МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ ТЕХНОЛОГІЙ ТА ДИЗАЙНУ
ФАКУЛЬТЕТ МЕХАТРОНІКИ ТА КОМП'ЮТЕРНИХ ТЕХНОЛОГІЙ

КАФЕДРА КОМП'ЮТЕРНИХ НАУК

КВАЛІФІКАЦІЙНА РОБОТА

на тему:

Розроблення інформаційної системи для контролю за використанням комп'ютерних мереж підприємства

Рівень вищої освіти **магістерський**

Спеціальність **122 Комп'ютерні науки**

Спеціалізація (за наявності)

_____

Освітня програма **Комп'ютерні науки**

Виконав: студент групи МгІТ-1-22
Кириченко Ілля Андрійович

Науковий керівник к.т.н., доц.

Яхно Володимир Михайлович

Рецензент д.т.н.

проф. Володимир Щербань

Київ 2023

КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ ТЕХНОЛОГІЙ ТА ДИЗАЙНУ

Факультет / Інститут **Факультет Мехатроніки Та Комп'ютерних Технологій**

Кафедра **Комп'ютерних Наук**

Рівень вищої освіти **магістерський**

Спеціальність **122 Комп'ютерні науки**

Спеціалізація _____

Освітня програма **Комп'ютерні науки**

ЗАТВЕРДЖУЮ

Завідувач кафедри КН

_____ _____

(підпис) (Власне ім'я та ПРІЗВИЩЕ)

«_____» _____2023р

ЗАВДАННЯ

НА КВАЛІФІКАЦІЙНУ РОБОТУ СТУДЕНТУ

Кириченко Іллі Андрійовичу

1. Тема роботи: Розроблення інформаційної системи для контролю за використанням комп'ютерних мереж підприємства ,

Науковий керівник роботи Яхно Володимир Михайлович к.т.н., доцент.

затверджені наказом КНУТД від «___»_____20__ року №___

2. Вихідні дані до роботи: Розробки кафедри комп'ютерних наук та технологій, рекомендована література, додатки.

3. Зміст кваліфікаційної роботи: Розділ 1. Торетичний Огляд Основних Складових Технологій Інформаційних Систем Та Комп'ютерних Мереж, Розділ 2. Дослідження Аналогів Та Огляд Технологій Для Розробки Программного Продукту, Розділ 3. Програмна Реалізація

4. Дата видачі завдання 20.10.2023

5.Консультанти розділів кавліфікаційної магістерської роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		Завдання видав	Завдання прийняв
Розділ 1	Яхно Володимир Михайлович к.т.н., доцент.		
Розділ 2	Яхно Володимир Михайлович к.т.н., доцент.		
Розділ 3	Яхно Володимир Михайлович к.т.н., доцент.		
Висновки	Яхно Володимир Михайлович к.т.н., доцент.		

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів дипломної магістерської роботи	Термін виконання	Примітка про виконання
1	Вступ	25.10.2023	
2	Розділ 1. Торетичний Огляд Основних Складових Технологій Інформаційних Систем Та Комп'ютерних Мереж	26.10.2023	
3	Розділ 2. Дослідження Аналогів Та Огляд Технологій Для Розробки Программного Продукту	02.11.2023	
4	Розділ 3. Програмна Реалізація	07.11.2023	
5	Висновки	10.11.2023	
6	Оформлення (чистовий варіант)	14.11.2023	
7	Подача кваліфікаційної роботи (проєкту) науковому керівнику для відгуку (за 14 днів дозахисту)	15.11.2023	
8	Подача кваліфікаційної роботи (проєкту) для рецензування (за 12 днів дозахисту)	15.11.2023	
9	Перевірка кваліфікаційної роботи (проєкту) на наявність ознак плагіату (за 10 днів до захисту)	15.11.2023	
10	Подання кваліфікаційної роботи (проєкту) на завідувачу кафедри (за 7днів до захисту)	15.11.2023	

З завданням ознайомлений:

Студент _____ Ілля Кириченко

Науковий керівник роботи _____ Володимир Яхно

АНОТАЦІЯ

Кириченко І.А. Розроблення інформаційної системи для контролю за використанням комп'ютерних мереж підприємства.

Дипломна магістерська робота за спеціальністю 122 - «Комп'ютерні науки». – Київський національний університет технологій та дизайну, Київ, 2023 рік.

Розглянуті теоретичні аспекти та питання практичної реалізації прийняття рішень для контролю за використанням комп'ютерних мереж підприємства і розв'язання задач оновлення специфічного виду обладнання – програмних засобів. Інформаційна система для контролю за використанням комп'ютерних мереж підприємства є основою для формування моделей прийняття рішень параметри моделей є не чіткими. Інформаційна система містить інформацію для методів узгодження рішень та координації між особами, які приймають рішення. Кожний керівник відповідає за деяку (важливу) частину загальної проблеми рішень оновлення та проектування комп'ютерних мереж. Вірне рішення має вирішальне значення для досягнення загальної продуктивності підприємства. В роботі наведені принципи формування та функції програмних засобів системи прийняття рішень для реалізації задач проектування комп'ютерних мереж на принципах використання нечітких уявлень про оцінки якості та ефективність комп'ютерних мереж.

Ключові слова: проектування комп'ютерних мереж

Технології: C#, Sql Server, Oracle VM Virtual Box

ANNOTATION

Kirichenko I.A. Development of an information system for monitoring the use of enterprise computer networks.

Master's thesis in specialty 122 - "Computer Science". - Kyiv National University of Technology and Design, Kyiv, 2023.

Considered theoretical aspects and issues of practical implementation of decision-making to control the use of computer networks of the enterprise and solving the problems of updating a specific type of equipment - software tools. The information system for monitoring the use of the company's computer networks is the basis for the formation of decision-making models, the parameters of the models are not clear. An information system contains information for methods of agreement and coordination between decision makers. Each manager is responsible for some (important) part of the overall problem of updating and designing computer networks. The right decision is critical to achieving the overall productivity of the enterprise. The paper presents the principles of formation and function of software tools of the decision-making system for the implementation of computer network design tasks based on the principles of using vague ideas about the evaluation of the quality and efficiency of computer networks.

Keywords: **design of computer networks**

Technologies: **C#, Sql Server, Oracle VM Virtual Box**

ЗМІСТ

ВСТУП

У сучасному підприємства, компанії та маленькі контори, які випускають будь-який продукт, усе більше розростаються та вдосконалюються в методах організації робочого процесу та умов для виконання своєї роботи ще краще. Розвиток комп'ютерних мереж зробив дуже великий внесок у цей процес.

Усе більше структури потребують спрощення та більш ефективних способів контролювати усю інформацію про кількість, ціну продукту та усе що пов'язано з ефективністю праці, розроблення продукту тощо. Особливо гострим питання стає, коли мова заходить о підприємствах та компаніях, які мають комп'ютерне обладнання.

Для того щоб вирішити ці питання і нам допомагають:

1. Інформаційні системи

2. Компьютерні мережі

В наш час інформаційні системи дуже розповсюджені і грають дуже важливу роль у будь-якій справі.

Інформаційна система (ІС) - система, призначена для зберігання, пошуку та обробки інформації, і відповідні організаційні ресурси (людські, технічні, фінансові та т. Д.), Які забезпечують і поширюють інформацію .

ІС призначена для своєчасного забезпечення належних людей належної інформацією , тобто для задоволення конкретних інформаційних потреб в рамках певної предметної області, при цьому результатом функціонування інформаційних систем є інформаційна продукція - документи, інформаційні масиви, бази даних та інформаційні послуги.

В наш час інформаційні системи дуже розповсюджені і грають дуже важливу роль у будь-якій справі.

Інформаційна система (ІС) - система, призначена для зберігання, пошуку та обробки інформації, і відповідні організаційні ресурси (людські, технічні, фінансові та т. Д.), Які забезпечують і поширюють інформацію .

ІС призначена для своєчасного забезпечення належних людей належної інформацією , тобто для задоволення конкретних інформаційних потреб в рамках певної предметної області, при цьому результатом функціонування інформаційних систем є інформаційна продукція - документи, інформаційні масиви, бази даних та інформаційні послуги.

Комп'ютерні мережі, які також називають обчислювальними мережами, або мережами передачі даних є логічним результатом еволюції двох найважливіших науково-технічних галузей сучасної цивілізації — комп'ютерних і телекомунікаційних технологій, З одного боку, мережі являють собою окремий випадок розподілених обчислювальних систем, в яких група комп'ютерів узгоджено виконує набір взаємозалежних завдань, обмінюючись даними автоматичному режимі.

Таким чином у цій роботі нам необхідно дослідити та розробити інформаційну систему яка буде працювати з даними використання комп’ютерної мережі на підприємстві.

РОЗДІЛ 1. ТОРЕТИЧНИЙ ОГЛЯД ОСНОВНИХ СКЛАДОВИХ ТЕХНОЛОГІЙ ІНФОРМАЦІЙНИХ СИСТЕМ ТА КОМП'ЮТЕРНИХ МЕРЕЖ

1.1 Інформаційні системи

1.1.1 Історія

Інформаційні системи здавна знаходять (в тому чи іншому вигляді) досить широке застосування в життєдіяльності людства. Це пов'язано з тим, що для існування цивілізації необхідний обмін інформацією — передача знань, як між окремими членами і колективами суспільства, так і між різними поколіннями.

Інформаційні системи існують з моменту появи суспільства, оскільки на кожній стадії його розвитку існує потреба в управлінні. Місією інформаційної системи є переробка інформації, потрібної для ефективного управління всіма ресурсами організації, створення інформаційного та технічного середовища для управління її діяльністю.

Інформаційна система може існувати і без застосування комп'ютерної техніки — це питання економічної доцільності.

В будь-якій інформаційній системі управління вирішуються задачі трьох типів:

- задачі оцінки ситуації (деколи їх називають задачами розпізнавання образів);
- задачі перетворення опису ситуації (розрахункові задачі, задачі моделювання);
- задачі прийняття рішень (в тому числі і оптимізаційні).

1.1.2 Ролі, функції інформації в інформаційних системах

Інформаційні технології широко визначаються як сукупність комп'ютерів системи, що використовуються організацією. Інформаційні технології, у вузькому вигляді визначення відноситься до технологічної

сторони інформаційної системи. Це включає обладнання, програмне забезпечення, бази даних, мережі та інші електронні пристрої. Це можна розглядати як підсистему інформаційної системи. іноді, правда, термін інформаційна технологія також використовується як синонім інформаційна система. Термін IT у найширшому сенсі використовується для опису організації сукупність інформаційних систем, їх користувачів і управління ними, наглядає за ними.

