно.

					ВКРБ-123.26.0046.00.00.ПЗ	Арк.
Вим.	Арк.	№ докум.	Підпис	Дата		28

Ключові принципи SDN – поділ процесів передачі й керування даними, централізація керування мережею за допомогою уніфікованих програмних засобів, віртуалізація фізичних мережевих ресурсів.

Програмно-визначаємі мережі (SDN) завдяки автоматизації робочих процесів із застосуванням правил забезпечують прискорене розгортання й доставку застосунків в організаціях при значному скороченні витрат на інформаційні технології.

Технологія програмно-визначаємих мереж служить платформою для архітектур хмарних середовищ завдяки автоматизованій доставці застосунків по запиті й масштабованій мобільності.

Програмно-визначаємі мережі підсилюють переваги віртуалізації центрів обробки даних, сприяючи розширенню вибору ресурсів, їхньому економічному використанню, скороченню витрат на впровадження й експлуатацію інфраструктури.

Метою розробки такої технології як Software Defined Network (SDN), є спрощення основних мережевих компонентів, а також, за рахунок цього, зниження їхньої вартості й складності обслуговування.

Одними з найважливіших моментів є збільшення пропускної здатності каналу передачі даних і можливість централізованого керування мережею.

За рахунок цього в нас значно спрощується динамічної зміна мережі, оскільки всіма змінами керує оператор на єдиному контролері.

Ключова ідея – це створення спеціального програмного забезпечення, що дозволяє відокремити керування мережевим устаткуванням від рівня передачі даних.

В архітектурі SDN розрізняють площини даних, керування й застосунків.

На рівні застосунків, прямо повідомляється SDN контролеру про вимоги до мережі і її поведженні, за допомогою NBI-інтефейсу.

Крім цього, тут запитується інформація від контролера, що дозволяє йому мати цілісний погляд на мережу в цілому.

					ВКРБ-123.26.0046.00.00.ПЗ	Арк.
Вим.	Арк.	№ докум.	Підпис	Дата		29

Рівень передачі даних складається із пристроїв, названих SDN datapath, які представляють із себе мережевий пристрій, що забезпечує перенапрямок трафіку й декількох функцій обробки трафіку.

Взаємодія між SDN-контролером і SDN Datapath відбувається за протоколом OpenFlow [4], тому дуже часто такі пристрої також називають OpenFlow-пристроями або OpenFlow- комутаторами.

SDN-інтерфейс між площинами керування й даними (CDPI) є інтерфейсом, певним між SDN-контролером і SDN Datapath.

Даний інтерфейс повинен забезпечувати як мінімум наступні можливості:

- Програмне керування всіма операціями перенапрямку.
- Одержання всіляких статистичних звітів від OpenFlow-комутаторів.
- Оповіщення про події.

Передбачається, що CDPI буде реалізований відкритим, не залежним від виробника й інтероперабельним способом. Протоколом, що забезпечує взаємодію площини керування й площини даних, є протокол OpenFlow.

SDN Northbound інтерфейси (NBI) є інтерфейсами між SDN- додатками й SDN-контролерами. Вони надають абстрактний погляд на мережу й безпосередньо забезпечують необхідне мережеве поводження. Вони можуть бути реалізовані на будь-якому рівні абстракції й з різним набором функціональностей.[1] Розглянемо проблему зміни конфігурації на прикладі звичайних і SDN мереж.

Очевидно, забезпечення безпеки не повинне обмежуватися лише установкою спеціалізованого устаткування (брандмауерів або систем для уніфікованого керування погрозами – Unified Threat Management, UTM), оскільки воно дозволяє захистити тільки зовнішній периметр мережі.

Мережа повинна бути стійка до будь-якої цілеспрямованої або випадкової дій, чиненим користувачами усередині мережі через неуважність, через незнання внутрішніх регламентів або навмисного ігнорування складних і часто заплутаних

процедур доступу.[3] У роботі практично будь-якої мережі неминуча зміна її топології або конфігурації.

Це може бути: запланована зміна або переконфігурація устаткування, вихід устаткування з ладу, а також впливу зловмисника.

При зміні конфігурації топології мережі губиться довірений стан, у якому мережа функціонувала й ми маємо нову топологію або нову конфігурацію, при якій нам або відомо об зміні, що відбулося, або немає.

Маючи недовірений стан ми не можемо гарантувати подальшу безпечну роботу з мережею, до здійснення повторної перевірки на доручення конфігурації.

У такий спосіб нам необхідно провести оперативну оцінку змін, що відбулися, для ухвалення рішення, при якому ми зможемо продовжити роботу з мережею.

Це може бути як зміна параметрів засобів безпеки, так і простої підтвердження про те, що дана топологія й конфігурація мережі як і раніше безпечні, для подальшого функціонування мережі. Аналогічна ситуація характерна не тільки для традиційних мереж, але й також для мереж SDN. Тут ми бачимо, що передача даних в SDN мережі можлива по декількох каналах. При виході їх будуючи одного з маршрутизаторів ми не можемо гарантувати, що при переході на інший канал передачі даних ми можемо гарантувати довірений стан мережі, при якому можлива подальша безпечна робота з мережею.

При використанні засобів захисту, нам ніхто не може гарантувати, що вони не мають своїх недоліків, недоробок або бекдорів, що змушує адміністраторів мережі використовувати тільки комбінації засобів захисту від різних виробників, оскільки кілька незалежних джерел дають більше точну оцінку безпеки по однієї й тій же мережі.

Маючи ті самі потоки даних, два різних засоби повинні одержати однакову підсумкову оцінку стану мережі.

Для збереження безпеки SDN мережі ми хочемо використовувати спеціальний багатофункціональний програмно-апаратний комплекс, здатний

					ВКРБ-123.26.0046.00.00.ПЗ	Арк.
Вим.	Арк.	№ докум.	Підпис	Дата		31

виконувати ролі від звичайного дзеркалювання трафіку до ролей систем виявлення витоків і брандмауерів, залежно від завантаженої в нього функціональності, здатної динамічно змінюватися в невеликі проміжки часу.

Дані зміни будуть завантажуватися по спеціальному порту із сервера безпеки, і після чого автоматично встановлюватися в плинні декількох мінут.

Пристрій встановлюється в розрив між комутаторами SDN і здатний відслідковувати поточну конфігурацію мережі.

Цей пристрій призначений для підтримки довіреного стану мережі.

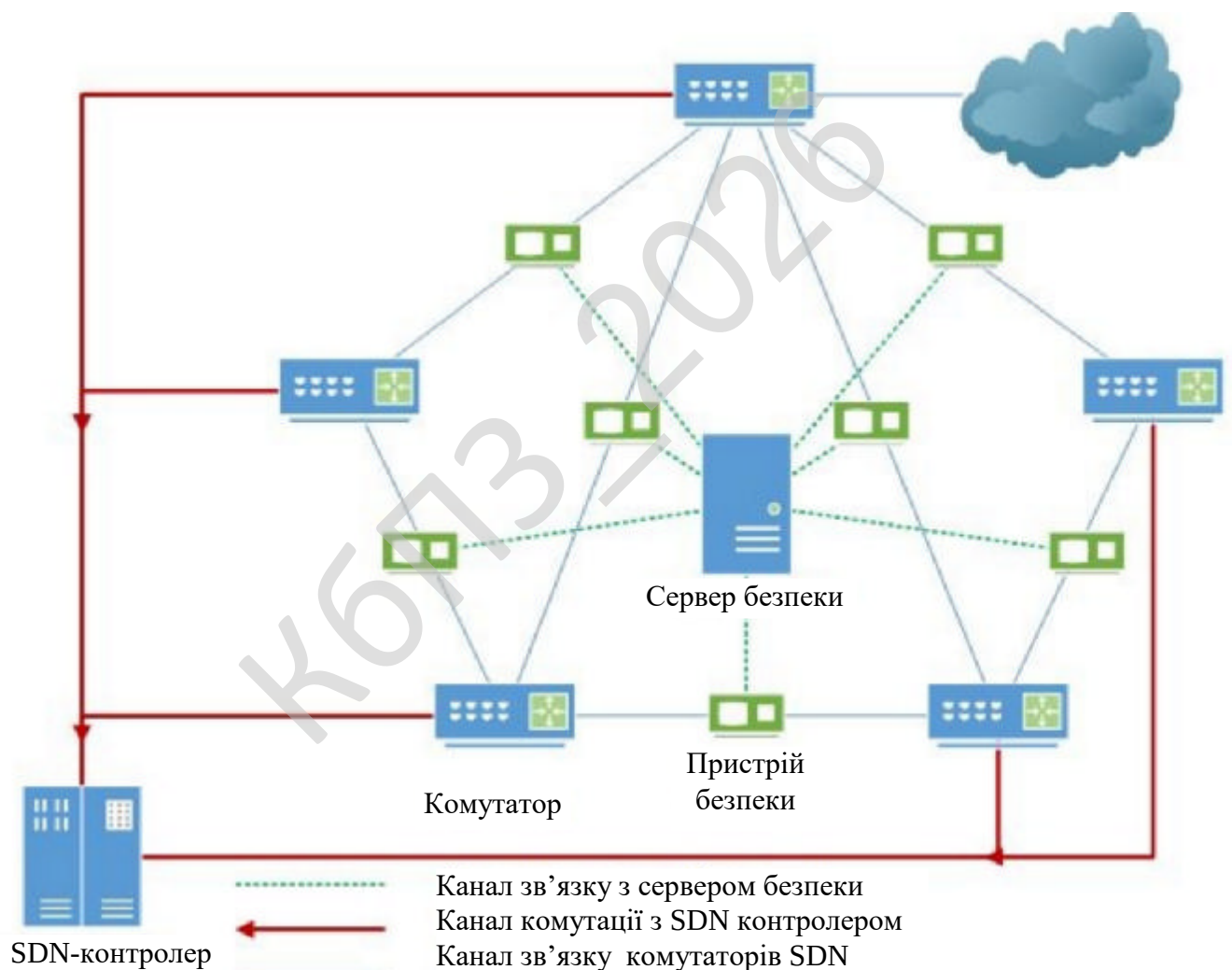


Рисунок 3.1 – Структурна схема системи

Пристрої безпеки розташовуються між комутаторами таким чином, що вони абсолютно не помітні для основної мережі. Вони мають по одному виході на кожний комутатор і третій вихід для зв'язку із сервером безпеки, що є керуючим для наших пристроїв безпеки. На ньому зберігається вся інформація про топологію мережі й конфігурація кожного пристрою безпеки.

Як концепт пристрої буде використовуватися комп'ютер Raspberry Pi, що має 3 ethernet виходи, поміщений у пилевологозахищений корпус. Unix-подібна операційна система дозволить обмежитися тільки необхідним набором функціональності ОС, і дозволить скоротити витрати пам'яті.

Програмне забезпечення, установлене на операційну систему ділиться на два види – це керуючі програми й прикладне програмне забезпечення по забезпеченню захисту мережі.

З керуючих програм виділяються засіб визначення зміни топології мережі й засіб для зміни конфігурації пристрою.

Засіб визначення зміни топології мережі

Засіб виконує порівняння поточної моделі й нової просканованої моделі. На основі отриманих результатів видається список змін состава й/або характеристики мережі(у тому числі пропускна здатність і завантаженість каналу).Порівняння виконується шляхом зіставлення графів досяжності мережі. По черзі рівняються стани вузлів(вершин) і ліній зв'язку(ребер) графів. У випадку, коли перебувають розходження – новий граф відправляється в БД можливих станів мережі. Далі граф відправляється на АРМ офіцера інформаційної безпеки для подальшого розгляду. Якщо змін не було виявлено, робота триває в штатному режимі.

Автоматизований засіб побудови моделі мережі

Завданням даного модуля є складання моделі мережі на основі опитування доступних для даного пристрою безпеки SDN вузлів. Із пристрою посилаємо broadcast запит з обох портів. Ці дві лінії зв'язку будуть ребром нашого графа, а вузли, які нам відповіли вершинами.

У процесі побудови моделі виділяємо наступні аспекти: досяжність вузла, час відгуку, пропускна здатність, поточного навантаження. Після складання моделі, вона передається в засіб визначення зміни топології мережі.

База даних можливих топологій мережі

У базі зберігаються всі стани топології, що зустрічаються раніше, мережі. У відповідність їм ідуть конфігурації Пристроїв безпеки SDN.

АРМ офіцера інформаційної безпеки

АРМ офіцера інформаційної безпеки займається дозволом питань про поточну конфігурацію Пристроїв безпеки SDN. Після одержання нового графа стану мережі, конфігурація пристроїв можлива в декількох режимах залежно від масштабів зміни графа. У випадку незначних змін виробляється автоматична переконфігурація або підтвердження того, що зміна конфігурації не потрібно.

Якщо зміни значні й подібний граф уже зустрічався, адміністратор може тільки підтвердити, що це аналогічний випадок і переконфігурування мережі потрібно виконати відповідно.

Якщо відбулися значні зміни й подібний граф не зустрічався, то адміністратор вручну встановлює нову конфігурацію.

Засіб для зміни конфігурації пристрою

Завданням даного модуля є одержання інформації від сервера безпеки SDN, прийняття дані прошивання й переконфігурації пристрою.

При зміні топології мережі, сервер безпеки одержує необхідні дані й ухвалює рішення щодо подальших функціональних особливостях кожного модуля. Він висилає необхідні конфігурації пристроїв безпеки у вигляді прошивань, які надалі будуть установлені даним модулем.

Представлений пристрій буде здатний забезпечити безпека SDN мережі на рівні передачі даних поза залежністю від зовнішніх факторів. У цьому випадку це дозволить надати незалежну оцінку безпеки мережі. При переконфігурації мережі всі необхідні зміни будуть відбуватися в автоматичному або напівавтоматичному режимі, таким чином, що довірений стан мережі не буде порушено.

3.3 Розробка функціональної схеми

Функціональна схема розробленої системи зображена на рисунку 3.2. Життєвий цикл процесу ухвалення рішення по формуванню профілю захисту в програмно-визначаємій мережі на базі технології SDNS, починається з аналізу проблемної ситуації особами, відповідальними за ініціацію процесу ухвалення рішення. Перш ніж ухвалити рішення щодо шляхів вирішення проблемної ситуації, відповідальні особи повинні зрозуміти зміст і ступінь її актуальності, і цей зміст досягається не індивідуально, а в процесі колективного всебічного аналізу проблемної ситуації, що здійснюється на 1 етапі життєвого циклу процесу ухвалення рішення по формуванню профілю захисту в програмно-визначаємій мережі на базі технології SDNS.