Головна роль IT – сприяти організаційній діяльності та процеси. Ця роль ставатиме все важливішою з часом. Тому це необхідно, щоб кожен керівник і професійний співробітник дізнався про IT не лише у своїй спеціалізованій галузі, а й у всій організації також у міжорганізаційних умовах. Очевидно, що ви будете ефективнішими в обраній професії, якщо ви зрозуміти, як будуються, використовуються та керуються успішні інформаційні системи. Ви також будете більш ефективними, якщо знаєте, як розпізнавати та уникати невдалі системи та збої. Крім того, багато в чому, маючи рівень комфорту з інформаційними технологіями дозволить вам поза роботою та в приватному житті життя, щоб скористатися перевагами нових IT-продуктів і систем у міру їх розробки.

(Хіба ви б не хотіли пояснювати друзям, як новий продукт працює, ніж той, хто про це запитує?) Нарешті, вам варто дізнатися про IT адже знання інформаційних технологій також може підвищитися можливості працевлаштування. Незважаючи на те, що комп’ютеризація усуває деяких робочих місць, це також створює набагато більше. Попит на персонал із традиційних інформаційних технологій, наприклад програмістів, системних аналітиків і дизайнерів — це істотно. В додаток, багато чудових можливостей з’являється в нових сферах, таких як інтернет та електронна комерція, m-commerce, мережева безпека, об'єктно-орієнтоване

програмування, телекомунікації, дизайн мультимедіа та документ управління.

1.1.3 Типи інформаційних систем

1. Система знань

Існують різні системи управління знаннями, які організація впроваджує, щоб забезпечити постійний потік нових і оновлених знань у компанію та її процеси. Система роботи з знаннями (KWS) — це одна із систем управління знаннями, яка полегшує інтеграцію нової інформації чи знань у бізнес-процес. Крім того, KWS також пропонує підтримку та ресурси для різних методів створення знань, програм штучного інтелекту та систем групової співпраці для обміну знаннями, серед іншого. Він також використовує графіку, візуальні елементи тощо для поширення нової інформації. Нижче наведено деякі програми, які працюють на основі основних основ KWS:

Дизайнери часто використовують системи автоматизованого проектування (САПР) для автоматизації процесу проектування.

Фінансові робочі станції використовуються для аналізу величезних обсягів фінансових даних за допомогою нових технологій.

Системи віртуальної реальності можна знайти в науковій, освітній та бізнес-сферах для використання графіки та різних систем для представлення даних.

2. Система управління інформацією

Інформаційна система управління надає допомогу менеджерам, автоматизуючи різні процеси, які спочатку виконувалися вручну. Ділова діяльність, як-от відстеження та аналіз ефективності бізнесу, прийняття

бізнес-рішень, складання бізнес-плану та визначення робочого процесу. Він також забезпечує зворотний зв’язок з менеджерами шляхом аналізу ролей і обов’язків. Система управління інформацією вважається важливою програмою, яка надзвичайно допомагає менеджерам. Ось деякі з переваг інформаційної системи:

Це підвищує ефективність і продуктивність компанії Він надає чітку картину діяльності організації

Це додає цінність існуючим продуктам, впроваджує інновації та покращує розвиток продукту

Це допомагає в спілкуванні та плануванні бізнес-процесів

Це допомагає організації забезпечити конкурентну перевагу

3. Система підтримки прийняття рішень

Система підтримки прийняття рішень — це інформаційна система, яка аналізує бізнес-дані та іншу інформацію, пов’язану з підприємством, щоб запропонувати автоматизацію прийняття рішень або вирішення проблем. Менеджер використовує його в моменти негараздів, що виникають під час роботи підприємства. Як правило, система підтримки прийняття рішень використовується для збору інформації щодо доходів, показників продажів або запасів. Вона використовується в різних галузях промисловості, а система підтримки прийняття рішень є популярною інформаційною системою.

4. Система автоматизації діловодства

Система автоматизації офісу — це інформаційна система, яка автоматизує різні адміністративні процеси, зокрема документування, запис даних і офісні операції тощо. Система автоматизації діловодства поділяється

на управлінську та діловодну діяльність. Ось деякі бізнес-діяльності, які здійснюються за допомогою цього типу інформаційної системи:

Електронна пошта

Голосова пошта

Обробка текстів

5. Система обробки транзакцій

Система обробки транзакцій автоматизує процес збирання, модифікації та пошуку транзакцій. Особливістю цього типу інформаційної системи є те, що вона підвищує продуктивність, надійність і узгодженість бізнес-операцій. Це допомагає підприємствам виконувати повсякденні операції без зайвих зусиль. Як тільки ви добре розбираєтесь у різних типах інформаційних систем, розуміння застосування цих систем стає легким для розуміння.

6. Система підтримки керівників

Система підтримки керівників або ESS допомагає керівникам найвищого рівня планувати та контролювати робочий процес і приймати бізнес-рішення. Це дуже схоже на систему управління інформацією або MIS.

Ось деякі з унікальних характеристик ESS:

Він забезпечує чудові телекомунікаційні можливості, кращі обчислювальні можливості та ефективні варіанти відображення для керівників.

Він надає їм інформацію через статичні звіти, графіки та текстову інформацію на вимогу.

Це допомагає відстежувати продуктивність, відстежувати стратегії конкурентів і прогнозувати майбутні тенденції, серед іншого.

1.1.4 Фінансування інформаційних технологій

Витрати на інформаційні системи в організаціях, включаючи обладнання, програмне забезпечення та витрати на людей, які пов'язані з наданням послуг інформаційних систем різні функціональні відділи повинні оплачуватися відділом або джерелом фінансування в організації. Одна з ключових ролей інформаційного директора (CIO) гарантує, що відділ інформаційних технологій в організації щорічно отримує свою частку коштів. Існує три основні методи фінансування інформаційних систем в організаціях, які описують, як стягуються ці витрати: накладні витрати, повернення платежу та розподіл. Кожен метод має свої переваги та недоліки, які зведені в **Табл. 1**. Не дивно, що у кожного методу також є свої прихильники та противники групи зацікавлених сторін в організаціях.

	Переваги	Недоліки
Накладні витрати	• Легко впровадити, оскільки Є витрати розподіляються на всіх функціональні зони • Заохочує інновації оскільки з користувачів не стягується плата для перевірених нових інноваційних технологій	• Функція інформаційних систем розглядається лише як центр витрат • Відсутність відповідальності за використання ІБ ресурси за функціональними напрямками • Відділ ІБ може бути відсутнім

		підзвітність і реагування на функціональні сфери
Повернення платежу	• Забезпечує функціональність відділів з точним вартість того, що вони використовують • Забезпечує більшу підзвітність функціональних відділів • Інформаційні системи відділ визнано за надані послуги • Інформаційні системи відділ може нести відповідальність за послуги (і прибутковість, якщо це мета) • Функціональні менеджери можуть краще контролювати свої витрати на ІБ	• Важко обчислити всі прямі і непрямі витрати точно • Важке і дороге обслуговування кошторису витрат та актуальну інформацію про використання
Розподіл	• Інформаційні системи	• Відділів може не бути

	відділ може бути відповідальний за послуги та прибутковість • Простіший у використанні та обслуговуванні ніж метод повернення коштів	справедливо платять за інформацію системні ресурси, які вони використовують • Важко визначити основу виділення (коефіцієнт)

Табл. 1.1

Накладні витрати є найпростішим методом фінансування, оскільки він передбачає розгляд усіх витрат на інформаційні системи як загальних витрат для організації. Під час формування річного бюджету витрати на інформаційні системи додаються до нього окремою статтею бюджет. У цьому сенсі, коли використовується підхід фінансування накладних організацію інформаційних систем часто називають центром витрат (не забезпечує жодного прибутку організації).

Метод повернення платежів, як він передбачає, передбачає повернення коштів до функціоналу відділами свої прямі витрати на ресурси та послуги інформаційних систем вони використовують. У цьому відношенні метод повернення платежів дозволяє відділам, які роблять більше використання інформаційних систем, щоб платити більше, ніж відділи, які використовують менше ресурсів інформаційних систем. Звичайно, це вимагає, щоб відділ інформаційних систем міг чітко визначити всі витрати для кожного функціонального підрозділу.

Метод розподілу також передбачає відшкодування витрат функціональним відділам за ресурси та послуги інформаційних систем. Однак кількість коштів, що стягуються з функціональних відділів, є функцією метрики або співвідношення як на противагу фактичним прямим видаткам відомств. Ця метрика може базуватися про чисельність працівників відділу, доходи відділу, або будь-який інший показник чи співвідношення. У деяких аспектах метод розподілу забезпечує баланс між методом повернення платежів і методом накладних витрат.

1.1.5 Визначення витрат на інформаційні системи

До того, як витрати можна буде розподілити або стягнути з відділів, або до бюджету ІС навіть можна встановити, важливо, щоб витрати були чітко визначені. Багато підходи можуть бути використані для визначення цих витрат. Є два популярні методи калькуляції на основі діяльності (ABC) і загальну вартість володіння (TCO). Останніми роками ABC стає все менш поширеним, оскільки TCO дає більш повну картину всіх пов’язаних витрат, як прямих, так і непрямих.

Приклади витрат, які необхідно визначити для інформаційних систем, включають витрати на обладнання, програмне забезпечення, мережі та дані (наприклад, зберігання, резервне копіювання). Важливо, щоб усвідомлюйте, що існують також м’які витрати, які слід визначити, наприклад витрати на людей, технічну підтримку, впровадження програмного забезпечення, оцінку технологій або навчання кінцевих користувачів. Крім того, як і в інших сферах організації, витрати на інформаційні системи можна класифікувати як прямі або непрямі витрати (їх часто називають накладними витратами). Прямі витрати безпосередньо пов’язані з виробництвом продукту чи послуги. Непрямий
витрати пов’язані не з продуктом чи послугою, а скоріше із загальним функціонуванням організація. Прямі витрати можна далі поділити на постійні витрати, ті, що є безпосередньо пов’язані з продуктом або послугою,

але не залежать від рівня виробництво цього продукту чи послуги та змінні витрати, пов’язані з обсяг виробленого товару або послуги.

Підхід загальної вартості володіння (TCO) враховує додаткові витрати, початкові витрати на придбання апаратного та програмного забезпечення, пов’язані з впровадженням інформаційної системи. Ці додаткові витрати включають витрати на персонал для встановлення встановлення та обслуговування систем, витрати на навчання кінцевих користувачів, витрати на оновлення до підтримувати систему в актуальному стані тощо. Існують навіть витрати, пов’язані з виходом на пенсію активи інформаційних систем після закінчення терміну їх корисного використання. Ці витрати також повинні бути включені в розрахунку загальної вартості володіння. Ви можете собі уявити, що придумати оцінки загальної вартості володіння активів інформаційних систем є дуже складним завданням, що вимагає багато інформації та припущення. Це краще робити особам, які мають значний досвід інформаційних систем і всередині організації.

Калькуляція на основі діяльності (ABC) використовує концепцію діяльності для розподілу накладних витрат інформаційних систем на кожну діяльність, що відбувається в організації. Діяльність може бути завданням, функцією або процесом, необхідним для розвитку організації або виробляти свої продукти чи послуги. Тоді підхід ABC призначить їх накладні витрати на вироблені продукти або послуги.

Щоб використовувати підхід ABC, організації необхідно визначити витрати на обладнання, програмне забезпечення та персонал, пов’язані з кожною діяльністю. Це часто включає в себе обчислення кількість часу, витраченого на діяльність персоналом інформаційних систем. На наприклад, є витрати, пов’язані з налаштуванням системи, її тестуванням і навіть оцінкою перед її придбанням. Ці витрати необхідно визнати, навіть якщо система є лише одним із інструментів, що використовуються для розробки кінцевого продукту. Наприклад, оцінка брандмауера, необхідного для захисту даних, які використовуються в розробці споживчого продукту

потрібно було б визнати. Тоді витрати на діяльність розподіляється на продукти чи послуги, які є результатами цієї конкретної діяльності.

1.1.6 Безпека та конфіденційність ІС

План безпеки інформаційних систем (ІС) — це комплексний документ, який визначає загрози безпеці, з якими може зіткнутися організація, і всі засоби контролю, реалізовані для усунення цих загроз. План також включає політику та процедури інформаційної безпеки. Як обговорюється в книзі, політики описують те, що загальне інструкції з безпеки призначені для організації, тоді як процедури є конкретними заявами, що описують, як реалізувати політику безпеки. Перед створенням плану безпеки організація повинна оцінити рівень безпека, яку це вимагає. Потреба у вищих рівнях безпеки (більше інструментів безпеки та функції) часто передбачає більші витрати та меншу гнучкість для користувачів систем. Витрати більші не тільки тому, що обладнання дорожче, але й тому що організація повинна наймати людей з навичками керування цим обладнанням. Гнучкість зменшується, оскільки засоби контролю безпеки можуть обмежувати те, що є співробітниками здатний зробити; наприклад, коли вони подорожують у віддалені місця. Якщо організація інвестує значні суми грошей у високо захищені системи, які співробітники не можна використовувати, це перешкоджає меті. Так само, якщо працівники не бажають дотримуватися процедури безпеки, навіть найкращі системи безпеки можуть не забезпечити належного рівня безпеки.

Розробка плану безпеки ІС – це велике підприємство, яке вимагає залучення ключового персоналу інформаційних систем, а також кінцевих користувачів і управління вищого рівня.

Визначити загрози непросто, хоча деякі з них мають бути очевидними: зловмисні атаки, як хакери проникають у системи, катастрофи (наприклад, повені, пожежі), поломки (наприклад, електропостачання
втрати, збої) або навіть віруси, передані недбалими працівниками. Людські помилки також необхідно враховувати. Інша складність виявлення загроз

полягає в тому, що необхідно враховувати всі обчислювальні середовища організації. Це включає веб-сайти, інтрамережі, канали зв'язку, сервери, клієнтські комп'ютери, мобільні пристрої, програмне забезпечення та люди, щоб назвати декілька. Є кілька способів оцінити, як організація справляється з цими загрозами. Наприклад, організація може подивитися на її використання системи, події та інші дані тривог у своїх журналах або виконувати власний контроль порушення безпеки, щоб перевірити, чи є вони вразливими. Нарешті, деякі люди критично ставляться до функціонування системи, і якщо ці люди загинуть або втратять дієздатність, це може спричинити серйозний ризик для організації. Після визначення ризиків вони ранжуються від найбільш критичного до найменш критичного.

На другому етапі процесу розробки плану безпеки ІБ організації необхідно визначити засоби контролю безпеки, які необхідно запровадити, починаючи з найбільш критичних ризиків. Елементи керування оцінюються в контексті більш широких політик і процедур безпеки. У плані безпеки буде перераховано, які системи існують для захисту проти кожної загрози. Експерти погоджуються, що автоматизовані механізми примусу забезпечують найкраща безпека. Такі технології забезпечують автоматичне реагування на події безпеки (наприклад, автоматичне вимкнення облікового запису користувача після кількох невдалих спроб входу) або здатні виявляти, ідентифікувати та зупиняти зловмисників до того, як буде завдано збитків, наприклад за допомогою сервера перехоплення. Як обговорювалося, контроль доступу також є важливою частиною безпеки. Тому план безпеки включатиме інформацію про те, хто має право доступу до кожної послуги, як встановлюється доступ і які існують винятки. Нарешті, план також обговорюватиме аварійне відновлення.

Після розробки плану безпеки ІБ, а потім періодично, план необхідно оцінити та перевірити. Є кілька способів перевірити план безпеки. Організації часто використовують тренування, де вони створюють порушення безпеки, щоб перевірити вжиті заходи (які мають відповідати

плану). Вимірюються показники, наприклад, скільки часу потрібно, щоб повернутися в Інтернет або скільки часу потрібно, щоб відновити нормальний бізнес операції. Існують інші оцінки, щоб перевірити, чи дотримуються користувачі заходів безпеки чи ефективні засоби контролю (наприклад, чи запобігають атакам, чи точні дані). Оскільки обчислювальне середовище має тенденцію швидко змінюватися сьогодні план безпеки необхідно регулярно переглядати.

1.2 Комп’ютерні мережі

1.2.1 Простіший випадок взаємодії двох комп’ютерів

У найпростішому випадку зв’язок комп’ютерів може бути реалізована за допомогою тих самих засобів, які використовуються для зв’язку комп’ютера з периферією, наприклад, через послідовний інтерфейс RS-232C. При цьому, в

На відміну від процедури обміну даними комп’ютера з периферійним пристроєм, коли програма працює, як правило, тільки з одного боку (з боку комп’ютера), тут відбувається взаємодія двох програм, що виконуються на кожному з комп’ютерів.

Програма, що працює на одному комп’ютері, не може отримати безпосередній доступ до ресурсів іншого комп’ютера — його дисків, файлів, принтера. Вона може лише «попросити» про це іншу програму, яку виконує тому комп’ютері, якому належать ці ресурси. Ці «прохання» виражаються у вигляді повідомлень, що передаються каналами зв’язку між комп’ютерами. Повідомлення можуть містити не тільки команди на виконання деяких дій, а й власне інформаційні дані (наприклад, вміст деякого файлу).

Розглянемо випадок, коли користувачеві, який працює з текстовим редактором

на персональному комп’ютері А, потрібно прочитати частину деякого файлу, розташованого на диску персонального комп’ютера. Припустимо, що ми пов’язали ці комп’ютери по кабелю через COM-порти, які, як відомо, реалізують інтерфейс RS-232C (таке з’єднання часто називають нуль модемним). Нехай для певності комп’ютери працюють під керуванням MS-DOS, хоча важливого значення у разі це має.

Драйвер COM-порту разом із контролером COM-порту працюють приблизно так само, як і в описаному вище випадку взаємодії Периферійного Пристрою з комп’ютером. Однак при цьому роль пристрою керування ПП виконують контролер і драйвер COM-порту іншого

комп’ютера. Разом вони забезпечують передачу по кабелю між комп’ютерами одного байта інформації.

Драйвер комп’ютера В періодично опитує ознаку завершення прийому, встановлюваний контролером при правильно виконаній передачі даних, і при його появі зчитує прийнятий байт з буфера контролера в оперативну пам’ять, роблячи його тим самим доступним для програм комп’ютера. У деяких випадках драйвер викликається асинхронно, за перериваннями від контролера. Аналогічно реалізується і передача байта в інший бік – від комп’ютера В до комп’ютера А.

Таким чином, у розпорядженні програм комп’ютерів А і В є засоби для побайтового обміну даними. Але розглянута в нашому прикладі завдання значно складніше, оскільки, по-перше, потрібно отримати з віддаленого комп’ютера не окремий байт, а певну частину заданого файлу, по-друге, ці дані перебувають над оперативної пам’яті цього комп’ютера, але в його периферійний пристрій. Всі пов’язані з цим додаткові проблеми ми повинні вирішити програми вищого, ніж драйвери COM-портів, рівня. Для певності назвемо такі програми комп’ютерів А та В додатком А та додатком відповідно.

Отже, додаток А має сформувати повідомлення-запит для програми У. У запиті необхідно вказати ім’я файлу, тип операції (у разі — читання), зміщення та розмір області файлу, що містить необхідні дані. Це повідомлення міститься в буфер в оперативній пам’яті. Щоб передати цей запит віддаленому комп’ютеру, додаток А звертається до драйвера COM-порту власного комп’ютера і повідомляє йому адресу буфера, в якому знаходиться повідомлення. Потім по щойно описаній схемою драйвер і контролер COM-порту комп’ютера А, взаємодіючим з драй віром і контролером COM-порту комп’ютера, передають повідомлення байт за байтом додатку В.

1.2.2 Адресація вузлів мережі

На початку існування Інтернету було лише кілька хостів (переважно міні-комп’ютерів), підключених до
мережі. Найпопулярнішими програмами були віддалений вхід і передача файлів. До 1983 року їх було вже п'ять сотні хостів, підключених до Інтернету. Кожен із цих хостів ідентифікувався унікальною адресою IPv4. Форсування користувачам запам’ятати IPv4-адреси віддалених хостів, які вони хочуть використовувати, було незручно. Люди вважають за краще запам'ятовувати імена та використовувати їх у разі потреби. Використання імен як псевдонімів для адрес є а поширена техніка в інформатиці. Це спрощує розробку програм і дозволяє розробнику ігнорувати деталі низького рівня.

Щоб мати можливість зв’язатися з кореневими серверами імен, кожен DNS-клієнт повинен знати свої IP-адреси. Це означає, що
DNS-клієнти повинні підтримувати оновлений список IP-адрес кореневих серверів імен 9. Без цього списку це неможливо зв'язатися з кореневими серверами імен. Змусити всі Інтернет-хости підтримувати найновішу версію цей список буде складним з оперативної точки зору. Щоб вирішити цю проблему, розробники DNS представив особливий тип DNS-сервера: DNS-перетворювачі. Резолвер - це сервер, який надає ім'я послуга вирішення проблеми для набору клієнтів. Мережа зазвичай містить кілька резольверів. Кожен хост у цих мережах є налаштований надсилати всі свої DNS-запити через один із своїх локальних резолверів. Ці запити називаються рекурсивними. Резолвер повинен рекурсивно пройти через ієрархію серверів імен, щоб отримати відповідь.