На даному етапі ініціатор процесу ухвалення рішення по формуванню профілю захисту в програмно-визначаємій мережі на базі технології SDNS, маючи дані про виниклу проблемну ситуацію, готує первинний її опис і обґрунтовану пропозицію по її вирішенню, після чого передає обґрунтовану пропозицію з вирішення проблемної ситуації в орган, уповноважений за організацію процесу ухвалення рішення по формуванню профілю захисту в програмно-визначаємій мережі на базі технології SDNS. Після одержання обґрунтованої пропозиції з вирішення виниклої проблемної ситуації орган, уповноважений за організацію процесу ухвалення рішення, повинен організувати всебічний його аналіз із залученням експертів, після чого прийняти рішення, чи заслуговує виникла проблемна ситуація вирішення, чи ні. Для цього створюється експертна група, що оцінює актуальність проблемної ситуації, і проводить її експертизу, метою якої є максимально наочний опис виниклої проблеми в термінах предметної області. Після підготовки експертного висновку, де визначена актуальність проблемної ситуації й намічені основні шляхи її рішення, орган, уповноважений за організацію процесу ухвалення рішення по формуванню профілю захисту в програмно-визначаємій мережі на базі технології SDNS,

					ВКРБ-123.26.0046.00.00.ПЗ	Арк.
Вим.	Арк.	№ докум.	Підпис	Дата		35

ухвалює рішення щодо включенні проблемної ситуації до реєстру прийнятих рішень, що у свою чергу запустить наступний етап процесу ухвалення рішення . Орган, уповноважений за організацію процесу ухвалення рішення по формуванню профілю захисту в програмно-визначаємій мережі на базі технології SDNS, на основі експертного висновку може також відхилити пропозиція з вирішення проблемної ситуації, якщо вона визнана експертами несуттєвою або недостатньо актуальною.

На другому етапі процесу прийняття рішень – етапі постановки завдання – здійснюється вибір найбільш прийняттого варіанта вирішення завдання (будь-яка складна проблемна ситуація припускає кілька можливих шляхів виходу зі сформованої ситуації, серед яких найчастіше буває дуже складно вибрати найбільш ефективний), а також підготовка й узгодження правового акту, вирішує проблемну ситуацію.

На етапі ухвалення рішення (третій етап) підготовлений правовий акт, вирішує проблемну ситуацію, розглядається на засіданні колегіального органа ухвалення рішення або керівництвом організації. Після затвердження даний правовий акт включається до реєстру правових актів.

Нарешті, на вирішальному четвертому етапі процесу ухвалення рішення по формуванню профілю захисту в програмно-визначаємій мережі на базі технології SDNS здійснюється моніторинг і оцінка результативності виконання ухваленого рішення органами виконавчої влади, результатом якого є експертний висновок з оцінкою результативності ухваленого рішення. На основі даного експертного висновку приймається рішення про включення виконаного рішення до реєстру прецедентів проблемних ситуацій. Це необхідно для того, щоб забезпечити можливість нагромадження успішного досвіду дозволу проблемних ситуацій з метою його наступного використання в майбутньому при розробці рішень при виникненні аналогічних проблем.

					ВКРБ-123.26.0046.00.00.ПЗ	Арк.
Вим.	Арк.	№ докум.	Підпис	Дата		36

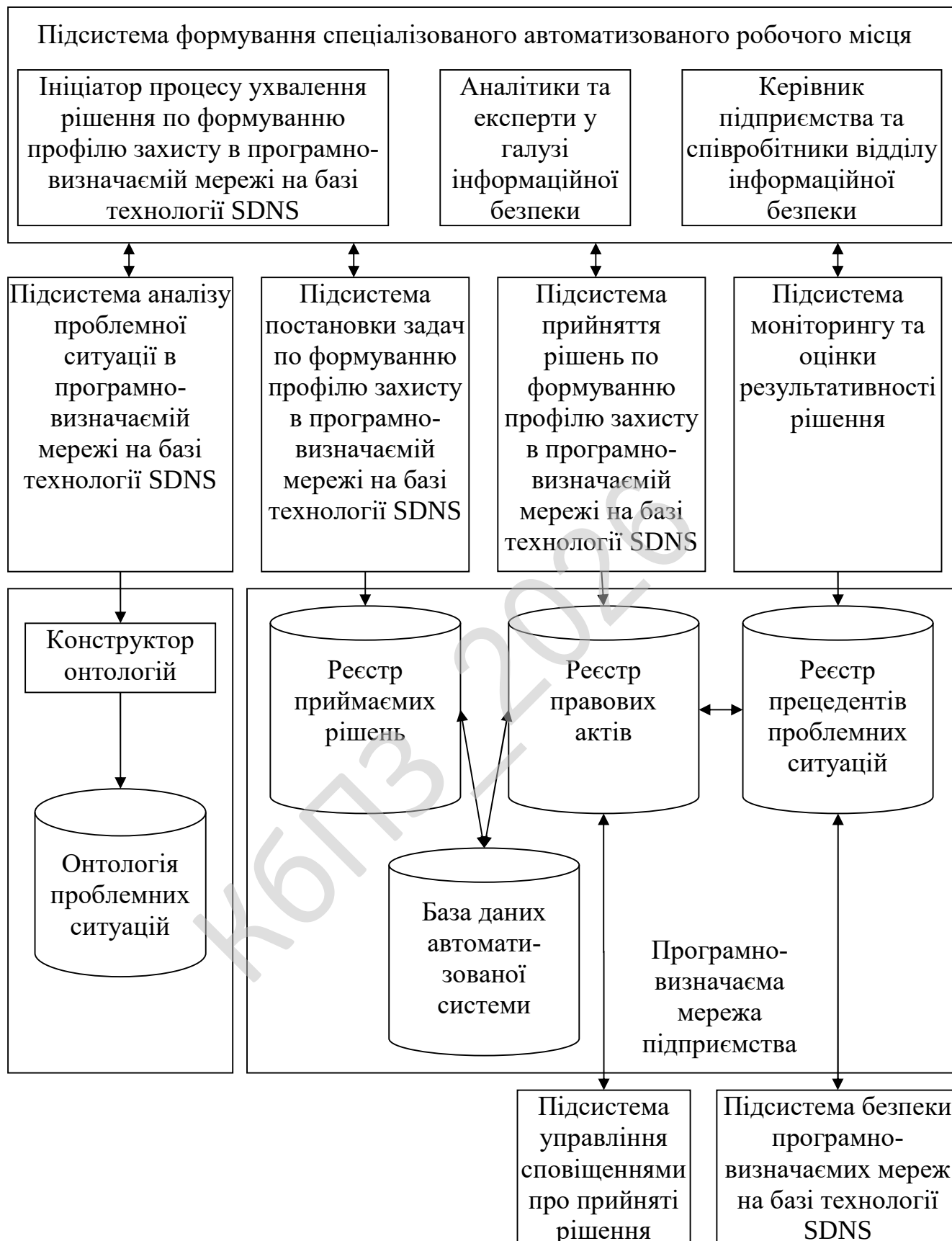


Рисунок 3.2 – Функціональна схема системи

Важливо відзначити, що процес прийняття управлінських рішень, по своїй природі, носить ітераційний характер. Це означає, що якщо отриманий варіант рішення проблемної ситуації за якимись причинами не влаштовує відповідальних осіб, здійснюється повернення на попередні етапи, де виробляється новий варіант рішення проблемної ситуації. Такі ітерації тривають доти, поки результат не буде погоджений всіма відповідальними особам або, можливо, ситуація зміниться й відпаде необхідність у вирішенні проблемної ситуації.

Аналіз періодично виникаючих проблемних ситуацій в області інформаційної безпеки, показує, що причина їхнього виникнення, насамперед, полягає у відсутності єдиної системи обліку однотипних рішень по одній проблемній ситуації з різних джерел.

Для усунення подібної ситуації було ухвалено рішення про вдосконалювання існуючої централізованої автоматизованої інформаційної системи забезпечення інформаційної безпеки. Поставлено завдання:

1. Створити єдину систему, що буде узагальнювати за схемою: проблема – проблемна ситуація – проблемний напрямок всі проблемні питання, рішення яких повинен забезпечити відділ захисту інформації.

2. На підставі єдиного реєстру проблемних ситуацій організувати більш ефективний часовий і змістовний моніторинг ефективності їхнього рішення.

Управлінським ефектом від реалізації 1 і 2 пунктів стане формування інструмента, що реально стимулює взаємодію між відділами підприємства.

Практична цінність від впровадження подібного програмного забезпечення полягає у формуванні єдиної системи обліку проблемних питань пов'язаних із захистом інформації на підприємстві, поставлених для рішення керівництвом організації. Наявність даного переліку не дозволить (при всьому бажанні) проігнорувати проблемну ситуацію відповідальною особою. Реально виникає механізм, який дозволяє інструментально вирішити проблему усунення суб'єктивних бар'єрів взаємодії між відділами.

					ВКРБ-123.26.0046.00.00.ПЗ	Арк.
Вим.	Арк.	№ докум.	Підпис	Дата		38

Реалізація викладеного підходу до прийняття міжгалузевих колегіальних рішень практично неможлива без використання спеціальних програмних засобів. На рисунку 3.2 представлена функціональна схема системи.

Функціональна схема системи підтримки процесу прийняття рішень реалізована у відповідності з розглянутою моделлю процесу ухвалення рішення по формуванню профілю захисту в програмно-визначаємій мережі на базі технології SDNS.

Онтологія – це спроба всеосяжної й детальної формалізації деякої області знань за допомогою концептуальної схеми. Звичайно така схема складається зі структури даних, утримуючі всі релевантні класи об'єктів, їхні зв'язки й правила (теореми, обмеження), прийняті в цій області. Цей термін в інформатиці є похідним від древнього філософського поняття «онтологія». В області штучного інтелекту онтологією називається експліцитна специфікація концептуалізації. Онтології використовуються в процесі програмування як форма подання знань про реальний світ або його частину. Основні сфери застосування – моделювання бізнес-процесів, семантична павутина (Semantic Web), штучний інтелект.

Склад модулів інформаційної системи підтримки прийняття рішень повною мірою відповідає завданням забезпечення необхідної функціональності системи формування профілю захисту в програмно-визначаємій мережі на базі технології SDNS, у тому числі підтримує завдання:

- забезпечення доступу співробітників підприємства, експертів і фахівців підвідомчих організацій до відповідних інформаційних ресурсів (баз знань і баз даних);
- забезпечення керування електронними процедурами в процесі прийняття й узгодження рішень по формуванню профілю захисту в програмно-визначаємій мережі на базі технології SDNS;
- забезпечення в організації інформаційно-управлінської взаємодії усередині підприємства;

					ВКРБ-123.26.0046.00.00.ПЗ	Арк.
Вим.	Арк.	№ докум.	Підпис	Дата		39

– забезпечення формування інформаційної інфраструктури, що підтримує експертні взаємодії, у тому числі організацію експертної групи, підтримку організаційних і інформаційних взаємодій усередині групи, обмін експертною інформацією в процесі прийняття управлінських рішень;

– забезпечення збору, обробки й зберігання експертної інформації (інформації про проблемні ситуації), а також застосування до неї аналітичних інструментів;

– забезпечення формування електронних архівів: реєстру прийнятих рішень, реєстру правових актів, реєстру прецедентів проблемних ситуацій. При цьому доступ до даних реєстрів, у рамках прав доступу, забезпечується в рамках автоматизованої системи підприємства

– забезпечення оперативного інформування про хід процесу ухвалення управлінського рішення;

– забезпечення прозорості діяльності учасників процесу прийняття управлінських рішень;

– забезпечення моніторингу й контролю виконання прийнятих рішень;

– забезпечення пошуку в інформаційних ресурсах системи;

– забезпечення захисту інформаційних ресурсів і програмно-технологічної інфраструктури від погроз інформаційної безпеки;

– забезпечення керування правами доступу до інформації й повноваженнями по роботі в системі, включаючи повноваження по зміні структури організації інформаційної системи (організації інформаційних взаємодій).

Основними результатами, очікуваними від впровадження й використання пропонованих методів і програмних засобів ситуаційного керування в регіоні, є наступні:

– Формування єдиної системи обліку проблемних ситуацій по інформаційній безпеці.

– Відстеження в реальному часі розвиток проблемної ситуації.

					ВКРБ-123.26.0046.00.00.ПЗ	Арк.
Вим.	Арк.	№ докум.	Підпис	Дата		40

– Формування повної історії рішення проблемної ситуації від її виникнення до оцінки результатів, створення бібліотеки колективного досвіду знаходження виходу з однотипних проблемних ситуацій.

3.4 Розробка діаграми процесів

Розглянемо розроблену діаграму процесів яка зображена на рисунку 3.3. Основна будова діаграми процесів полягає у графічному представленні складу сукупностей даних, що характеризуються як співвідношення різних частин кожної з сукупностей. Склад статистичної сукупності графічно може бути представлений як за допомогою абсолютних, так і відносних показників. Графічне зображення складу сукупності по абсолютними і відносними показниками сприяє проведенню більш глибокого аналізу і дозволяє проводити аналіз системи.

Діаграма взаємодії процесів використовується для візуалізації процесів обробки даних (структурне проектування).

Для розробника вважається звичним спочатку креслити діаграму взаємодії процесів даних рівня контексту, завдяки чому буде показано взаємодію системи.

Ця діаграма в подальшому підлягає уточненню шляхом деталізації процесів та потоків даних з метою показати систему що розробляється.

При детальному її розгляді можна побачити як саме проходить взаємодія у розробленій системі. Використовується модель проектування, графічне представлення «потоків» даних в інформаційній системі.

Діаграми потоків даних містять чотири типи елементів:

- Зовнішні по відношенню до системи сутності.
- Потoki даних між елементами трьох попередніх типів.
- Процеси які являють собою трансформацію даних в рамках описуваної системи.
- Сховища даних (репозиторії).

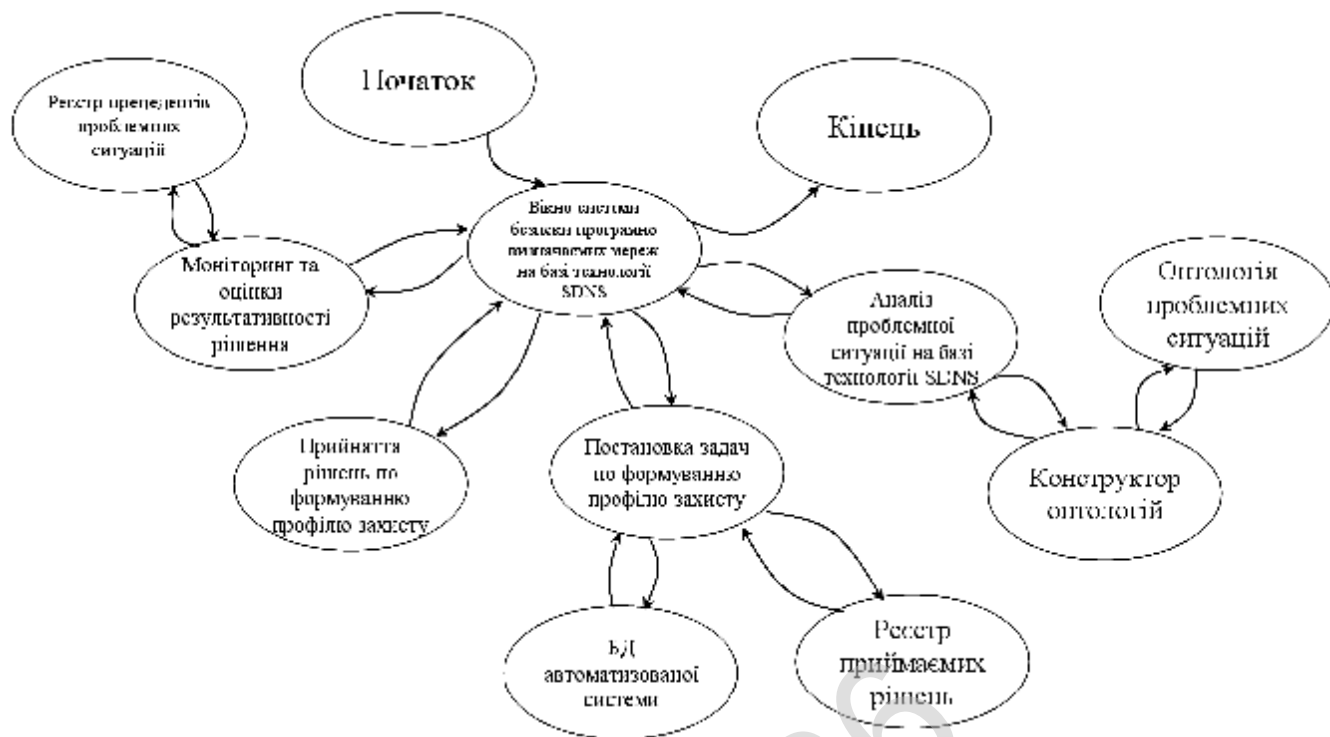


Рисунок 3.3 – Діаграма взаємодії процесів

Таким чином, розглянувши опис системи, структурну, функціональну схеми системи, та діаграму взаємодії процесів перейдемо до опису блок-схем основної програми, та підпрограм, які використовуються, для реалізації системи.

4 РЕАЛІЗАЦІЯ ПРОЕКТУ. РОЗРАХУНКИ І ЕКСПЕРИМЕНТАЛЬНІ ДАНІ, ЩО ПІДТВЕРДЖУЮТЬ ПРАВИЛЬНІСТЬ ПРОЕКТНИХ РІШЕНЬ

4.1 Блок-схеми та опис алгоритмів функціонування системи

Розглянемо реалізацію бакалаврської дипломної роботи. Були проведені розрахунки і підібрані набори тестових даних для перевірки правильності реалізації проектних рішень.

Було створено блок-схеми роботи системи. Перед їх розглядом необхідно провести роз'яснення який саме тип блок-схем використовується. Блок-схема це представлення задачі для її аналізу або розв'язування за допомогою спеціальних символів (геометричних образів), які позначають такі елементи, як операції, потік, дані тощо. Блок вхідних та вихідних даних прийнято позначати паралелограмом, блок обчислень (обробки) даних – прямокутником, блок прийняття рішень – ромбом, еліпсом – початок та кінець алгоритму.

У інформаційних технологіях функціональна схема складається з функціональних блоків, які являють собою конструктивно відособлені частини (елементи або пристрої) автоматичних систем, які виконують певні функції. Функціональні блоки на схемі позначають прямокутниками, всередині яких надписують їх найменування відповідно до функцій, що виконуються. Зв'язки між функціональними блоками (внутрішні впливи) позначаються лініями зі стрілками, які вказують напрям впливів.

Функціональні схеми можуть виконуватися в укрупненому і розгорненому вигляді. У першому випадку на схемі зображають найважливіші блоки системи і зв'язки між ними.

У другому варіанті схема відображається більш детально, що полегшує її читання та ілюструє принцип роботи.

					ВКРБ-123.26.0046.00.00.ПЗ	Арк.
Вим.	Арк.	№ докум.	Підпис	Дата		43

Основні елементи схем алгоритму це термінатор, процес, рішення, зумовлений процес (підпрограма), дані та з'єднувач. Термінатор це елемент відображає вхід із зовнішнього середовища або вихід з неї (найчастіше застосування – початок і кінець програми). Всередині фігури записується відповідна дія.

Процес це виконання однієї або кількох операцій, обробка даних будь-якого виду (зміна значення даних, форми подання, розташування). Всередині фігури записують безпосередньо самі операції. Рішення це показує рішення або функцію перемикального типу з одним входом і двома або більше альтернативними виходами, з яких тільки один може бути обраний після обчислення умов, визначених всередині цього елемента.

Вхід в елемент позначається лінією, що входить зазвичай у верхню вершину елемента. Якщо виходів два чи три то зазвичай кожен вихід позначається лінією, що виходить з решти вершин (бічних і нижній). Якщо виходів більше трьох, то їх слід показувати однією лінією, що виходить з вершини (частіше нижній) елемента, яка потім розгалужується.

Відповідні результати обчислень можуть записуватися поруч з лініями, що відображають ці шляхи. Зумовлений процес (підпрограма) це символ відображає виконання процесу, що складається з однієї або кількох операцій, що визначені в іншому місці програми (у підпрограмі, модулі).

Всередині символу записується назва процесу і передані в нього дані. Дані це перетворення у форму, придатну для обробки (введення) або відображення результатів обробки (виведення). Цей символ не визначає носія даних (для вказівки типу носія даних використовуються специфічні символи). З'єднувач це символ відображає вихід в частину схеми і вхід з іншої частини цієї схеми.

Використовується для обриву лінії та продовження її в іншому місці (приклад: поділ блок-схеми, що не поміщається на листі). Відповідні сполучні символи повинні мати одне (при тому унікальне) позначення.

Блок-схеми показують весь процес роботи системи з підсистемами та частково доказують правильність вибраних проектних рішень. Тому від точності і детальної блок-схеми залежить результат всієї програми.

При виборі початкової точки відліку при побудові схем було враховано, що виходячи з вибору мови програмування і інших технічних засобів, програма буде об'єктно-орієнтована що вимагає високого рівня декомпозиції задач на класи.

На рисунку 4.1 зображена основна блок-схема програми, на рисунку 4.2 зображено роботу підпрограми.

З яких видно що робота основної програми складається з початкових етапів ініціалізації ПЗ, перевірки наявності ресурсів системи, блоку початку основного циклу з чеканням запиту від користувача в якому відбувається виклик підсистеми та останньої стадії – перевірка поточного стану з завершенням роботи розробленого ПЗ. При роботі підпрограми виконується основний функціонал системи з циклічними послідовностями, перевіркою поточного стану та поверненням в основну програму прапорів стану виконання.

Система керування базами даних (СКБД, Database Management System, DBMS) – набір взаємопов'язаних даних (база даних) і програм для доступу до цих даних. Надає можливості створення, збереження, оновлення та пошуку інформації в базах даних з контролем доступу до даних. Першим поколінням СКБД прийнято вважати ієрархічні й мережеві системи. Ці системи отримали широке поширення в 1970-х роках, а першою комерційною системою цього типу була система IMS компанії IBM. У 1980-х роках ці системи були витіснені системами другого покоління – повсюдно використовуваними і донині реляційними СКБД. У цих системах використовувалися непроцедурні мови управління даними (SQL) і передбачався значний ступінь незалежності даних. Реляційні системи внесли значні удосконалення в управління даними: графічний користувацький інтерфейс (GUI), клієнт-серверні застосунки, розподілені бази даних, паралельний пошук даних та інтелектуальний аналіз даних.

					ВКРБ-123.26.0046.00.00.ПЗ	Арк.
Вим.	Арк.	№ докум.	Підпис	Дата		45



У 1991 був утворений консорціум ODMG, основною метою якого стало вироблення промислового стандарту об'єктно-орієнтованих баз даних. Між 1993 та 2001 роками ODMG опублікувала п'ять ревізій своїх специфікацій. Остання версія стандарту має індекс 3.0, після чого група розпустилася. До кінця 1990-х існувало близько десяти компаній, що виробляли комерційні продукти, що позиціонуються на ринку я СКБД. Найбільш відомими системами даного класу стали Objectivity, Versant виробництва однойменних компаній, а також СКБД

Jasmine, випущена компанією CA. Незважаючи на переваги, що дозволяють ефективніше вирішувати певний ряд завдань, об'єктно-орієнтовані системи так і не змогли завоювати значущу частку ринку СКБД, залишившись «нішевим» продуктом.

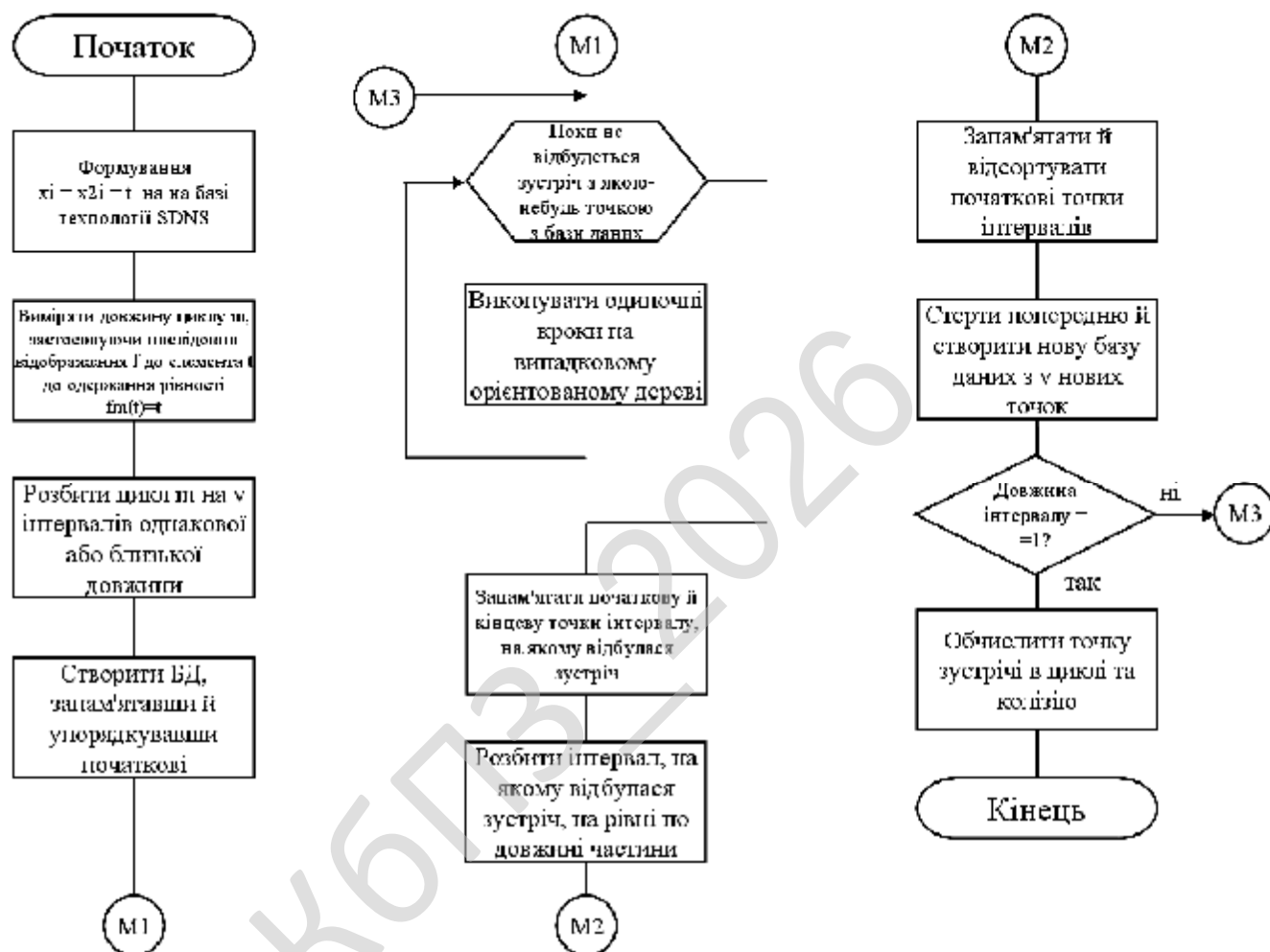


Рисунок 4.2 – Блок-схема роботи підпрограми

Постачальниками традиційних реляційних СКБД також була проведена значна робота з об'єднання об'єктно-орієнтованих і реляційних систем. Розробники постаралися розширити мову SQL, щоб включити в неї концепції об'єктно-орієнтованого підходу, зберігаючи переваги реляційної моделі (об'єктні розширення мови SQL були зафіксовані в стандарті SQL:1999).

Основний принцип – це еволюційний розвиток можливостей СКБД без поломки попередніх підходів та зі збереженням наступності з системами попереднього покоління.

Поняття СКБД третього покоління, якими, власне кажучи, і є об'єктно-реляційні СКБД, з'явилося після опублікування групою відомих фахівців в області баз даних «Маніфесту систем баз даних третього покоління». Основні принципи СКБД третього покоління, позначені в маніфесті:

Крім традиційних послуг з управління даними, СКБД третього покоління повинні забезпечити підтримку розвиненіших структур об'єктів і правил. Розвинутіша структура об'єктів характеризує засоби, необхідні для зберігання і маніпулювання нетрадиційними елементами даних (тексти, просторові дані, мультимедіа).

СКБД третього покоління повинні включити в себе СКБД другого покоління. Системи другого покоління внесли вирішальний вклад у двох областях – непроцедурний доступ за допомогою мови запитів SQL і незалежність даних. Ці досягнення обов'язково повинні враховуватися в системах третього покоління.

СКБД третього покоління повинні бути відкриті для інших підсистем. Це включає оснащення різноманітними інструментами підтримки прийняття рішень, доступом з багатьох мов програмування, інтерфейсами до існуючих популярних систем і бізнес-застосунків, можливістю запуску програм з бази даних на іншій машині і розподілені СКБД. Весь набір інструментів і СКБД має ефективно функціонувати на різноманітних апаратних платформах з різними операційними системами.

Крім того, СКБД, що розраховує на широку сферу застосування, повинна бути оснащена мовою четвертого покоління (4GL).

У середині 1990 років було лише кілька дослідних прототипів СКБД, які поєднали найкращі риси реляційних і об'єктно-орієнтованих СКБД. Першим комерційним продуктом, якому були властиві об'єктно-реляційні риси, став Universal Server компанії Informix(згодом була поглинена IBM). В даний час

більшість цих ідей вже втілено в реальних комерційних рішеннях, в тому числі і в продуктах основних постачальників СКБД (Oracle Database і IBM DB2).