На практиці використовуються кілька типів DNS RR. Тип A використовується для кодування адреси IPv4, яка відповідає вказане ім'я. Тип AAAA використовується для кодування адреси IPv6, яка відповідає вказаному імені. Запис NS містить назву DNS-сервера, який відповідає за даний домен. Наприклад, запит для Запис, пов’язаний з назвою www.ietf.org,

повертає таку відповідь. Ця відповідь містить декілька частин інформації. По-перше, назва www.ietf.org пов’язана з IP-адресою 64.170.98.32. По-друге, доменом ietf.org керують шість різних серверів імен. Три з цих серверів імен доступні через IPv4 та IPv6. Два з них недоступні через IPv6, а ns0.ietf.org доступний лише через IPv6. Запит для запису AAAA, пов’язаного з www.ietf.org, повертає 2001:1890:1112:1::20 і те саме повноваження та додаткові розділи.

DNS в основному використовується для пошуку IP-адреси, яка відповідає даному імені. Однак іноді це корисно щоб отримати назву, яка відповідає IP-адресі. Це робиться за допомогою PTR (вказівника) RR. Частина RData PTR RR містить назву, тоді як частина Name RR містить IP-адресу, закодовану в in-addr.arpa домен. Адреси IPv4 закодовані в in-addr.arpa шляхом заміни чотирьох цифр, які складають пунктирну десяткове представлення адреси. Наприклад, розглянемо адресу IPv4 192.0.2.11. Пов’язане ім’я хоста на цю адресу можна знайти, запитавши PTR RR, який відповідає 11.2.0.192.in-addr.arpa.

1.2.3 Корпоративні мережі

Корпоративна мережа – це логічно відокремлена група комп’ютерів, маршрутизаторів та інших частин IT-інфраструктури, які функціонують поза традиційними межами Інтернету. Її часто називають інтранет. Термін інтранет описує мережу, яка, на відміну від Інтернету, призначена для доступу лише певної групи людей . Існує безліч методів, які можна використати для побудови ексклюзивної комп’ютерної мережі компанії. Пристрої можна з’єднувати разом за допомогою криптографії та VPN через відкритий Інтернет. Абревіатура VPN розшифровується як Virtual Private Network - абстрактна мережа, яка об'єднує лише певні машини, які мають підключення до глобальної мережі Інтернет. Він може бути недоступним із-за меж штаб-квартири компанії або без відповідних облікових даних.

Основна мета створення корпоративних мереж полягає в забезпеченні безпеки та надійності інформаційної системи управління компанією, системи обробки даних, передачі даних і зв'язку - співробітникам може не знадобитися використовувати програмне забезпечення з обмеженими даними, коли вони знаходяться поза робочим місцем. Крім того, розміщення додатків, які актуальні лише для членів компанії виключно в корпоративних мережах, може бути набагато менш складним. Корпоративні мережі використовують різні типи пристроїв LAN та WAN, а також широко доступні протоколи та стандарти зв’язку для надання таких функцій.

Корпоративну мережу також можна описати як набір компаній з формальними чи неформальними відносинами (контрактами, фінансами, матеріально-технічним забезпеченням тощо), які діють для досягнення спільної ринкової мети. Такі мережі створюють середовище для взаємного зростання, і їх можна будувати на основі різноманітних факторів. Приклади форм корпоративної мережі включають:

Корпоративні кластери - відносини ґрунтуються на географічному розташуванні та призводять до розширення компаній, що працюють у схожій сфері. Є багато прикладів таких територій, які відомі в усьому світі: Кремнієва долина, Digital Media City навколо Сеула або «моторні міста» – Штутгарт і Детройт.

Японські дзайбацу – виросли після Першої світової війни, дзайбацу – це набори закладів, побудованих навколо сімейних кланів, які складаються з великої компанії-власника, кількох менших фірм і сусідніх фінансових установ.

Корейські chaebols — схожі на дзайбацу, chaebols — це сімейні групи компаній, що працюють у різних сферах ринку .

1.2.4 Інтерфейс прикладного програмування (сокети)

Хоча кожна операційна система може вільно визначати власний мережевий API (і більшість має), з часом деякі з цих API отримали широку підтримку; тобто їх було перенесено на інші операційні системи їх рідна система. Ось що сталося з інтерфейсом сокета спочатку наданий дистрибутивом Unix у Берклі, який тепер підтримується практично в усіх популярних операційних системах і є основою мовні інтерфейси, такі як бібліотека сокетів Java. Переваги галузевої підтримки єдиного API полягають у тому, що програми можуть легко переносити з однієї ОС на іншу, і розробники можуть легко писати програми для кількох операційних систем. Кожен протокол надає певний набір служб, а API надає синтаксис, за допомогою якого ці служби можуть бути викликані в конкретній комп'ютерній системі. Реалізація є потім відповідає за відображення матеріального набору операцій і об'єктів визначений API на абстрактний набір послуг, визначених протоколом. Якщо ви добре визначили інтерфейс, тоді так і буде можна використовувати синтаксис інтерфейсу для виклику служб багатьох різні протоколи. Така загальність, безумовно, була метою розетки інтерфейс, хоча він далеко не ідеальний.

Основною абстракцією інтерфейсу сокета, як не дивно, є гніздо. Хороший спосіб уявити сокет як точку, де локальний процес підключення до мережі. Інтерфейс визначає операції для створення сокета, підключення сокета до мережі, надсилання/отримання повідомлень через сокет і закриття сокета. Щоб спростити обговорення, ми обмежимося показом того, як бувають сокети використовується з TCP.

1.2.5 TCP: Протокол керування передачею

Незважаючи на те, що протоколи TCP і UDP використовують один і той самий мережевий рівень (IP), протокол TCP забезпечує повне інший сервіс для прикладного рівня, ніж UDP. TCP забезпечує надійну службу потоку байтів, орієнтовану на підключення. Термін «орієнтований на з'єднання» означає дві програми, які використовують TCP (зазвичай

вважається а клієнт і сервер) повинні встановити TCP-з’єднання один з одним, перш ніж вони зможуть обмінюватися даними. Типовою аналогією є набір номера телефону в очікуванні іншого відповісти на дзвінок і сказати «привіт», а потім сказати, хто дзвонить.

TCP забезпечує надійність, виконуючи такі дії:

● Дані програми розбиваються на фрагменти, які TCP вважає найкращим розміром для надсилання.
Це повністю відрізняється від UDP, де кожен запис програмою генерує а Дейтаграма UDP такого розміру. Одиниця інформації, що передається TCP до IP, називається а
сегмент. (Див. Малюнок 1.7) У розділі 18.4 ми побачимо, як TCP вирішує, що це розмір сегмента становить.
● Коли TCP надсилає сегмент, він підтримує таймер, очікуючи на іншу сторону
підтвердження прийому сегмента. Якщо підтвердження не отримано вчасно, сегмент передається повторно. У розділі 21 ми розглянемо адаптивний тайм-аут TCP і стратегія ретрансляції.
● Коли TCP отримує дані з іншого кінця з’єднання, він надсилає підтвердження. Це підтвердження надсилається не відразу, а зазвичай затримується на частку секунди, як ми обговорюємо в Розділі 19.3.
● TCP підтримує контрольну суму свого заголовка та даних. Це наскрізна контрольна сума метою якого є виявлення будь-яких змін даних під час передачі. Якщо приходить сегмент з недійсною контрольною сумою TCP відкидає її та не підтверджує її отримання. (Це очікує від відправника тайм-ауту та повторної передачі.)
● Оскільки сегменти TCP передаються як IP-дейтаграми, а IP-дейтаграми можуть надходити не в порядку, сегменти TCP можуть надходити не в

порядку. Приймаючий TCP за потреби виконує повторне упорядкування даних, передаючи отримані дані в правильному порядку до
додаток.

● Оскільки IP-дейтаграми можуть дублюватися, TCP-одержувач повинен відкидати повторювані дані.

● TCP також забезпечує керування потоком. Кожен кінець TCP-з'єднання має кінцеву кількість
буферний простір. Одержуючий TCP дозволяє лише іншій стороні надсилати стільки даних, скільки приймач має буфери для. Це заважає швидкому хосту використовувати всі буфери на а повільніший хост.

1.2.6 Звіт про помилки (ICMP)

Наступне питання полягає в тому, як Інтернет трактує помилки. Хоча IP цілком готовий відкинути дейтаграми, коли ситуація стає складною, наприклад, коли а маршрутизатор не знає, як переслати дейтаграму або коли один фрагмент дейтаграми не досягає місця призначення — це не обов'язково провалитися мовчки. IP завжди налаштовується за допомогою супутнього протоколу, відомого як Протокол керуючих повідомлень Інтернету (ICMP), який визначає набір повідомлення про помилки, які надсилаються назад на вихідний хост щоразу, коли маршрутизатор або хост не може успішно обробити IP-дейтаграму. Наприклад, ICMP визначає повідомлення про помилки, які вказують на те, що хост призначення недоступний (можливо, через збій посилання), що процес повторного складання не вдався, що TTL досяг 0, контрольна сума IP-заголовка не вдалася тощо. ICMP також визначає декілька керуючих повідомлень, які може надсилати маршрутизатор
назад до вихідного хосту. Одне з найкорисніших керуючих повідомлень називається ICMP-перенаправлення повідомляє вихідному хосту, що існує кращий маршрут до призначення. ICMP-перенаправлення використовуються в наступній ситуації. Припустимо хост, підключений до мережі, яка має два підключених до нього маршрутизатора, називається

R1 і R2, де хост використовує R1 як маршрутизатор за замовчуванням. Якщо R1 коли-небудь отримати дейтаграму від хоста, де на основі його таблиці пересилання вона знає, що R2 був би кращим вибором для певного пункту призначення адресу, він надсилає ICMP-перенаправлення назад на хост, вказуючи йому використовувати R2 для всіх майбутніх дейтаграм, адресованих цьому адресату. Тоді господар додає цей новий маршрут до своєї таблиці пересилання.

1.2.7 Моделі для мережевої безпеки

Аспекти безпеки вступають у гру, коли необхідно або бажано захистити передача інформації від опонента, яка може становити загрозу конфіденційності, достовірності тощо. Усі способи забезпечення безпеки мають два компоненти:

1. Пов’язане з безпекою перетворення інформації, яка надсилається. Приклади включають шифрування повідомлення, яке кодує повідомлення, щоб воно було нерозбірливий для опонента, і додавання коду на основі вмісту повідомлення, за допомогою якого можна перевірити особу відправника.
2. Якась секретна інформація, якою поділилися два директора і, сподіваємось, невідома
до супротивника. Прикладом є ключ шифрування, який використовується в поєднанні з перетворення, щоб зашифрувати повідомлення перед передачею та розшифрувати це.