Розвиток індустрії систем керування базами даних базується на значних фундаментальних наукових дослідженнях. Найчастіше, між самими дослідженнями та їхньою конкретною реалізацією в прикладних рішеннях минають роки, а іноді й десятиліття. Роботу в області управління даними проводять як університетські дослідницькі групи (MIT, Berkeley), так і центри розробок основних постачальників СКБД (Oracle, IBM, Microsoft). Інвестування в управління даними – це довгострокове, і разом з тим, вигідне вкладення коштів. В даний час дослідники мають у своєму розпорядженні засоби, що дозволяють ефективно реалізувати найскладніші запити, що маніпулюють терабайтами й петабайтами різних даних.

Основними тенденціями, які дали привід для проведення різних масштабних досліджень в області баз даних стали:

Експонентний ріст даних. Обсяг даних, у тому числі синтетичних, що генеруються автоматизованими системами, значно зріс. Збільшилося і число прикладних областей, в яких вимагається обробка великих обсягів даних. До таких областей тепер відносяться не тільки традиційні корпоративні програми, пошук у веб, але також і наукові дослідження, обробка природних мов, аналіз соціальних мереж тощо. Значне ускладнення структур використовуваних даних. Прості види даних у вигляді чисел і символьних рядків стали доповнятися численною мультимедійною інформацією, просторовими, процедурними даними та великою кількістю інших складних форматів. Широке поширення дешевих високопродуктивних апаратних засобів. Щорічно ми спостерігаємо зростання обчислювальних можливостей мікропроцесорів, збільшення ємності і зниження вартості доступних і зручних в експлуатації пристроїв дискової оперативної пам'яті.

Активний розвиток засобів комунікації та «всесвітньої павутини» World Wide Web. WWW стає єдиним інформаційним середовищем, що пронизує весь світ і об'єднує величезне число користувачів та електронних пристроїв.

					ВКРБ-123.26.0046.00.00.ПЗ	Арк.
Вим.	Арк.	№ докум.	Підпис	Дата		49

Поява нових важливих областей застосування СКБД. У першу чергу, це пов'язано з інтелектуальним аналізом даних, сховищами даних, а останнім часом – з паралельними обчисленнями і хмарними технологіями.

Основні характеристики СКБД

1. Контроль за надлишковістю даних.
2. Несуперечливість даних.
3. Підтримка цілісності бази даних (коректність та несуперечливість).
4. Цілісність описується за допомогою обмежень.
5. Незалежність прикладних програм від даних.
6. Спільне використання даних.
7. Підвищений рівень безпеки.

Можливості СКБД

1. Дозволяється створювати БД (здійснюється за допомогою мови визначення даних DDL (Data Definition Language)).

2. Дозволяється додавання, оновлення, видалення та читання інформації з БД (за допомогою мови маніпулювання даними DML, яку часто називають мовою запитів)

3. Можна надавати контрольований доступ до БД за допомогою: Системи забезпечення захисту, яка запобігає несанкціонованому доступу до БД; Системи керування паралельною роботою прикладних програм, яка контролює процеси спільного доступу до БД; Система відновлення – дозволяє відновлювати БД до попереднього несуперечливого стану, що був порушений в результаті збою апаратного або програмного забезпечення.

Основні компоненти середовища СКБД

1. Апаратне забезпечення.
2. Програмне забезпечення.
3. Дані.
4. Процедури – інструкції та правила, які повинні враховуватись при проектуванні та використанні БД.

5. Користувачі: адміністратори даних (керування даними, проектування БД, розробка алгоритмів, процедур) та БД (фізичне проектування, відповідальність за безпеку та цілісність даних); розробники БД; прикладні програмісти; кінцеві користувачі.

Архітектура СКБД

Існує трирівнева система організації СКБД ANSI-SPARC, при якій існує незалежний рівень для ізоляції програми від особливостей представлення даних на нижчому рівні.

Рівні:

1. Зовнішній – представлення БД з точки зору користувача.
2. Концептуальний – узагальнене представлення БД, описує які дані зберігаються в БД і зв'язки між ними. Підтримує зовнішні представлення, підтримується внутрішнім рівнем.

3. Внутрішній – фізичне представлення БД в комп'ютері.

Логічна незалежність – повна захищеність зовнішніх моделей від змін, що вносяться в концептуальну модель.

Фізична незалежність – захищеність концептуальної моделі від змін, які вносяться у внутрішню модель.

Microsoft SQL Server – комерційна система керування базами даних, що розповсюджується корпорацією Microsoft.

Мова, що використовується для запитів – Transact-SQL, створена спільно Microsoft та Sybase.

Transact-SQL є реалізацією стандарту ANSI/ISO щодо структурованої мови запитів SQL із розширеннями. Використовується як для невеликих і середніх за розміром баз даних, так і для великих баз даних масштабу підприємства. Багато років вдало конкурує з іншими системами керування базами даних.

Походження

Базовий код MS SQL Server (до версії 7.0) ґрунтувався на коді Sybase SQL Server. Це дозволило Microsoft вийти на ринок баз даних для підприємств, де

конкурували Oracle, IBM, і, пізніше, сама Sybase. Microsoft, Sybase і Ashton-Tate спочатку об'єдналися для створення і випуску на ринок першої версії програми, що отримала назву SQL Server 1.0 для OS/2 (близько 1989 року), яка фактично була еквівалентом Sybase SQL Server 3.0 для Unix, VMS та ін. Microsoft SQL Server 4.2 був випущений у 1992 році та входив до складу операційної системи Microsoft OS/2 версії 1.3.

Офіційний реліз Microsoft SQL Server версії 4.21 для ОС Windows NT відбувся одночасно з релізом самої Windows NT (версії 3.1). Microsoft SQL Server 6.0 був першою версією SQL Server, створеною виключно для архітектури NT і без участі в процесі розробки Sybase.

До того часу, як вийшла на ринок ОС Windows NT, Sybase і Microsoft розійшлися та створювали вже власні моделі цього програмного продукту. Microsoft намагалася отримати виняткові права на всі версії SQL Server для Windows. Пізніше Sybase змінила назву свого продукту на Adaptive Server Enterprise щоб уникнути плутанини з Microsoft SQL Server. До 1994 року Microsoft отримала від Sybase три повідомлення про авторські права як натяк на походження Microsoft SQL Server.

Після розділення компанії зробили декілька самостійних релізів програм. SQL Server 7.0 був першим сервером баз даних зі справжнім графічним інтерфейсом адміністрування. Для усунення претензій з боку Sybase у порушенні авторських прав, весь успадкований код в сьомій версії був переписаний. Це забезпечило також й успіх SQL Server 2000, який був першою редакцією, орієнтованою на архітектуру IA-64.

Протягом подальших шести років корпорація Microsoft працювала над вдосконаленням вже існуючої версії SQL Server 2000 доки не збудувала зручнішу систему Microsoft SQL Server 2005. Були вдосконалені продуктивність, клієнтські інструменти інтегрованого середовища розробки, а також у декількох додаткових системах, що встановлюються разом із SQL Server 2005. Змінено: інструментарій процесів керування сховищами даних (SQL Server Integration Services або SSIS),

					ВКРБ-123.26.0046.00.00.ПЗ	Арк.
Вим.	Арк.	№ докум.	Підпис	Дата		52

сервер звітів, сервер OLAP та інтелектуального аналізу даних (Analysis Services), а також декілька технологій повідомлень, особливо Service Broker та Notification Services.

SQL Server 2025

SQL Server 2025 включає низку вдосконалень для роботи з критичними бізнес-застосунками і бізнес-аналітикою як в традиційних дата-центрах, так і в приватних, публічних і гібридних хмарних середовищах. Серед нових можливостей SQL Server 2025 виділяються SQL Server AlwaysOn (рішення підтримки високого рівня доступності даних та аварійного відновлення), xVelocity (технологія збільшення продуктивності сховищ даних та програм бізнес-аналітики), нові рішення в області візуалізації PowerPivot і PowerView для створення звітів і аналітичних програм з Excel і SharePoint, покращені інструменти для інтеграції даних і управління ними, включаючи SQL Server Data Quality Services і Master Data Services, нова семантична модель бізнес-аналітики та інструмент для адміністраторів баз даних і розробників застосунків SQL Server Data Tools.

Також Microsoft зробила значні інвестиції в області Big Data, а саме в інтеграцію SQL Server і популярних інструментів для бізнес-аналітики з неструктурованою інформацією. Microsoft робить SQL Server доступним у різноманітних варіантах, які різняться наборами властивостей в залежності від цілей кінцевого користувача. Це такі редакції як:

SQL Server Compact Edition (SQL CE) – Компактний механізм бази даних. Завдяки малому обсягу (2 Мб для DLL) має зменшені властивості у порівнянні з іншими варіантами. Розмір бази даних обмежений 4 Гб і не може використовуватися як служба Windows.

SQL Server Express Edition – Раніше відомий під назвою MSDE (Microsoft SQL Server Desktop Engine), Microsoft SQL Server Express є вільно поширюваною версією SQL Server. Дана версія має деякі технічні обмеження, також відсутні графічні інструменти адміністрування. Такі обмеження роблять її непридатною

					ВКРБ-123.26.0046.00.00.ПЗ	Арк.
Вим.	Арк.	№ докум.	Підпис	Дата		53

для розгортання великих баз даних. В основному вона використовується у застосунках, при проектуванні, або для самостійного вивчення. Розмір бази даних обмежений 4 Гб, розмір пам'яті, що може бути адресованою—1 Гб, підтримує лише один процесор.

SQL Server Workgroup Edition – SQL Server Workgroup Edition включає функціональність ядра бази даних, але не включає додаткові сервіси.

SQL Server Standard Edition – SQL Server Standard edition включає механізми ядра бази даних, а також автономні сервіси. Відрізняється від варіанту Enterprise Edition тим, що підтримує менше активних вузлів та не включає деякі функції збільшення продуктивності.

SQL Server Enterprise Edition – SQL Server Enterprise Edition—це повнофункціональна версія SQL Server

SQL Server Developer Edition – SQL Server Developer Edition включає ті самі функції, що й SQL Server Enterprise Edition, але містить обмеження щодо використання його лише для розробки та тестування.

Його ліцензія не дозволяє використання як виробничого сервера.

Функціональність

Microsoft SQL Server як мову запитів використовує версію SQL, що отримала назву Transact-SQL (скорочено T-SQL), яка є реалізацією SQL-92 (стандарт ISO для SQL) з багатьма розширеннями. T-SQL дозволяє використовувати додатковий синтаксис процедур, що зберігаються і забезпечує підтримку транзакцій (взаємодія бази даних з керуючим застосунком).

Microsoft SQL Server та Sybase ASE для взаємодії з мережею використовують протокол рівня застосунка під назвою Tabular Data Stream (TDS, протокол передачі табличних даних).

Microsoft SQL Server також підтримує Open Database Connectivity (ODBC)—інтерфейс взаємодії застосунків з СУБД. Версія SQL Server 2005 надає можливість підключення користувачів через веб-сервер-сервіси, що використовують протокол SOAP. Це дозволяє клієнтським програмам, не

					ВКРБ-123.26.0046.00.00.ПЗ	Арк.
Вим.	Арк.	№ докум.	Підпис	Дата		54

призначеним для Windows, кроссплатформенно з'єднуватися з SQL Server. Microsoft також випустила сертифікований драйвер JDBC, що дозволяє застосункам під керування Java (таким як BEA і IBM Websphere) з'єднуватися з Microsoft SQL Server 2000 і 2005.

SQL Server підтримує дзеркалювання та кластеризацію баз даних. Кластер серверу SQL—це сукупність однаково конфігурованих серверів; така схема допомагає розподілити робоче навантаження між декількома серверами. Усі сервери мають одне віртуальне ім'я, а дані розподіляються за IP-адресами машин кластеру протягом робочого циклу. Також у разі відмови або збою на одному з серверів кластеру доступне автоматичне перенесення навантаження на інший сервер.

SQL Server підтримує надлишкове дублювання даних за трьома сценаріями:

1. Знімок: Виконується «знімок» бази даних, який сервер відправляє одержувачам.

2. Історія змін: Всі зміни бази даних безперервно передаються користувачам.

3. Синхронізація з іншими серверами: Бази даних декількох серверів синхронізуються між собою. Зміни усіх баз даних відбуваються незалежно на кожному сервері, а під час синхронізації відбувається звірка даних. Дублювання такого типу передбачає можливість вирішення протиріч між базами даних.

SQL Server 2005 має вбудовану підтримку .NET Framework. Завдяки цьому, процедури бази даних, що зберігаються, можуть бути написані на будь-якій мові платформи .NET з використанням повного набору бібліотек, доступних для .NET Framework. На відміну від інших процесів, .NET Framework виділяє додаткову пам'ять і будує засоби керування SQL Server, не використовуючи вбудовані засоби Windows. Це підвищує продуктивність порівняно із загальними алгоритмами Windows, оскільки алгоритми розподілу ресурсів спеціально налагоджені для використання у структурах SQL Server.

					ВКРБ-123.26.0046.00.00.ПЗ	Арк.
Вим.	Арк.	№ докум.	Підпис	Дата		55

При розробці ПЗ було використано підходи ризик-менеджменту – це система управління ризиками, яка включає в себе стратегію та тактику управління, направлені на досягнення основних цілей. Ефективний ризик-менеджмент включає:

- систему управління;
- систему ідентифікації і вимірювання;
- систему супроводження (моніторингу та контролю).

Сучасна наука представляє ризик як вірогідну подію, в результаті настання якої можуть відбутися позитивні, нейтральні або негативні наслідки. Якщо ризик припускає наявність як позитивних, так і негативних результатів, він відноситься до спекулятивних ризиків. Якщо ж наслідки негативні, або відсутні взагалі, такий ризик іменується чистим.

Мета ризик-менеджменту – підвищення конкурентоспроможності господарюючих суб'єктів за допомогою захисту від реалізації чистих ризиків.

Теорія ризик-менеджменту ґрунтується на трьох базових поняттях: корисності, регресії і диверсифікації.

У 1738 швейцарський математик Даніель Бернуллі доповнив теорію вірогідності методом корисності або привабливості того або іншого результату подій. Ідея Бернуллі полягала в тому, що в процесі ухвалення рішення люди приділяють більше уваги розміру наслідків різних результатів, ніж їх вірогідність.

В кінці XIX століття англійський дослідник Ф. Гальтон запропонував вважати регресію або повернення до середнього значення універсальною статистичною закономірністю. Суть регресії трактувалася ним як повернення явищ до норми з часом. Згодом було доведено, що правило регресії діє в найрізноманітніших ситуаціях, починаючи з азартних ігор та розрахунку вірогідності виникнення нещасних випадків, і закінчуючи прогнозуванням коливань економічних циклів.