Для безпечної передачі може знадобитися надійна третя сторона. Для наприклад, третя сторона може нести відповідальність за розповсюдження секретної інформації двом керівникам, утримуючи це від будь-якого супротивника. Або третя сторона може бути необхідні для вирішення спорів між двома принципалами щодо автентичності передачі повідомлення. Ця загальна модель показує, що є чотири основні завдання при проектуванні спеціальної служби безпеки:

1.Розробити алгоритм для виконання перетворення, пов’язаного з безпекою.

Алгоритм має бути таким, щоб супротивник не міг перемогти його мету.
2. Створіть секретну інформацію для використання з алгоритмом.
3. Розробити методи розповсюдження та обміну секретною інформацією.
4. Укажіть протокол, який використовуватимуть два принципали, які використовують захист алгоритм і секретну інформацію для досягнення конкретної служби безпеки.

Усі з проблемами, викликаними наявність хакерів, які намагаються проникнути в системи, до яких можна отримати доступ мережа. Хакером може бути хтось, хто без злого наміру просто отримує задоволення від злому та проникнення в комп'ютерну систему. Порушником може бути а незадоволений працівник, який бажає завдати шкоди, або злочинець, який прагне використовувати комп'ютерні активи для отримання фінансової вигоди (наприклад, отримання номерів кредитних карток або виконання незаконних грошових переказів).

Іншим видом небажаного доступу є розміщення в комп'ютерній системі логіка, яка використовує вразливі місця в системі та може впливати на прикладні програми, а також на службові програми, такі як редактори та компілятори. Програми можуть представляти два види загроз:
1. Загрози доступу до інформації: перехоплення або зміна даних від імені користувачів, які не повинні мати доступу до цих даних.
2. Загрози службі: використовуйте недоліки служб у комп'ютерах, щоб перешкоджати використанню законними користувачів
Віруси та хробаки є двома прикладами програмних атак. Такі напади можуть бути вводиться в систему за допомогою диска, який містить небажану логіку, приховану в корисному програмному забезпеченні. Їх також можна вставити в систему через мережу; цей останній механізм є більш важливим для безпеки мережі.

Механізми безпеки, необхідні для боротьби з небажаним доступом, діляться на дві широкі категорії. Першу категорію можна назвати гейткіпером функція. Він включає процедури входу на основі пароля,

призначені для заборони доступ до всіх користувачів, окрім авторизованих, і логіка перевірки, призначена для виявлення та відкидати хробаків, віруси та інші подібні атаки. Коли небажаний користувач або небажане програмне забезпечення отримує доступ, друга лінія захисту складається з низки внутрішніх засобів контролю, які відстежують діяльність і аналізують інформацію, що зберігається в спробах виявити присутність небажаних зловмисників.

Багато методів безпеки та програм мають визначено як стандарти. Крім того, були розроблені стандарти охоплюють практики управління та загальну архітектуру механізмів безпеки і послуги. Найважливіші стандарти в використовують або розробляють для різних аспектів криптографії та безпеки мережі. Різні організації були залучені до розробки або просування ці стандарти. Найважливіша (в сучасних умовах) з цих організацій є такими:

■ Національний інститут стандартів і технологій: NIST є федеральним агентством США який займається вимірювальною наукою, стандартами та технологіями, пов’язаними з Використання урядом США та просування інновацій у приватному секторі США. Незважаючи на національну сферу дії, Федеральні стандарти обробки інформації NIST (FIPS) і спеціальні публікації (SP) мають світовий вплив.

■ Інтернет-спільнота: ISOC є професійним членським товариством, що працює по всьому світу організаційне та індивідуальне членство. Він забезпечує лідерство у вирішенні проблем, які постають перед майбутнім Інтернетом, і є домом для організації для груп, відповідальних за стандарти інфраструктури Інтернету, включаючи Internet Engineering Task Force (IETF) і Internet Architecture Board (IAB). Ці організації розробляють стандарти Інтернету та відповідні специфікації, усі вони публікуються як запити на коментарі (RFC).

1.3 Висновок до першого розділу

У цьому розділі ми детально розібрали таке явище як інформаційні системи. Дізналися про те для чого вони є, як їх використовують, основні принципи розробки, витрати для підприємства в розробці та підтримці ІС.

Також ми розібрали комп’ютерні мережі, основні принципи їх роботи, методи моделювання і розробки систем на підприємствах, розгляд з точки зору програмного забезпечення, дізналися про проблеми безпеки, негативних чинників та моделі систем безпеки.

Найголовніше те, що ми зрозуміли усю складність і важливість цих двох складових, для організації праці в компаніях та підприємствах у сучасному світі.

РОЗДІЛ 2. ДОСЛІДЖЕННЯ АНАЛОГІВ ТА ОГЛЯД ТЕХНОЛОГІЙ ДЛЯ РОЗРОБКИ ПРОГРАММНОГО ПРОДУКТУ

2.1 Програми для моніторингу мереж

Network Olympus

Програма працює як служба і має веб-інтерфейс, що дає набагато більшу можливостей в роботі. Головна особливість - конструктор сценаріїв, що дозволяє відійти від виконання багатої кількості перевірок, які не дозволяють враховувати обставини роботи технічних пристроїв у мережі. З його допомогою можна організовувати схеми моніторингу будь-якої складності, щоб точно виявляти проблеми і помилки у роботі мережи, а також автоматизувати процес їх усунення.

В основі сценарію лежить сенсор, від якого можна вибудовувати логічні ланцюжки, які в залежності від успішності перевірки будуть генерувати різні оповіщення та дії, спрямовані на рішення ваших завдань. Кожен елемент може бути змінений в будь-який час і відразу застосується для всіх технічних пристроїв, за якими закріплений сценарій. Вся мережева активність буде відслідковуватися за допомогою журналу активності.

Див. Фото 2.1, Табл. 2.1

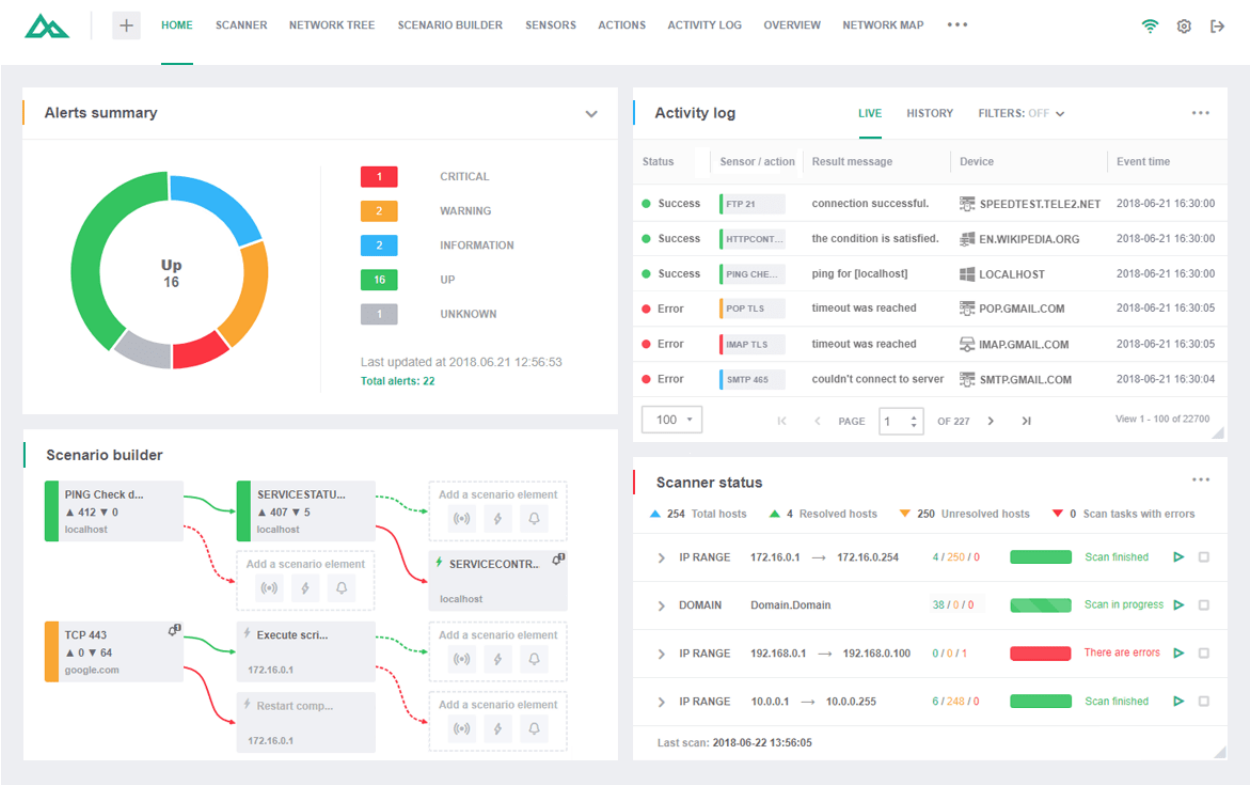


Фото 2.1

Плюси	Мінуси
Групові сенсори	Тільки веб-інтерфейс
Простота налаштування	Установка тільки під Windows
Нескладно освоїти	Немає багатокористувацького доступу
Конструктор сценаріїв моніторингу	

Табл 2.1

Observium

Observium — це платформа для моніторингу та керування мережею, яка в режимі реального часу надає інформацію про стан і продуктивність мережі. Він може автоматично виявляти мережеві пристрої та служби, збирати показники продуктивності та генерувати сповіщення при виявленні проблем.

Observium містить веб-інтерфейс, який дозволяє користувачам переглядати стан мережі та показники продуктивності в режимі реального часу, а також історичні дані. Він розроблений таким чином, щоб бути простим у використанні та обслуговуванні, з акцентом на надання інформації, необхідної адміністраторам мережі для швидкого виявлення та вирішення проблем.

Observium підтримує широкий спектр типів пристроїв, платформ і операційних систем, включаючи Cisco, Windows, Linux, HP, Juniper, Dell, FreeBSD, Brocade, Netscaler, NetApp і багато інших.