У 1952 аспірант Університету Чикаго Гарі Марковіц в статті «Диверсифікація вкладень» («Portfolio Selection») математично обґрунтував

стратегію диверсифікації інвестиційного портфеля, зокрема, він показав, як шляхом продуманого розподілу вкладень мінімізувати відхилення прибутковості від очікуваного показника. У 1990 Г. Марковіцу присуджена Нобелівська премія за розробку теорії і практики оптимізації портфеля фондових активів.

Етапи ризик-менеджменту

У ризик-менеджменті прийнято виділяти декілька ключових етапів:

- на першому етапі відбувається виявлення ризику з супутньою оцінкою вірогідності його реалізації і масштабу наслідків;

- на другому етапі здійснюється розробка ризик-стратегії з метою зниження вірогідності реалізації ризику і мінімізації можливих негативних наслідків;

- на третьому етапі вибираються методи і інструменти управління виявленим ризиком;

- на четвертому етапі проводиться безпосереднє управління ризиком;

- на завершальному етапі оцінюються досягнуті результати і коректується ризик-стратегія.

За ключовий етап ризик-менеджменту вважається етап вибору методів і інструментів управління ризиком.

Методи і інструментарій ризик-менеджменту

Базовими методами ризик-менеджменту є відмова від ризиків, зниження, передача і ухвалення.

Ризик-інструментарій значно ширший. Він включає політичні, організаційні, правові, економічні, соціальні інструменти, причому ризик-менеджмент як система допускає можливість одночасного застосування декількох методів і інструментів ризик-управління.

Найбільш часто вживаним інструментом ризик-менеджменту є страхування. Страхування припускає передачу відповідальності за відшкодування передбачуваного збитку сторонній організації (страхової компанії). Прикладами інших інструментів можуть бути відмова від надмірно ризикової діяльності

					ВКРБ-123.26.0046.00.00.ПЗ	Арк.
Вим.	Арк.	№ докум.	Підпис	Дата		57

(метод відмови), профілактика або диверсифікація (метод зниження), аутсорсинг витратних ризикових функцій (метод передачі), формування резервів або запасів (метод ухвалення).

Незважаючи на те що я працював над ПЗ один в реалізації програми я використовував підходи пришвидшення розробки на основі методологій Agile.

Гнучка розробка програмного забезпечення (Agile software development, agile-методи) – клас методологій розробки програмного забезпечення, що базується на ітеративній розробці, в якій вимоги та розв'язки еволюціонують через співпрацю між самоорганізовуваними багатофункціональними командами.

Гнучка розробка – найкращий засіб для підвищення продуктивності розробників програмного забезпечення.

Більшість гнучких методологій націлені на мінімізацію ризиків, шляхом зведення розробки до серії коротких циклів, що мають назву ітерацій, які зазвичай тривають один-два тижні. Кожна ітерація сама по собі виглядає як програмний проект в мініатюрі, і включає всі завдання, необхідні для видачі мінімального приросту за функціональністю: планування, аналіз вимог, проектування, кодування, тестування і документування. Хоча окрема ітерація, як правило, недостатня для випуску нової версії продукту, мається на увазі те, що гнучкий програмний проект готовий до випуску наприкінці кожної ітерації. Після закінчення кожної ітерації, команда виконує переоцінку пріоритетів розробки.

Agile акцентує увагу на безпосередньому спілкуванні «віч-на-віч». Більшість agile команд розташовані в одному офісі, його іноді називають bullpen. Як мінімум вона включає і «замовників» (замовники, які визначають продукт, також це можуть бути менеджери продукту, бізнес аналітики або клієнти). Офіс може також включати тестувальників, дизайнерів інтерфейсу, технічних авторів і менеджерів. Основною метрикою agile методів є робочий продукт. Віддаючи перевагу безпосередньому спілкуванню, agile-методи зменшують обсяг письмової документації в порівнянні з іншими методами. Це привело до критики цих методів як недисциплінованих.

					ВКРБ-123.26.0046.00.00.ПЗ	Арк.
Вим.	Арк.	№ докум.	Підпис	Дата		58

Agile – родина процесів розробки, а не єдиний підхід в розробці програмного забезпечення, і визначається Agile Manifesto. Agile не включає практик, а визначає цінності та принципи, якими керуються успішні команди.

Agile Manifesto розроблений і прийнятий 17 розробниками 11-13 лютого 2001 року на лижному курорті The Lodge at Snowbird в горах Юти. Маніфест підписали представники наступних методологій Extreme programming, Scrum, DSDM, Adaptive software development, Crystal Clear, Feature driven development, Pragmatic Programming. Agile Manifesto містить 4 основні ідеї та 12 принципів. Примітно, що Agile Manifesto не містить практичних порад.

Основні ідеї:

- Особистості та їхні взаємодії важливіші, ніж процеси та інструменти.
- Робоче програмне забезпечення важливіше, ніж повна документація.
- Співпраця із замовником важливіша, ніж контрактні зобов'язання.
- Реакція на зміни важливіша, ніж дотримання плану.

Принципи, які роз'яснює Agile Manifesto:

- задоволення клієнта за рахунок ранньої та безперебійної поставки коштовного програмного забезпечення;
- вітання змін вимог навіть наприкінці розробки (це може підвищити конкурентоспроможність отриманого продукту);
- часта поставка робочого програмного забезпечення (кожен місяць або тиждень або ще частіше);
- тісне, щоденне спілкування замовника з розробниками впродовж всього проекту;
- проектом займаються мотивовані особистості, які забезпечені потрібними умовами роботи, підтримкою і довірою;
- рекомендований метод передачі інформації – особиста розмова (віч-на-віч);
- робоче програмне забезпечення – найкращий вимірювач прогресу;

					ВКРБ-123.26.0046.00.00.ПЗ	Арк.
Вим.	Арк.	№ докум.	Підпис	Дата		59

- спонсори, розробники та користувачі повинні мати можливість підтримувати постійний темп на невизначений термін;
- постійну увагу поліпшенню технічної майстерності та зручному дизайну;
- простота – мистецтво не робити зайвої роботи;
- найкращі технічні вимоги, дизайн та архітектура виходять у самоорганізованої команди;
- постійна адаптація до мінливих обставин.

Маніфест та Принципи гнучкої розробки містять високорівневі ідеї щодо того, як потрібно вибудовувати процес розробки програмного забезпечення, щоб успішно завершувати проекти й створювати команди, в яких приємно та цікаво працювати.

Документи визначають, що потрібно для цього зробити, але не говорять, як це зробити. По-іншому й не могло бути, оскільки Маніфест та Принципи народилися внаслідок консенсусу представників різних (хоча й споріднених) напрямів, які могли знайти спільну основу лише на рівні базових цінностей та принципів.

Критика. Багато керівників проектів, що працюють у традиційних методологіях на кшталт «водоспаду», критикують agile-методи.

Один з повторюваних пунктів критики: при agile-підході часто нехтують створенням «дорожньої карти» розвитку продукту, так само як і управлінням вимогами, в процесі якого і формується така «карта». Гнучкий підхід до управління вимогами не має на увазі далекосяжних планів (по суті, управління вимогами просто не існує в даній методології), а має на увазі можливість замовника раптом і несподівано наприкінці кожної ітерації виставляти нові вимоги, що часто суперечать архітектурі вже створеного і поставленого продукту. Таке іноді призводить до катастрофічних «авралів» з масовим рефакторингом і переробками практично на кожній черговій ітерації.

Крім того вважається, що робота в agile мотивує розробників вирішувати всі прибулі завдання найпростішим і найшвидшим можливим способом, при

					ВКРБ-123.26.0046.00.00.ПЗ	Арк.
Вим.	Арк.	№ докум.	Підпис	Дата		60

цьому часто не звертаючи уваги на коректність коду з точки зору вимог базової платформи (підхід «працює, та й добре», при цьому не враховується, що може перестати працювати при найменшій зміні або ж породити важкі до відтворення дефекти після реального розгортання у клієнта). Це призводить до зниження якості продукту і накопиченню дефектів.

Методології. Існують методології, які дотримуються цінностей і принципів заявлених в Agile Manifesto, деякі з них:

1. Agile Modeling – набір понять, принципів і прийомів (практик), що дозволяють швидко і просто виконувати моделювання і документування в проектах розробки програмного забезпечення. Не включає в себе детальну інструкцію з проектування, не містить описів, як будувати діаграми на UML.

Основна мета – ефективне моделювання і документування; але не охоплює програмування та тестування, не включає питання управління проектом, розгортання і супроводу системи. Однак включає в себе перевірку моделі кодом.

2. Agile Unified Process (AUP) спрощена версія IBM Rational Unified Process (RUP), розроблена Скоттом Амблером, яка описує просте і зрозуміле наближення (модель) для створення програмного забезпечення для бізнес-додатків.

3 Agile Data Method – група ітеративних методів розробки програмного забезпечення, в яких вимоги та рішення досягаються в рамках співпраці різних крос-функціональних команд.

4. DSDM заснований на концепції швидкої розробки додатків (Rapid Application Development, RAD). Являє собою ітеративний і інкрементний підхід, який надає особливого значення тривалій участі в процесі користувача/споживача.

5. Essential Unified Process (EssUP).

6. Екстремальне програмування (Extreme programming, XP).

7. Feature driven development (FDD) – функціонально-орієнтована розробка.

Використовуване в FDD поняття функції або властивості (feature) Системи досить

					ВКРБ-123.26.0046.00.00.ПЗ	Арк.
Вим.	Арк.	№ докум.	Підпис	Дата		61

близько до поняття прецеденту використання, використовуваному в RUP, істотна відмінність – це додаткове обмеження: «кожна функція повинна допускати реалізацію не більше, ніж за два тижні». Тобто якщо сценарій використання досить малий, його можна вважати функцією. Якщо ж великий, то його треба розбити на декілька відносно незалежних функцій.

8. Getting Real – ітераційний підхід без функціональних специфікацій, що використовується для веб-додатків. У даному методі спершу розробляється інтерфейс програми, а потім її функціональна частина.

9. OpenUP – це ітераційно-інкрементний метод розробки програмного забезпечення. Позиціюється, як легкий і гнучкий варіант RUP. OpenUP ділить життєвий цикл проекту на чотири фази: початкова фаза, фази уточнення, конструювання та передачі. Життєвий цикл проекту забезпечує надання зацікавленим особам та членам колективу точок ознайомлення і прийняття рішень впродовж усього проекту. Це дозволяє ефективно контролювати ситуацію і вчасно приймати рішення про задовільність результатів. План проекту визначає життєвий цикл, а кінцевим результатом є остаточний додаток.

10. Scrum встановлює правила керування процесом розробки та дозволяє використовувати вже існуючі практики кодування, коректуючи вимоги або вносячи тактичні зміни. Використання цієї методології дає можливість виявляти і усувати відхилення від бажаного результату на більш ранніх етапах розробки програмного продукту.

11. Бережлива розробка програмного забезпечення (lean software development). Використовує підходи з концепції бережливого виробництва.

Хоча я реалізовував програму сам, було використано підходи Scrum для саморозвитку та пришвидшенню розробки, розглянемо цей метод. Scrum – підхід управління проектами для гнучкої розробки програмного забезпечення. Скрам чітко робить акцент на якісному контролі процесу розробки.

Підхід вперше описали Гіротакі Такеучі та Ікуджіро Нонака в статті The New New Product Development Game (Гарвардський Діловий Огляд, січ–лют

					ВКРБ-123.26.0046.00.00.ПЗ	Арк.
Вим.	Арк.	№ докум.	Підпис	Дата		62

1986). Вони відзначили, що проекти, над якими працюють невеликі, крос-функціональні команди, зазвичай систематично продукують кращі результати, і пояснили це, як «підхід регбі». У 1991 році ДеҐрейс та Шталь у книжці Злі проблеми, справедливі рішення послалися на цей підхід, як на Scrum (штовханина; сутичка навколо м'яча (у регбі)), спортивний термін, згаданий в статті Такеучі і Нонака. Кен Швабер на початку 1990-х використовував підхід який привів Scrum в його компанію.

Вперше метод Scrum було представлено на загальний огляд задокументованим, чітко сформульованим та описаним спільно Сазерлендом та Швабером на OOPSLA'96 в Остіні. Швабер та Сазерленд протягом наступних років працювали разом щоб обробити та описати весь їхній досвід та найкращі практичні зразки для індустрії в одне ціле, в ту методологію, що відома сьогодні як Scrum. Швабер об'єднав зусилля з Майком Бідлом в 2001, щоб детально описати метод в книжці Agile Software Development with SCRUM. Не зважаючи на те, що для Scrum нарікли долю управління проектами з розробки ПЗ, він може також використовуватися в роботі команд обслуговувань програмного забезпечення (software maintenance teams), або як підхід управління розробкою і супроводом програм: Scrum of Scrums.

Scrum – це кістяк процесу, який включає набір методів і попередньо визначених ролей. Головні дійові особи – ScrumMaster, той хто опікується процесами, веде їх і працює як керівник проекту, Власник Продукту, людина, що представляє інтереси кінцевих користувачів та інших зацікавлених в продукті сторін, та Команду, яка включає розробників.

Протягом кожного спринту, 15-30 денного періоду (тривалість визначається командою), працівники створюють функціональний ріст програмного забезпечення. Набір можливостей, які імплементуються кожного спринту, приходять з етапу, що має назву product backlog (документація запитів на виконання робіт), який має найвищу пріоритетність за рівнем вимог до роботи, що повинна бути виконана.

					ВКРБ-123.26.0046.00.00.ПЗ	Арк.
Вим.	Арк.	№ докум.	Підпис	Дата		63

Запити на виконання робіт (backlog items), що визначені протягом наради з планування спринту (sprint planning meeting), переміщуються в етап спринту. Протягом цієї наради Власник Продукту інформує про завдання, які він хоче, аби були виконані. Тоді Команда визначає, скільки з бажаного вони можуть зробити, щоб завершити необхідні частини протягом наступного спринту. Протягом спринту команда виконує визначений фіксований список завдань (т.з. backlog items). Впродовж цього періоду ніхто не має права змінювати перелік запитів на виконання робіт, що слід розуміти, як заморожування вимог (requirements) протягом спринту.

Product backlog – це документ, який має список вимог до функціональності, які упорядковані згідно зі ступенем важливості. Product backlog представляє список того, що повинно бути реалізовано. Елементи цього списку називається «історіями» (user story) або елементами backlog-у (backlog items). Product backlog відкритий для редагування усім учасникам Scrum-процесу.

Обов'язкові поля:

1. ID – унікальний ідентифікатор, порядковий номер, який використовується для ідентифікації історій у разі їх перейменування.

2. Назва (Name) – стислий опис історії. Він повинен бути однозначним, щоб і розробники і product owner могли зрозуміти, про що йдеться і відрізнити одну історію від іншої.

3. Важливість (Importance) – ступінь важливості даної історії на погляд product owner 'а. Зазвичай являє собою натуральне число, іноді для цієї цілі використовуються числа Фібоначчі. Чим більше значення, тим більше пріоритет.

4. Попередня оцінка (initial estimate) – початкова оцінка об'єму робіт, необхідного для реалізації історії порівняно з іншими історіями. Вимірюється у story point'ах. Приблизно відповідає числу «ідеальних людино-днів».

5. Як продемонструвати (how to demo) – стисле пояснення того, як завершена задача буде продемонстрована у кінці спринта. Дане поле може являти собою код автоматизованого приймального тесту.

					ВКРБ-123.26.0046.00.00.ПЗ	Арк.
Вим.	Арк.	№ докум.	Підпис	Дата		64

Додаткові поля. Іноді, також, використовуються додаткові поля у product backlog, в основному для того, щоб допомогти product owner'у визначитися з його пріоритетами.

Категорія (track). Наприклад, «панель управління» чи «оптимізація». За допомогою цього поля product owner може легко вибрати усі пункти категорії «оптимізація» і задати їм низький пріоритет.

Компоненти (components) – указує, які компоненти (наприклад, база даних, сервер, клієнт) будуть зачеплені при реалізації історії. Дане поле складається з групи checkbox'ів, які відмічаються, якщо відповідні компоненти потребують змін.

Ініціатор запиту (requestor). Product owner може захотіти зберігати інформацію про усіх замовників, зацікавлених у даній задачі. Це потрібно для того, щоб тримати їх у курсі діла про хід виконання робіт.

ID у системі обліку помилок (bug tracking ID) – якщо ви використовуєте окрему систему обліку помилок, тоді у описі історії корисно зберігати посилання на всі дефекти, які до неї відносяться.

Sprint backlog – містить функціональність, обрану Product Owner із Product Backlog. Всі функції розбиті по задачах, кожна з яких оцінюється командою. Кожен день команда оцінює об'єм роботи, який необхідно провести для завершення задачі.

Burndown chart – показує, скільки вже виконано і скільки ще залишається зробити.

Планування спринта (Sprint Planning Meeting)

Проходить на початку нової ітерації Спринта:

– Із Product Backlog обираються задачі, зобов'язання по виконанню яких за спринт приймає на себе команда;

– На основі обраних задач створюється Sprint Backlog. Кожна задача оцінюється у ідеальних людино-годинах;

- Рішення задачі не повинно займати більше 12 годин або одного дня. При необхідності задача розбивається на підзадачі;
- Обговорюється та визначається, яким чином буде реалізовано цей об'єм робіт;
- Тривалість наради обмежена зверху 4–8 годинами в залежності від тривалості ітерації, досвіду команди тощо;
- (перша частина наради) Беруть участь Product Owner + Команда: обирають задачі із Product Backlog;
- (друга частина наради) Бере участь лише команда: обговорюють технічні деталі реалізації, наповнюють Sprint Backlog.

Щоденна нарада (Daily Scrum Meeting)

Відбувається кожен день протягом спринта. Є «пульсом» ходу спринта. Нараді властиві наступні обмеження:

- починається точно вчасно;
- всі можуть спостерігати, але говорять тільки обрані;
- триває не більш ніж 15 хвилин;
- проводиться в одному і тому ж місці протягом одного спринта.

Протягом наради кожен член команди відповідає на 3 запитання:

- Що зроблено з моменту попередньої щоденної наради?;
- Що буде зроблено з моменту поточної наради до наступної?;
- Які проблеми заважають досягненню цілей спринта? (Над рішенням цих

проблем працює ScrumMaster. Зазвичай це рішення проходить за рамками щоденної наради і у складі осіб, що безпосередньо займаються даною перешкодою.)

Демонстрація (Sprint Review Meeting):

- Проходить у кінці ітерації (спринта).
- Команда демонструє внесок функціональності до продукту всім зацікавленим особам.
- Залучається максимальна кількість глядачів.

					ВКРБ-123.26.0046.00.00.ПЗ	Арк.
Вим.	Арк.	№ докум.	Підпис	Дата		66

- Усі члени команди беруть участь у демонстрації (одна людина на демонстрацію або кожен показує, що зробив за спринт).
- Обмежена 4–ма годинами в залежності від тривалості ітерації і змін у продукті.

Ретроспектива (Sprint Retrospective):

- Члени команди висловлюють свою думку про минулий спринт.
- Відповідають на два основних запитання: Що було зроблено добре у минулому спринті?; Що потрібно покращити в наступному?.
- Виконують покращення процесу розробки (вирішують питання та фіксують вдалі рішення).
- Обмежена 1–3ма годинами.

Розглянемо визначення API. Це прикладний програмний інтерфейс (Application Programming Interface, API) – набір визначень підпрограм, протоколів взаємодії та засобів для створення програмного забезпечення.

Спрощено – це набір чітко визначених методів для взаємодії різних компонентів. API надає розробнику засоби для швидкої розробки програмного забезпечення. API може бути для веб-базованих систем, операційних систем, баз даних, апаратного забезпечення, програмних бібліотек.

Одним з найпоширеніших призначень API є надання набору широко використовуваних функцій, наприклад для малювання вікна чи іконок на екрані. API є абстрактним поняттям – програмне забезпечення, що пропонує деякий API, часто називають реалізацією (implementation) даного API. У багатьох випадках API є частиною набору розробки програмного забезпечення, водночас, набір розробки може включати як API, так і інші інструменти/апаратне забезпечення, отже ці два терміни не є взаємозамінювані.

Високорівневі API часто програють у гнучкості. Виконання деяких функцій нижчого рівня стає набагато складнішим, або навіть неможливим.

В об'єктно-орієнтованих мовах, прикладний програмний інтерфейс зазвичай включає в себе опис набору визначень класу, з набором форм поведінки,

					ВКРБ-123.26.0046.00.00.ПЗ	Арк.
Вим.	Арк.	№ докум.	Підпис	Дата		67

пов'язаних з цими класами. Це абстрактне поняття пов'язане з реальними функціями, які надані або надаватимуться, класами, які реалізуються в методах класу.

Прикладний програмний інтерфейс в даному випадку можна розглядати як сукупність всіх методів, які публічно доступні в класах (зазвичай званий інтерфейс класу). Це означає, що прикладний програмний інтерфейс вказує методи, за допомогою яких взаємодіє з об'єктами, отриманими з визначень класів і обробляє їх.

У більш загальному плані можна визначити Прикладний Програмний Інтерфейс як сукупність усіх видів об'єктів, які можна вивести з визначення класу, і пов'язаних з ними можливих варіантів поведінки.

Наприклад: клас, що представляє Stack, може просто виставити публічно два методи Push() (для додавання нового елемента в стек) і Pop() (для вилучення останнього пункту, ідеально розташований на вершині стека).

У цьому випадку Прикладний Програмний Інтерфейс може бути інтерпретованим як два методи pop() і push(), або, більш широко, використовується варіант, коли можна використовувати елемент типу Stack, який реалізує поведінку стека, надаючи йому можливість для додавання / видалення елементів з вершини. Друга інтерпретація видається більш доречною в дусі об'єктно-орієнтованого підходу.

Якість документації, пов'язаної з Прикладним Програмним Інтерфейсом, є часто ключовим фактором, що визначає його успішність з точки зору простоти використання.

ППІ, як правило, пов'язаний із бібліотеками програмного забезпечення: ППІ описує і вказує очікувану поведінку в той час, як бібліотека є фактичною реалізацією даного набору правил. Один ППІ може мати декілька реалізацій (або жодної, будучи абстрактним) у вигляді різних бібліотек, які мають такий же інтерфейс.

					ВКРБ-123.26.0046.00.00.ПЗ	Арк.
Вим.	Арк.	№ докум.	Підпис	Дата		68

Прикладний програмний інтерфейс також може бути пов'язаним з платформами програмування: платформа може бути заснована на кількох бібліотеках реалізує декілька інтерфейсів ППІ, але на відміну від звичайного використання ППІ, доступ до поведінки вбудований в платформу опосередкований шляхом розширення його змісту новими класами і вставлений в саму платформу. Крім того, загальний потік управління програми може бути під контролем абонента.

Прикладний програмний інтерфейс може бути також реалізацією протоколу.

Коли ППІ реалізує протокол, він може бути заснованим на проксі-методах віддалених викликів, що засновані на протоколі зв'язку. Роль ППІ може заключатися саме в тому, щоб приховати деталі транспортного протоколу. Наприклад: RMI є ППІ, який реалізує протокол або JRMP IIOP як RMI-IIOP.

Протоколи, як правило, розподіляються між різними технологіями і зазвичай дозволяють різним технологіям обмінюватися інформацією, діючи як абстракція між двома світами. ППІ, як правило, є специфічним для конкретної технології: звідси, інтерфейси даної мови не можуть бути використані на інших мовах, якщо виклики функції не будуть перетворені з конкретної адаптації бібліотеки.

Деякі мови, серед яких такі, що працюють на віртуальних машинах (наприклад: мови, сумісні з NET CLI середовища CLR і JVM сумісних мов у віртуальній машині Java) можуть ділитися програмними інтерфейсами.

У цьому випадку віртуальна машина дозволяє мові взаємодії завдяки спільному знаменнику віртуальної машини, що абстрагується від конкретної мови, використовувати проміжний байт-код і його мову.

При використанні прикладного програмного інтерфейсу в контексті веб-розробки, як правило, ППІ визначається набором повідомлень запиту HTTP, також визначається структура повідомлень-відповідей, зазвичай у розширенні мови розмітки XML або в форматі об'єктного запису JavaScript (JSON). У той час як прикладний програмний інтерфейс у Web історично був практично синонімом

					ВКРБ-123.26.0046.00.00.ПЗ	Арк.
Вим.	Арк.	№ докум.	Підпис	Дата		69

для веб-служби, останнім часом тенденція змінилась (так званий Web 2.0) на відхід від Simple Object Access Protocol (SOAP) на основі веб-сервісів і сервіс-орієнтованої архітектури (SOA) на більш прямі передачі репрезентативного стану (REST) стилів веб-ресурсів та ресурсів-орієнтованої архітектури (ROA).

Частина цієї тенденції пов'язана з рухом Семантичного веб-ресурсу до Опису Платформ (RDF), Концепції розвитку веб-технологій інженерних онтологій. Прикладні програмні інтерфейси у Web, що дозволяють комбінувати декількома прикладними програмними інтерфейсами в нові додатки називають гібридними.

4.2 Захист розробленого програмного забезпечення

Для захисту розробленого програмного забезпечення запропоновано використовувати алгоритм RC5, який являє собою блоковий шифр із безліччю параметрів: розміром блоку, розміром ключа й числом раундів. В алгоритмі RC5 передбачені три операції: XOR, додавання й циклічні зрушення. На більшості процесорів операції циклічного зрушення виконуються за постійний час, змінні циклічні зрушення являють собою нелінійну функцію. Циклічні зрушення залежать як від ключа, так і від даних.

В RC5 використовується блок змінної довжини, але в приводиться прикладі, що буде розглянутий, 64-бітовий блок даних. Шифрування використовує $2r+2$ залежних від ключа 32-бітових слів – $S_0, S_1, S_2, \dots, S_{2r+1}$ – де r – число раундів. Для шифрування спочатку потрібно розділити блок відкритого тексту на два 32-бітових слова: A и B . (При впакуванні байтів у слова в алгоритмі RC5 дотримується угода про прямий порядок (little-endian) байтів: перший байт займає молодші біти регістра A й т. ін.) Потім:

$$A = A + S_0$$

$$B = B + S_0$$

Для i від 1 до r :

$$A = ((A \oplus B) \lll B) + S_{2i}$$

$$B = ((B \oplus A) \lll A) + S_{2i+1}$$

					ВКРБ-123.26.0046.00.00.ПЗ	Арк.
Вим.	Арк.	№ докум.	Підпис	Дата		70

Вихід перебуває в регістрах A і B .

Розшифрування теж нескладно. Потрібно розбити блок відкритого тексту на два слова, A й B , а потім:

Для i від r до 1 із кроком -1:

$$B = ((B - S_{2i+1}) >>> A) \oplus A$$

$$A = ((A - S_{2i}) >>> B) \oplus B$$

$$B = B - S_i$$

$$A = A - S_0$$

Символом «>>>» позначене циклічне зрушення вправо. Звичайно ж, всі додавання й вирахування виконуються по модулю 2^{32} .

Створення масиву ключів складніше, але теж прямолінійно. Спочатку байти ключа копіюються в масив L із 32-бітових слів, доповнюючи при необхідності заключне слово нулями. Потім масив S ініціалізується за допомогою лінійного конгруентного генератора по модулю 2^{32} :

$$S_0 = P$$

Для i від 1 до $2(r + 1) - 1$:

$$S_i = (S_{i-1} + Q) \bmod 2^{32}$$

де $P = 0xb7e15163$ і $Q = 0x9e3779b9$.

Нарешті, потрібно підставити L в S :

$$i = j = 0$$

$$A = B = 0$$

Виконати $3n$ раз (де n – максимум від $2(r + 1)$ і c):

$$A = S_i = (S_i + A + B) <<< 3$$

$$B = L_i = (L_i + A + B) <<< (A + B)$$

$$i = (i + 1) \bmod 2(r + 1)$$

$$j = (j + 1) \bmod c$$

5 МЕТОДИКА ВПРОВАДЖЕННЯ СИСТЕМИ
В ПРОМИСЛОВУ ЕКСПЛУАТАЦІЮ

На рисунку 5.1 зображено розроблене у бакалаврської дипломної роботі система безпеки програмно визначаємих мереж на базі технології SDNS. З рисунку можна побачити що інтерфейс головного вікна розподілено на наступні функціональні розділи: Бокова панель навігації; Верхній інформаційний блок; Блок ключових показників; Блок вкладок; Блок стану компонентів SDNS; Блок оперативних дій; Блок рекомендацій системи; Таблиця мережевих подій та інцидентів; Блок статусного рядка.