Професійно розроблена та підтримувана командою досвідчених мережевих інженерів і системних адміністраторів, Observium — це платформа, розроблена та створена її користувачами.**Див. Фото 2.2, Табл. 2.2**

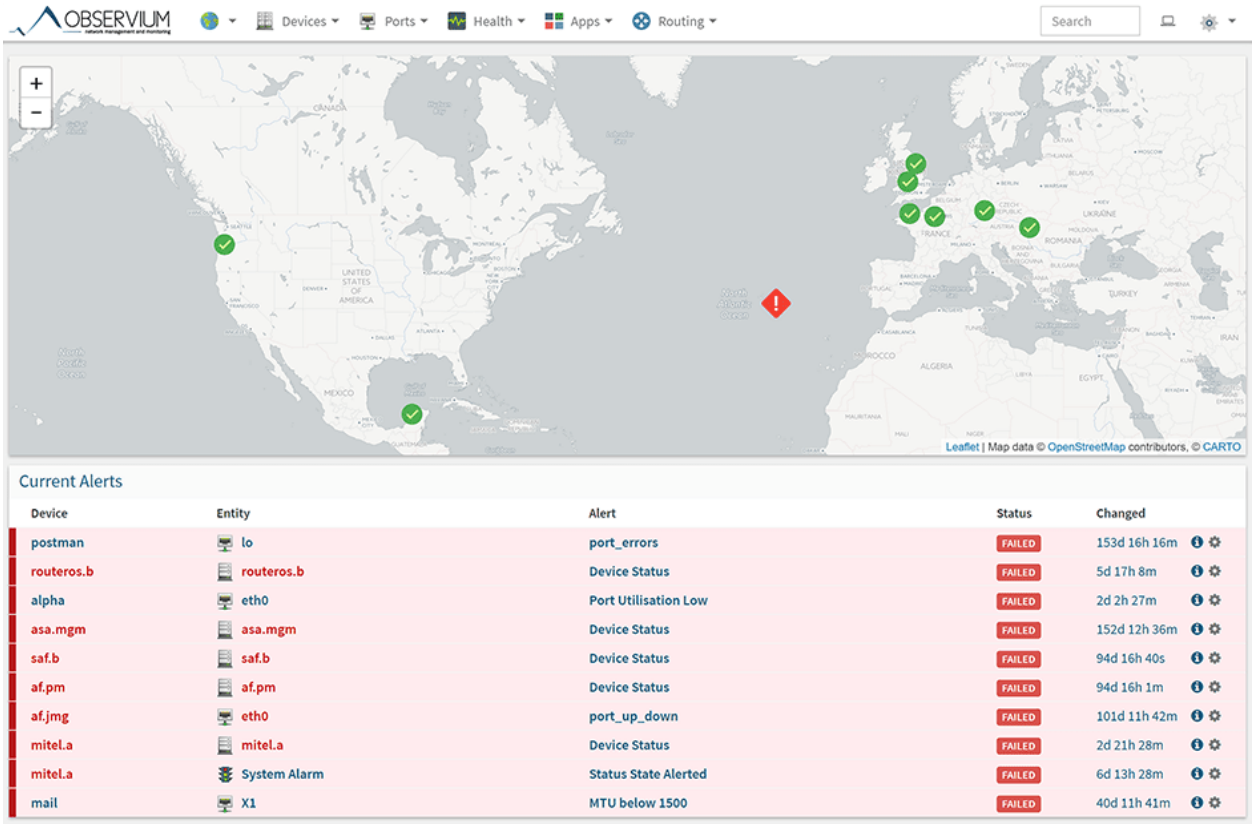


Фото 2.2

Плюси	Мінуси
Доступна безкоштовна версія	Немає підтримки мобільних пристроїв
"Граничні" сигнали	Не проста в установці
Функції автоматичного виявлення	Не для невеликих мереж
Доступна для багатьох систем	Недоліки безкоштовної версії

Табл. 2.2

Nagios

Система моніторингу Nagios через вказаний період часу перевіряє доступність хостів, сервісів і відправляє повідомлення на пошту, за допомогою SMS або іншим самописним скриптом. Також у Nagios є інтерфейс cgi і за допомогою веб-сервера (наприклад, apache), можна переглядати статус служб, хостів, протоколів, змінювати налаштування увідомлень (та деякі інші), графіки (також потрібна бібліотека GD), карту (дерево) моніторингу об'єктів. **Див. Фото 2.3, Табл. 2.3**

Деякі можливості моніторингу Nagios:
Моніторинг мережевих служб (SMTP, POP3, IMAP, FTP, SSH, HTTP, NNTP, PING та ін.)
Моніторинг ресурсів сервера (навантаження процесора, використання дисків та інших.). Інформація віддалених серверів збирається за допомогою доповнення (агента) NRPE addon
Можливість користувачам розробити їх власні перевірки служб
Повідомлення, коли хост недоступний та коли доступ до нього відновлено
Автоматична ротація логів
Веб-інтерфейс для перегляду статусу мережі, сповіщень та історії моніторингу, лог файлів тощо. Також є можливість зберігати результати перевірок у базі даних mysql за допомогою доповнення NDOUtils addon. Це дає можливість розробити новий веб-інтерфейс та побудувати свої графіки.
Розподілений моніторинг (Distributed Monitoring), тобто моніторинг з кількох серверів, при великій кількості хостів, а результати та статуси відправляються на головний сервер
Підтримка віддаленого моніторингу через шифровані тунелі SSH або SSL

.

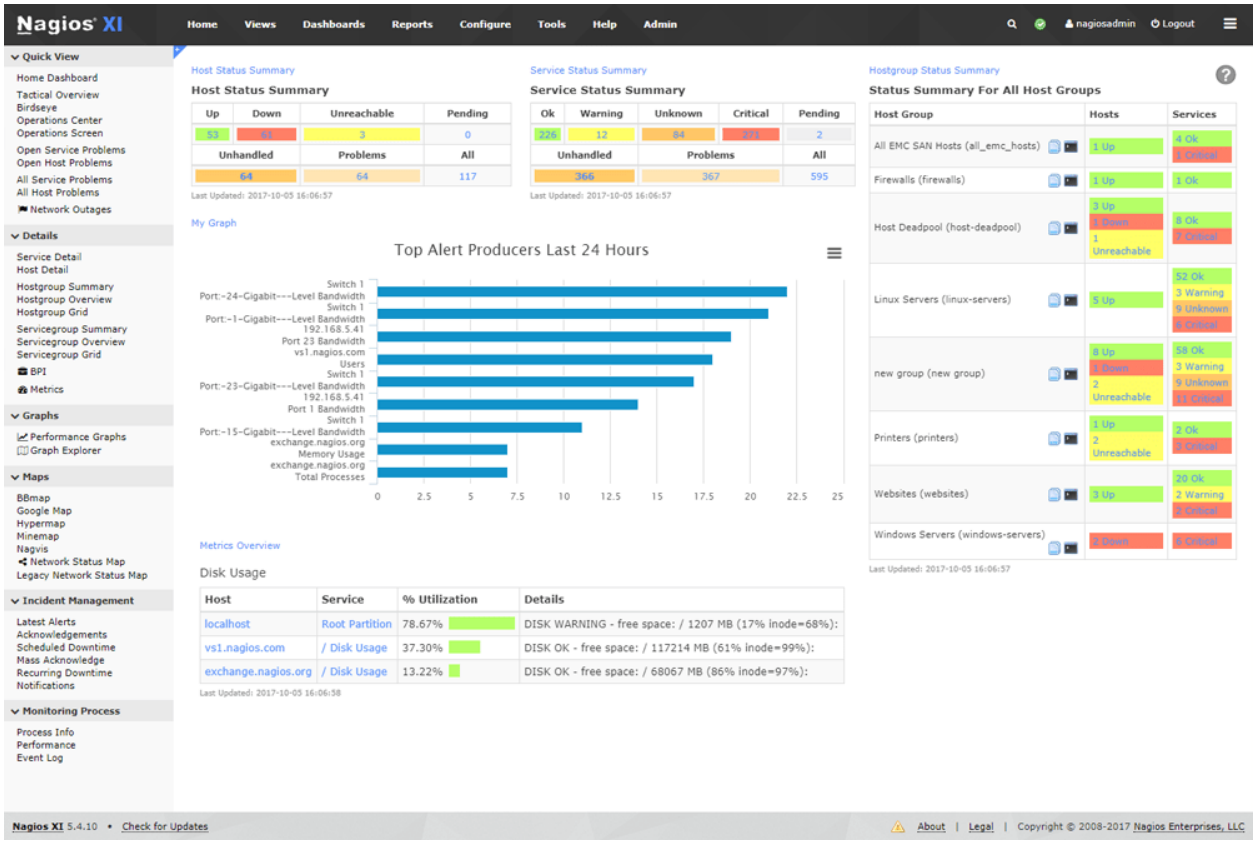


Фото 2.3

Плюси	Мінуси
Висока гнучкість	Трудомістке налаштування
Корисні шаблони	Не для великих організацій
Інтеграція з іншими додатками	

Табл. 2.3

PRTG Network Monitor

Програмний компонент PRTG, сумісний з пристроями на базі ОС Windows, призначений для моніторингу мереж. Він не безкоштовний (безкоштовним є лише пробний 30-ти денний період), використовується не тільки для сканування технічних пристроїв, які в даний момент підключені до локальної мережі, а й може послужити відмінним помічником у виявленні мережевих атак.

Серед найкорисніших мережевих сервісів PRTG: інспекція пакетів, аналіз і збереження статистичних даних в базу, перегляд карти мережі в реальному часу (також доступна можливість отримання історичних відомостей про поведінку мережі), збір технічних параметрів про пристрої, підключені до мережі, а також аналіз рівня навантаження на мережеве обладнання. Зауважимо, що він дуже зручний у використанні - перш за все, завдяки графічному інтерфейсу, який відкривається за допомогою будь-якого браузера. У разі необхідності, системний адміністратор може отримати і віддалений доступ до додатка, через веб-сервер.