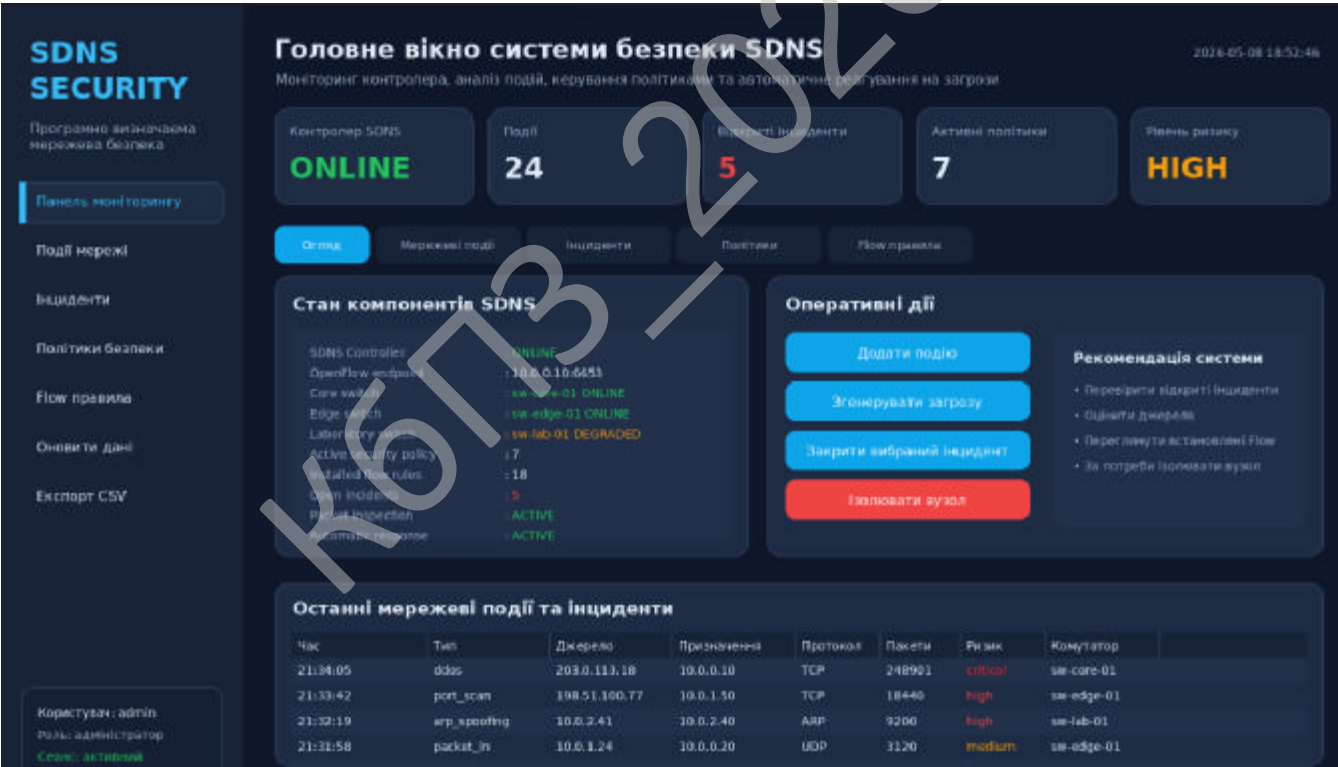


Рисунок 5.1 – Головне вікно ПЗ

Розроблена програма має дуже простий і зрозумілий інтерфейс з користувачем. Кожен, хто в достатньому обсязі володіє операційним

середовищем Windows без особливих складностей освоїть і цю програму, оскільки її інтерфейс інтуїтивно зрозумілий. Якщо програма не видала ніяких помилок, і працює, то можна використовувати, інакше слід слідувати інструкціям, які пропонує програма.

На рисунку 5.2 зображено авторські дані розробленого програмного забезпечення.

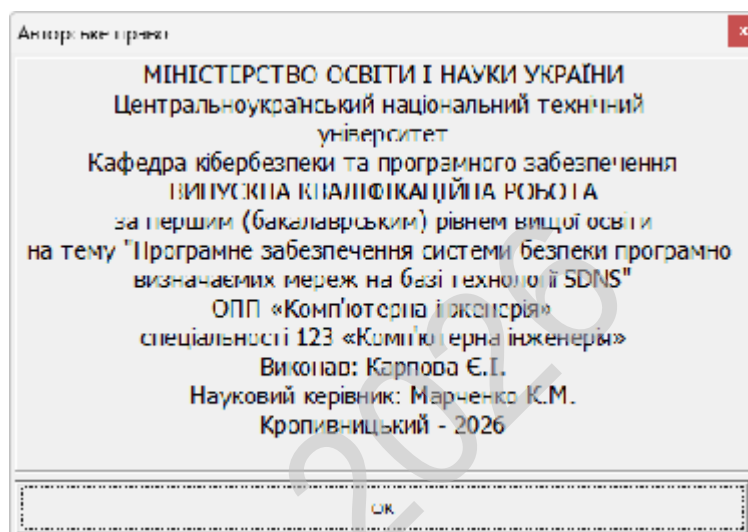


Рисунок 5.2 – Авторське право

Розглянемо процес впровадження програмного забезпечення, це процес налаштування програмного забезпечення під певні умови використання, а також навчання користувачів роботі з програмним продуктом. Впровадження програмного забезпечення це усі дії, що роблять розроблену програмну систему готовою до використання. Даний процес є частинною життєвого циклу програмного забезпечення.

Загалом процес розгортання складається з кількох взаємопов'язаних дій із можливими переходами між ними. Ця активність може відбуватися як з боку виробника так і з боку споживача. Оскільки кожна програмна система є унікальною, то усі процеси та процедури під час розгортання важко передбачити. Тому, "розгортання" можна трактувати як загальний процес відповідно до певних

					ВКРБ-123.26.0046.00.00.ПЗ	Арк.
Вим.	Арк.	№ докум.	Підпис	Дата		73

вимог та характеристик. Розгортання може здійснюватись програмістом і в процесі розробки програмного забезпечення.

До діяльностей пов'язаних із розгортанням програмного забезпечення відносять:

- Випуск.
- Встановлення та активація.
- Деактивація.
- Адаптація.
- Оновлення.
- Вмонтування.
- Відстежування версій.
- Видалення.
- Вилучення з обігу.

При впровадженні програмного забезпечення потрібно урахувати наступні дії:

– Виділення критичних, з точки зору загального результату, процедур в діяльності організації. Коли набір таких процедур визначений, необхідно в першу чергу використовувати ІТ рішення для автоматизації операцій усередині саме цих процедур. Таким чином, розроблене ІТ рішення автоматично стає життєво важливим і затребуваним для організації, а також буде забезпечена публічність процесу впровадження;

– Розширення нормативної бази організації шляхом включення до неї регламентів, що описують порядок виконання процедур автоматизованих процесів. В іншому випадку є небезпека виникнення неузгодженості між автоматизованими процедурами та іншими процесами організації.

– Виконання робіт з загальної стандартизації існуючої діяльності організації, коли виділяються кращі практики виконання процедур і включаються в ІТ рішення за принципом найбільшої корисності для більшості учасників. Відсоток таких процедур щодо загального обсягу автоматизації може бути

					ВКРБ-123.26.0046.00.00.ПЗ	Арк.
Вим.	Арк.	№ докум.	Підпис	Дата		74

невеликий, але це надає процесу побудови рішення вагу в організації за рахунок збільшення його необхідності.

Під час роботи над програмою було проведено тестування програмного забезпечення, тобто технічне дослідження, призначене для виявлення інформації про якість продукту відносно контексту, в якому воно має використовуватись.

Тестування включає як процес пошуку помилок або інших дефектів, так і випробування програмних складових з метою їх оцінки.

Проводилась оцінка:

- відповідності поставленим вимогам;
- правильна відповідь для усіх можливих вхідних даних;
- виконання функцій за прийнятний час;
- практичність;
- сумісність з ОС та стороннім ПЗ.

Оскільки число можливих тестів для програмних компонент практично нескінченне, тому стратегія тестування полягала в тому, щоб провести всі можливі тести з урахуванням наявного часу та ресурсів.

Як результат ПЗ тестувалось стандартним виконанням програми з метою виявлення помилок або інших дефектів.

Проводилось тестування форматом білої скриньки засноване на аналізі керуючої структури програми. Програма вважається повністю перевіреною, якщо проведено вичерпне тестування маршрутів (шляхів) її графа управління.

У цьому випадку формуються тестові варіанти, в яких:

- Гарантується перевірка всіх незалежних маршрутів програми.
- Знаходяться гілки True, False для всіх логічних рішень.
- Виконуються всі цикли (у межах їхніх кордонів та діапазонів).
- Аналізується правильність внутрішніх структур даних.

Недоліки тестування "білої скриньки":

- Кількість незалежних маршрутів може бути дуже велика.

					ВКРБ-123.26.0046.00.00.ПЗ	Арк.
Вим.	Арк.	№ докум.	Підпис	Дата		75

– Повне тестування маршрутів не гарантує відповідності програми вихідним вимогам до неї.

– У програмі можуть бути пропущені деякі маршрути.

– Не можна виявити помилки, поява яких залежить від даних.

Переваги тестування "білої скриньки" пов'язані з тим, що принцип «білої скриньки» дозволяє врахувати особливості програмних помилок:

– Кількість помилок мінімально в «центрі» і максимально на «периферії» програми.

– Попередні припущення про ймовірність потоку керування або даних у програмі часто бувають некоректними. У результаті типовим може стати маршрут, модель обчислень за яким опрацьована слабо.

– При записі алгоритму програмного забезпечення у вигляді тексту на мові програмування можливе внесення типових помилок трансляції (синтаксичних та семантичних).

– Деякі результати в програмі залежать не від вихідних даних, а від внутрішніх станів програми.

Обрано умови розповсюдження – proprietary software. Програмне забезпечення, на яке зберігаються як немайнові, так і майнові авторські права. Отримавши або придбавши таке програмне забезпечення, користувач отримує обмежені права користування ним: може бути заборонено або закрито доступ до коду (вивчення), внесення змін, тиражування, розповсюдження та перепродаж. Програмне забезпечення вважається власницьким, якщо наявне хоча б одне з перелічених обмежень.

Найчастіше основним методом захисту майнових прав на власницьке ПЗ, поза ліцензійною угодою, власник обирає закриття сирцевого коду, захищаючи свій продукт від модифікації і вбудовуючи системи обмеження користування через авторизацію. Таке програмне забезпечення називається закритим. Проте, код власницького продукту може бути і відкритим, але власник може обмежити права користувача умовами користувацької ліцензії.

					ВКРБ-123.26.0046.00.00.ПЗ	Арк.
Вим.	Арк.	№ докум.	Підпис	Дата		76

Власницьке програмне забезпечення та комерційне програмне забезпечення не є синонімами – власницьким може бути і безплатне (тобто, некомерційне) програмне забезпечення.

На противагу власницькому ПЗ існує вільне програмне забезпечення, автори і власники якого дозволяють вивчати, модифікувати і поширювати свій продукт. Саме визначення власницького програмного забезпечення виникло в результаті діяльності громадського руху вільного програмного забезпечення (представленого Фондом вільного програмного забезпечення та іншими організаціями) і осмислення умов свободи користування програмами. Визначенням власницького програмного забезпечення є не невідповідність хоча б одній з базових умов вільного програмного забезпечення. Сама назва власницьке ПЗ підкреслює визначальне значення власника у способі використання і можливостях розвитку цього програмного забезпечення.

					ВКРБ-123.26.0046.00.00.ПЗ	Арк.
Вим.	Арк.	№ докум.	Підпис	Дата		77

6 ОСНОВНІ ВИСНОВКИ

Програмне забезпечення, створене в результаті виконання випускної кваліфікаційної роботи за першим (бакалаврським) рівнем вищої освіти, призначено для системи безпеки програмно визначаємих мереж на базі технології SDNS.

В межах України в недостатній мірі представлені вітчизняні розробки в цій області.

Рішення завдання полягало у вирішенні наступних задач:

- Був проведений огляд існуючих систем безпеки програмно визначаємих мереж на базі технології SDNS.
- Досліджена система безпеки програмно визначаємих мереж на базі технології SDNS.
- На основі отриманих результатів досліджень створена програмна реалізація системи безпеки програмно визначаємих мереж на базі технології SDNS.

Розроблені під час виконання випускної кваліфікаційної роботи за першим (бакалаврським) рівнем вищої освіти алгоритми дозволяють успішно вирішувати завдання безпеки програмно визначаємих мереж на базі технології SDNS.

Розроблене програмне забезпечення має простий, дружній та зручний інтерфейс користувача, що забезпечує легкість у освоєнні роботи програмного продукту, зручність у використанні, і не потребує особливих спеціальних знань.

При створенні програмного забезпечення було використано об'єктно-орієнтований підхід, що відповідає сучасним тенденціям у галузі розробки комерційних програмних систем.

Програма реалізована на мові високого рівня Python. Дана мова програмування дозволяє найбільш ефективно обробляти дані призначені для системи безпеки програмно визначаємих мереж на базі технології SDNS. Це

					ВКРБ-123.26.0046.00.00.ПЗ	Арк.
Вим.	Арк.	№ докум.	Підпис	Дата		78

дозволило мінімізувати строк розробки програмного забезпечення, і, як слід, зменшити витрати на його розробку. Запропоноване програмне забезпечення ділиться на загальне програмне забезпечення, що поставляється із засобами обчислювальної техніки й спеціальне програмне забезпечення, що спеціально розроблене для даної конкретної системи й включає програми, що реалізують її функції.

Програма призначена для виконання під управлінням багатозадачної операційної системи Windows 10/11.

Даються необхідні рекомендації з установки розробленого програмного забезпечення.

Для підвищення рівня безпеки запропоновано застосовувати алгоритм RC5.

В цілому створене програмне забезпечення підтверджує правильність використаних проектних рішень та повністю відповідає вимогам технічного завдання. Створене програмне забезпечення має потенційну можливість для подальшого вдосконалення і застосування у різних галузях.

					ВКРБ-123.26.0046.00.00.ПЗ	Арк.
Вим.	Арк.	№ докум.	Підпис	Дата		79

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Wendell Odom. «CCNA 200-301 Official Cert Guide, Volume 1». Cisco Press. 2020. – 848 p.
2. Wendell Odom. «CCNA 200-301 Official Cert Guide, Volume 2 Premium Edition eBook and Practice Test». Cisco Press. 2020. – 624 p.
3. Scott Jernigan «CompTIA Network+ Certification All-in-One Exam Guide, Eighth Edition». 2022. – 976 p.
4. Doug Lowe «Networking For Dummies 12th Edition». 2020. – 480 p.
5. Ramon Nastase «Computer Networking: The Beginner's guide for Mastering Computer Networking, the Internet and the OSI Model». 2018. – 186 p.
6. Russ White & Ethan Banks «Computer Networking Problems and Solutions: An Innovative Approach to Building Resilient, Modern Networks». 2017. – 832 p.
7. Вінтенко Б., Смірнов О., Миронець І., Смірнова Т., Смірнов С. «Імітаційна модель шляхів вхідних даних комп'ютерної інтелектуальної системи підтримки оператора енергоблоку АЕС». *Комбінаторні конфігурації та їхні застосування: Матеріали XXVII Міжнародного науково-практичного семінару, присвяченого 125-річчю Національного університету «Запорізька політехніка» (Запоріжжя-Кропивницький-Київ, 4-6 червня 2025 р.)*. Запоріжжя: НУ «Запорізька політехніка», 2025. С.82-91.
8. Al-Azzeh, J., Ayyoub, B., Mesleh, A., Smirnova, T., Gnatyuk, S., Drieiev, O., Smirnov, O., Dorenskyi, O. «Cloud-Based Information System for Evaluating Caverns in the Process of Blasting Metal Surfaces of Details». *International Review on Modelling and Simulations* 18 (1), 2025. pp. 32-42.
9. Смірнова Т.В., Коноплицька-Слободенюк О.К., Буравченко К.О., Смірнов С.А., Кравчук О.В., Козірова Н.Л., Смірнов О.А. «Дослідження технологій забезпечення кібербезпеки хмарних сервісів IaaS, PaaS та SaaS». *Кібербезпека: освіта, наука, техніка*. 2024. №4(24), С. 6-27.