Див. Фото 2.4, Табл. 2.4

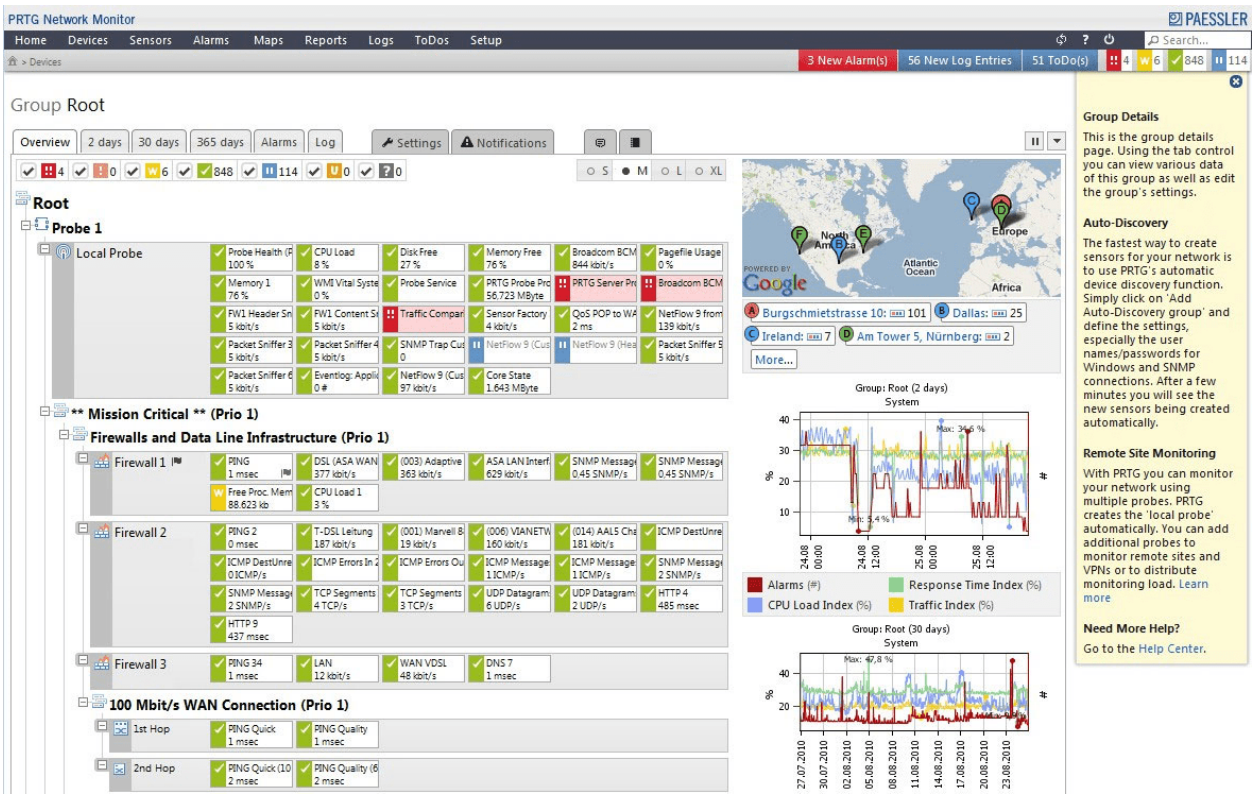


Фото 2.4

Плюси	Мінуси
Безліч функцій	Висока ціна
Настроювані панелі	Немає окремої бази даних
Гнучкий моніторинг	Немає групових сенсорів
Карта мережі	

Табл. 2.4

Kismet

Kismet – це багатофункціональна безкоштовна утиліта для роботи з бездротовими мережами Wi-Fi. Користувачам вона знайома в основному за статтями на тему злому, де використовується для виявлення прихованих мереж або захоплення пакетів. Зламувати чужі мережі - погано, а тим часом Kismet - це набагато більше, ніж відмичка в руках зловмисника. В арсеналі інженера інформаційної безпеки ця програма стає чудовим інструментом для спостереження та аналізу ефіру 802.11.

За його допомоги ви без зусиль знайдете некоректно сконфігуровані і навіть нелегально працюючі точки доступу (які зловмисники використовують для перехоплення трафіку) та інші приховані технічні пристрої, які можуть бути потенційно "шкідливі" для вашої мережі. Для цих цілей в додатку дуже добре опрацьована можливість виявлення різних типів атак - як на рівні мережі, так і на рівні каналів. Як тільки одна або кілька атак будуть виявлені, системний адміністратор отримає тривожний сигнал і зможе вжити заходів щодо усунення загрози.

Див. Фото 2.5, Табл. 2.5

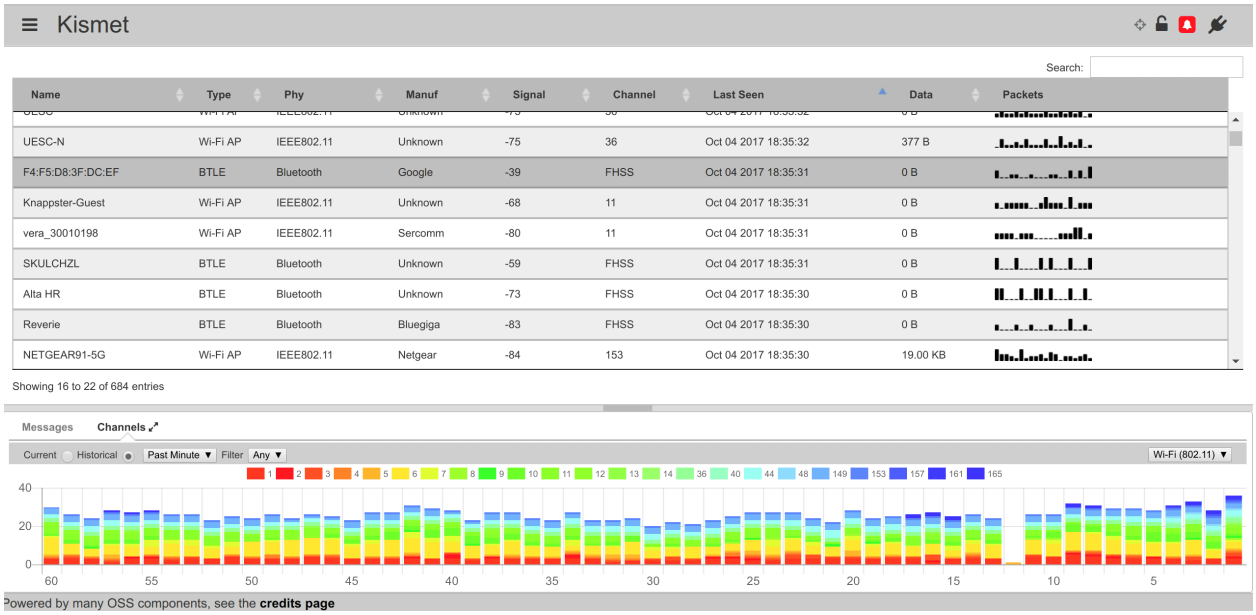


Фото 2.5

Плюси	Мінуси
Безкоштовна	Не проста у використанні
Пакетний сніфер	Повільний сканер
Мінімалістичний інтерфейс	Важко навчитися

Табл. 2.5

WireShark

Безкоштовний open-source аналізатор трафіку WireShark надає своїм користувачам широкий функціонал і по праву визнаний зразковим рішенням в області мережевої діагностики. Він ідеально інтегрується з системами на базі *NIX/Windows/macOS.

Замість не дуже добре зрозумілих для новачків веб-інтерфейсів і CLI, в яких потрібно вводити запити на спеціальній програмній мові, дане рішення використовує GUI (хоча, якщо у вас з'явиться необхідність модернізувати набір стандартних можливостей WireShark, ви запросто зможете запрограмувати їх на Lua).

Розгорнувши і налаштувавши його одного разу на своєму сервері, можна отримати централізований елемент для моніторингу за найдрібнішими змінами в роботі мережі і протоколах. Таким чином, ви зможете на ранніх етапах знаходити проблеми, що виникають в мережі.

Див. Фото 2.6, Табл. 2.6

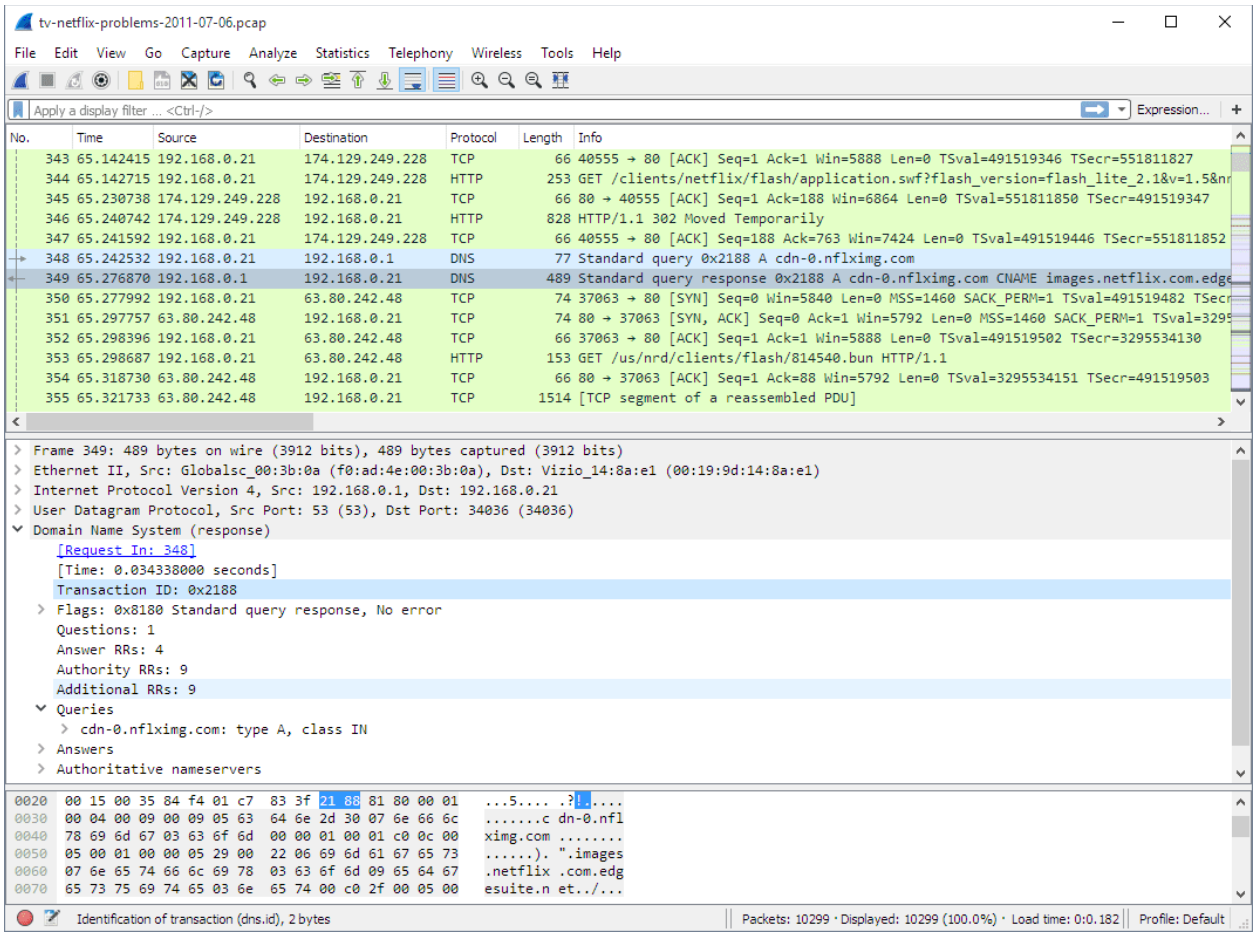


Фото 2.6

Плюси	Мінуси
Безкоштовна	Не інтуїтивна
Проста в установці	Не має мобільної підтримки
Аналізатор пакетів	Не для великих компаній
Гнучкий інтерфейс	

Табл. 2.6

NeDi

NeDi - це повністю безкоштовне програмне забезпечення, яке сканує мережу за MAC-адресами (також серед припустимих критеріїв пошуку є IP-адреси і DNS) і становить з них власну БД. Користуватися їм можна через веб-інтерфейс

Таким чином, ви можете в режимі он-лайн спостерігати за всіма фізичними пристроями і їх розташуванням в вашій локальної мережі (фактично, ви знайдете можливість вилучення даних щодо будь-якого мережевого вузла - починаючи від його прошивки і закінчуючи конфігурацією).