					ВКРБ-123.26.0046.00.00.ПЗ	Арк.
Вим.	Арк.	№ докум.	Підпис	Дата		80

10. Батрак О., Смірнова Т., Гнатюк В., Одарченко Р., Смірнов О. «Дослідження показників ефективності функціонування та перспектив розвитку систем ІР-телефонії». *Підводні технології*, 2024, № 13, с. 28-35.
11. Kuznetsov, O., Kryvinska, N., Ilchenko, O., Smirnova, T., Ulianovska, Y. «Comparative Analysis of Cryptocurrency Trading Platforms Using the Analytic Hierarchy Process». *CEUR Workshop Proceedings*, 2023, 3628, pp. 106-115.
12. Al-Mudhafar Aqeel, A.M., Smirnova, T., Buravchenko, K., Smirnov, O. «The method of assessing and improving the user experience of subscribers in software-configured networks based on the use of machine learning». *Advanced Information Systems*, 2023, 7(2), pp. 49-56.
13. Smirnov, O., Sydorenko, V., Aleksander, M., Zhyharevych, O., Yenchев, S. «Simulation of the cloud IoT-based monitoring system for critical infrastructures». *CEUR Workshop Proceedings*, Volume 3530, 2023, pp. 256-265.
14. Smirnov, O., Odarchenko, R., Smirnova, T., Bondar, S., Volosheniuk, D. «Optimal Structure Construction of Private 5G Network for the Needs of Enterprises». *Lecture Notes on Data Engineering and Communications Technologies*, 2023, 178, pp. 208–223.
15. Аль-Мудхафар Акіл Абдулхуссейн М., Смірнова Т.В., Буравченко К.О., Смірнов О.А. «Метод оцінки та підвищення користувальницького досвіду абонентів в програмно-конфігурованих мережах на основі використання машинного навчання». *Сучасні інформаційні системи*, 2023, том 7, № 2, С. 49-56.
16. Smirnova, T., Gnatyuk, S., Yudin, O., Sydorenko, V., Polozhentsev, A., «The Model for Calculating the Quantitative Criteria for Assessing the Security Level of Information and Telecommunication Systems». *CEUR Workshop Proceedings Volume 3156*, 2022, Pages 390-399.
17. Смірнова Т.В., Гнатюк С.О., Сидоренко В.М., Юдін О.Ю., Сидоренко С.Ю., «Модель визначення критичності галузевих інформаційно-телекомунікаційних систем». *Проблеми інформатизації та управління*, № 2(70). 2022. С. 28-37.

18. Смірнов О.А., Смірнова Т.В., Якименко Н.М., Смірнов С.А., Поліщук Л.І., «Дослідження стійкості до диференціального криптоаналізу запропонованої функції гешування удосконаленого модуля криптографічного захисту в інформаційно-комунікаційних системах» *Системи управління, навігації та зв'язку*, 2022, № 3(69). С. 93-98.

19. Смірнов О.А., Смірнова Т.В., Якименко Н.М., Поліщук Л.І., Смірнов С.А. «Дослідження статистичної стійкості та швидкісних характеристик запропонованої функції гешування удосконаленого модуля криптографічного захисту в інформаційно-комунікаційних системах» *Вісник Хмельницького національного університету. Серія: «Технічні науки»*, № 2 (307). С. 46-52. 2022.

20. Смірнов О.А., Смірнова Т.В., Константинова Л.В., Смірнов С.А., Якименко Н.М., «Дослідження стійкості до лінійного криптоаналізу запропонованої функції гешування удосконаленого модуля криптографічного захисту в інформаційно-комунікаційних системах» *Системи управління, навігації та зв'язку*, 2022, № 1(67). С. 84-89.

21. Smirnova T., Gnatyuk S., Berdibayev R., Avkurova Zh., Iavich M. «Cloud-Based Cyber Incidents Response System and Software Tools». *Communications in Computer and Information Science*, 2021, vol 1486. Springer, Cham. pp 169-184.

22. Smirnov O., Kuznetsov A., Kiian A., Kuznetsova T. «Non-binary constant weight coding technique». *CEUR Workshop Proceedings*. Volume 2740, 2020, Pages 102-114.

23. Smirnov O., Alimseitova Zh., Adranova A., Akhmetov B., Lakhno V., Zhilkishbayeva G. «Models and algorithms for ensuring functional stability and cybersecurity of virtual cloud resources». *Journal of theoretical and applied information technology* Vol.98. No 21, 2020, P. 3334-3346.

24. Smirnov O., Kuznetsov A., Kiian A., Cherep A., Kanabekova M., Chepurko I. «Testing of code-based pseudorandom number generators for post-quantum application». *2020 IEEE 11th International Conference on Dependable*

Systems, Services and Technologies (DESSERT), Ukraine, Kyiv, May 14-18. 2020. P. 172-177.

25. Smirnov O., Kuznetsov A., Pushkar'ov A., Serhiienko R., Babenko V., Kuznetsova T., «Representation of Cascade Codes in the Frequency Domain». In: Radivilova T., Ageyev D., Kryvinska N. (eds) *Data-Centric Business and Applications. Lecture Notes on Data Engineering and Communications Technologies*, vol 48. Springer, Cham. 2021. pp 557-587.

26. Smirnov, O., Markovets, O. Vovk, N., Turchyn, Y., «Model of informational support for social network administrators' content creation». *CEUR Workshop Proceedings* Volume 2616, 2020, Pages 125-136.

27. Smirnov, O., Drieieva, H., Drieiev, O., Polishchuk, Y., Brzhanov, R., Aleksander, M. «Method of fractal traffic generation by a model of generator on the graph». *CEUR Workshop Proceedings* Volume 2616, 2020, Pages 366-379.

28. Smirnov, O., Drieieva, H., Drieiev, O., Simakhin, V., Bondar, S., Odarchenko, R. «Managing multifractal properties of the binary sequence generated with the Markov chains», *CEUR Workshop Proceedings* Volume 2608, 2020, Pages 633-645.

29. Smirnov O. Kuznetsov A., Zaichenko Yu., Pastukhov M., Oleshko O., Kuznetsova K., «Formation of Discrete Signals with Special Correlation Properties». *International Conference on Information and Telecommunication Technologies and Radio Electronics, UkrMiCo 2019*; Odessa; Ukraine; 9-13 September 2019. P.22-28.

30. Smirnov, O., Kuznetsov, A., Kolovanova, I., Kuznetsova, T., «Noise immunity of the algebraic geometric codes». *International Journal of Computing*; 2019, Volume 18, Issue 4 – Research Institute for Intelligent Computer Systems – 2019. – P. 393-407.

31. Smirnov, O., Kuznetsov, A., Reshetniak, O., Ivko, N., Katkova, T., Kuznetsova, T., «Generators of Pseudorandom Sequence with Multilevel Function of Correlation». *2019 IEEE International Scientific-Practical Conference Problems of*

Infocommunications, Science and Technology (PIC S&T), Kyiv, Ukraine, 8 – 11 October 2019 . P.517-522.

32. Smirnov, O., Odarchenko, R., Abakumova, A., Usik, P., Kundyz, M., «QoE optimization technique for media delivery in 5G networks». *2019 IEEE International Scientific-Practical Conference Problems of Infocommunications, Science and Technology (PIC S&T)*, Kyiv, Ukraine, 8 – 11 October 2019. P.597-601.

33. Smirnov, O., Krasnobayev, V., Yanko, A., Kuznetsova, T. «Methods of nulling numbers in the system of residual classes». *CEUR Workshop Proceedings*, Vol 2588, P. 90-106, 2019.

34. Smirnov, O., Kuznetsov, A., Kovalchuk, D., Averchev, A., Pastukhov, M., Kuznetsova, K., «Formation of Pseudorandom Sequences with Special Correlation Properties», *2019 3rd International Conference on Advanced Information and Communications Technologies, AICT -2019/ Lviv, Ukraine, 2-6 July, 2019*, P. 395-399.

35. Smirnov, O., Kuznetsov, A., Kiian, A., Zamula, A., Rudenko, S., Hryhorenko, V., «Variance Analysis of Networks Traffic for Intrusion Detection in Smart Grids», *2019 IEEE 6th International Conference On Energy Smart Systems (2019 IEEE ESS)*, Kyiv, Ukraine April 17-19, 2019 P. 353-358.

36. Smirnov, O., Kuznetsov, A., Kavun, S., Babenko, B., Nakisko, O., Kuznetsova, K., «Malware Correlation Monitoring in Computer Networks of Promising Smart Grids», *2019 IEEE 6th International Conference On Energy Smart Systems (2019 IEEE ESS)*, Kyiv, Ukraine April 17-19, 2019 P. 347-352.

37. Smirnov, O., Kuznetsov, A., Kovalchuk, D., Pastukhov, M., Kuznetsova, K., Prokopovych-Tkachenko, D., «Discrete Signals with Special Correlation Properties», *CEUR Workshop Proceedings Volume 2353, CEUR Workshop Proceedings 2019*, Pages 618-629.

38. Smirnov A.A., Kuznetsov A.A., Danilenko D.A., Berezovsky A., «The statistical analysis of a network traffic for the intrusion detection and prevention systems», *Telecommunications and Radio Engineering*. – Volume 74, Issue 1. – Begel House Inc. – 2015. – P. 61-78.

39. Смірнов О.А., Смірнова Т.В., Буравченко К.О., Кравченко С.С., Горбов В.О., «Хмарна система підтримки прийняття рішень технологічного процесу відновлення поверхонь конструкцій і деталей машин». *Сучасні інформаційні системи*. 2021. Т. 5, № 4. С. 79-95

40. Смірнов О.А., Усік П.С., Миронець І.В., Буравченко К.О., Якименко Н.М. «Метод підвищення ефективності розподіленої обробки даних у комп'ютерних системах операторів стільникового зв'язку» *Вісник Черкаського державного технологічного університету. Технічні науки*. №4. С. 103-110. 2020.

41. О.А.Смірнов, Т.В.Смірнова, Л.І. Поліщук, К.О. Буравченко, А.О.Макевнін, «Дослідження хмарних технологій як сервісів», *Кібербезпека: освіта, наука, техніка*. № 3(7). С. 43-62. 2020.

42. Смірнов О.А., Коноплицька-Слободенюк О.К., Смірнов С.А., Буравченко К.О., Смірнова Т.В., Поліщук Л.І. Інформаційна безпека в комп'ютерних мережах. Навчальний посібник – Кропивницький: вид. Лисенко В.Ф. 2020. – 294 с.

43. О.А. Смірнов, П.С. Усік, «Дослідження перспектив використання технологічних рішень в мережах 5G» у *Кібербезпека та інформаційні технології: монографія*. – Х. : ТОВ «ДІСА ПЛЮС», 2020.С. 122-135.

44. Смірнов О.А., Дреєва Г.М., Дреєв О.М., Смірнова Т.В. «Фрактальний аналіз генератора самоподібного трафіку на основі ланцюга Маркова». *Центральноукраїнський науковий вісник. Технічні науки*. № 2(33). с. 161-172, 2019.

45. Смірнов О.А., Коноплицька-Слободенюк О.К., Смірнов С.А., Буравченко К.О., Смірнова Т.В. Поліщук Л.І. Проектування комп'ютерних систем та мереж. Навчальний посібник – Кропивницький: вид. Лисенко В.Ф. 2019. – 264 с.

46. Smirnov, O., Kuznetsov, A., Kuznetsova., K. Synthesis of Discrete Signals with Improved Correlation Properties. Монографія: In.: ISCI'2019: Information Security in Critical Infrastructures. Collective monograph. Edited by Ivan

					ВКРБ-123.26.0046.00.00.ПЗ	Арк.
Вим.	Арк.	№ докум.	Підпис	Дата		85

D. Gorbenko and Alexandr A. Kuznetsov, ASC Academic Publishing, USA, 2019, pp. 281-299. – ISBN: 978-0-9989826-8-7 (Hardback), ISBN: 978-0-9989826-9-4 (Ebook).

47. Смірнов О.А., Дреєва Г.М. Метод генерування фрактального трафіку за допомогою моделі генератора на графі. Монографія: Інформаційна безпека та інформаційні технології : монографія / за заг. ред. В. С. Пономаренка. – Х. : Вид. Рожко С.Г. 2019. С. 123-139

48. Дреєва Г.М., Смірнов О.А., Дреєв О.М. Метод генерування фрактальноподібної числової послідовності на основі скінченного автомату для моделювання трафіку у мережі. Центральнотрафіку український науковий вісник. Технічні науки. № 1(32). с. 173-183, 2019.

49. Смірнова Т.В., Солових Є.К., Смірнов О.А., Дреєв О.М. Побудова хмарних інформаційних технологій оптимізації технологічного процесу відновлення та зміцнення поверхонь деталей. Центральнотрафіку український науковий вісник. Технічні науки. № 1(32). с. 184-194, 2019.

50. Смірнов О.А., Смірнов С.А., Поліщук Л.І., Смірнова Т.В., Коноплицька-Слободенюк О.К. Метод формування антивірусного захисту даних з використанням безпечної маршрутизації метаданих. Кібербезпека: освіта, наука, техніка. – Том 3 № 3. – Київ: КУ ім. Бориса Грінченка. – 2019. – С. 63-87.

51. Смірнов О.А., Гнатюк С.О., Кавун С.В., Терейковський І.А., Жмурко Т.О., Смірнов С.А., Коваленко А.С. Основи безпеки в комп'ютерних мережах. Навчальний посібник – Кропивницький: вид. Лисенко В.Ф. 2018. – 177 с.

52. Смірнов О.А., Котелянець В.В. Стійкі до колізій стохастичні моделі функціонування безпроводових сенсорних мереж. Вісник інженерної академії України, №3, с. 145-152, 2018

53. Смірнов О.А., Смірнов С.А., Дідик А.К., Дреєв А.М. Алгоритми формування безлічі маршрутів передачі метаданих у антивірусні хмарні системи. Збірник наукових праць "Системи обробки інформації". - Випуск 5 (142). - Х.: ХУПС - 2016. - С. 148-152.

					ВКРБ-123.26.0046.00.00.ПЗ	Арк.
Вим.	Арк.	№ докум.	Підпис	Дата		86