Деякі професіонали задіюють NeDi для пошуку технічних пристроїв, які використовуються нелегально (наприклад, вкрадені). Для підключення до комутаторів або маршрутизаторів дане програмне забезпечення використовує протоколи CDP/LLDP. Це дуже корисне, хоча і непросте в освоєнні рішення.

Див. Фото 2.7, Табл. 2.7

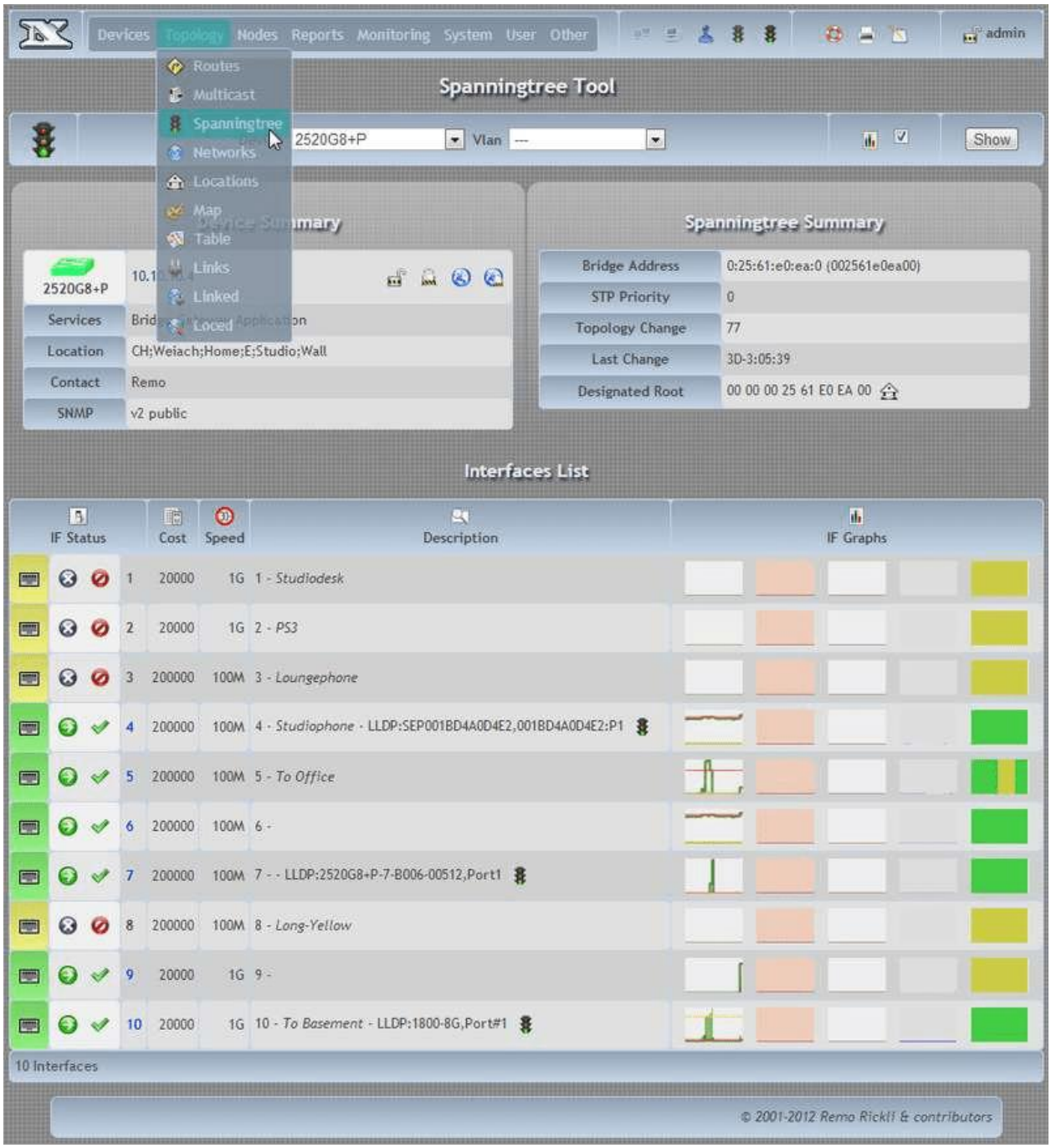


Фото 2.7

Плюси	Мінуси
Безкоштовна	Тільки для macOS
Безліч можливостей	Нелегко встановити
Відмінна карта мережі	Важко навчитися

Табл. 2.7

Zabbix

Система моніторингу Zabbix - це універсальне рішення для моніторингу мереж з відкритим кодом, яке може бути налаштоване під окремі мережеві моделі. Здебільшого, воно призначене для систем, які володіють багатосерверною архітектурою (зокрема, Zabbix інтегрується з серверами Linux/FreeBSD/Windows).

Цей додаток дозволяє одночасно керувати сотнями мережевих вузлів, що робить його вкрай ефективним інструментом в організації роботи системних адміністраторів, які працюють на великомасштабних підприємствах. Для розгортання Zabbix в своїй локальній мережі вам буде потрібно або запустити доменів, або використовувати SNMP-протокол (або інший протокол для захищеного віддаленого доступу); а для управління доведеться освоїти веб-інтерфейс на PHP.

Крім того, це програмне забезпечення надає повноцінний набір інструментів для відстеження стану апаратної частини мережі. Відзначимо, що для того, щоб повною мірою відчути всі переваги цього рішення, вашому адміністратору доведеться мати хочаб базові знання мов Perl або Python (або будь-яких інших мов, які можна спільно використовувати з Zabbix).

Див. Фото 2.8, Табл. 2.8

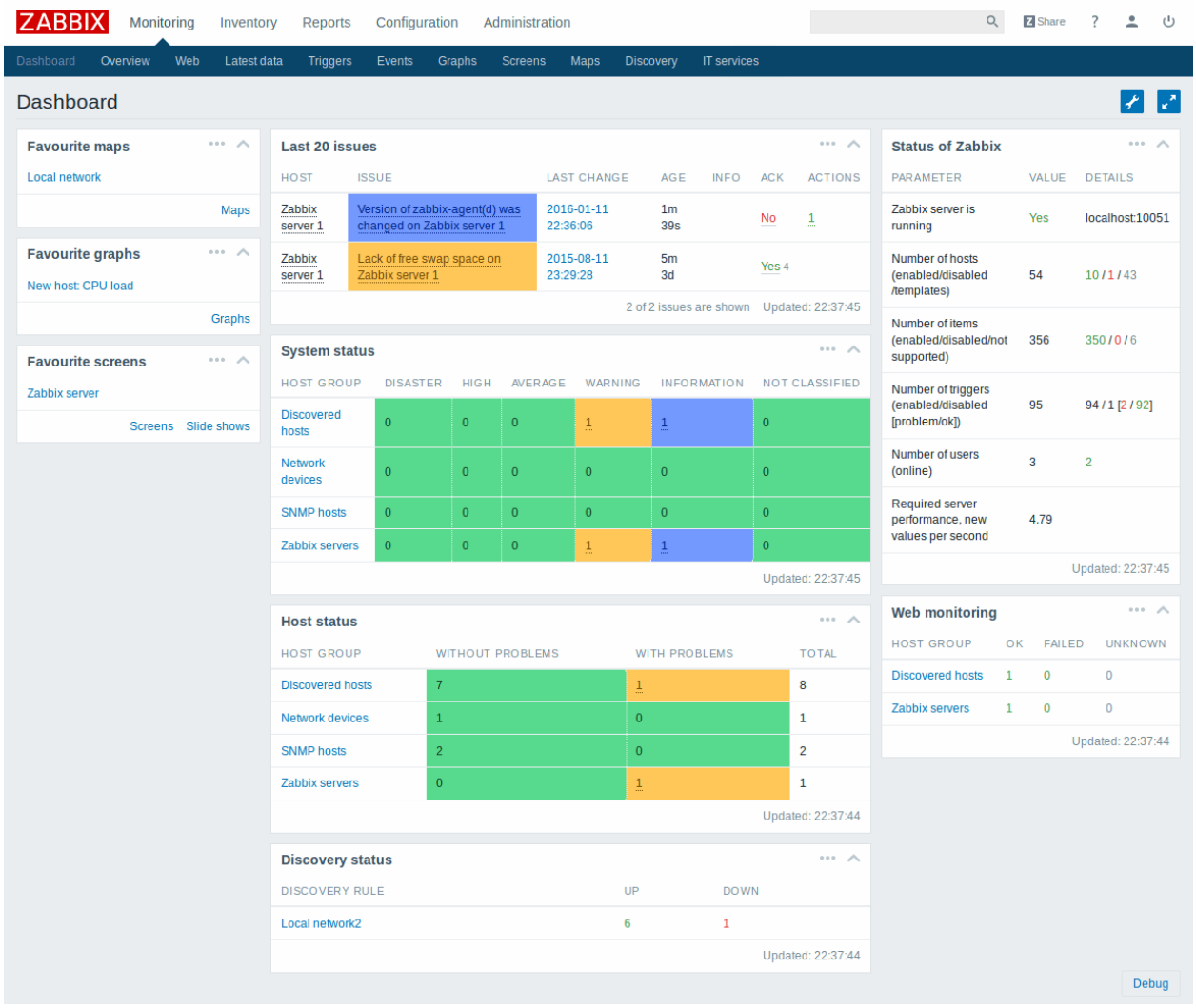


Фото 2.8

Плюси	Мінуси
Безкоштовна	Нема версії для Windows
Проста інсталяція	Складний громіздкий інтерфейс
Безліч плагінів	Високе навантаження на комп'ютер
Потужні налаштування сповіщень	Немає дашбордів

Табл. 2.8

10-Страйк: Моніторинг мережі

"Моніторинг мережі" - це програмне рішення з веб-інтерфейсом, яке повністю автоматизує всі аспекти мережевої безпеки. З його допомогою адміністратори можуть запобігати поширенню по локальній мережі вірусного програмного забезпечення, а також визначати причину виникнення різноманітних технічних проблем, пов'язаних з розривом кабелів або виходом з ладу окремих одиниць мережевої інфраструктури.

Крім того, дане програмне забезпечення в режимі онлайн провіряє температури, напруги, місця на дисках і інших параметрів по SNMP і WMI. Серед його недоліків - досить сильне навантаження на ЦП (про що чесно попереджає сам розробник) і висока ціна.

Див. Фото 2.9, Табл. 2.9

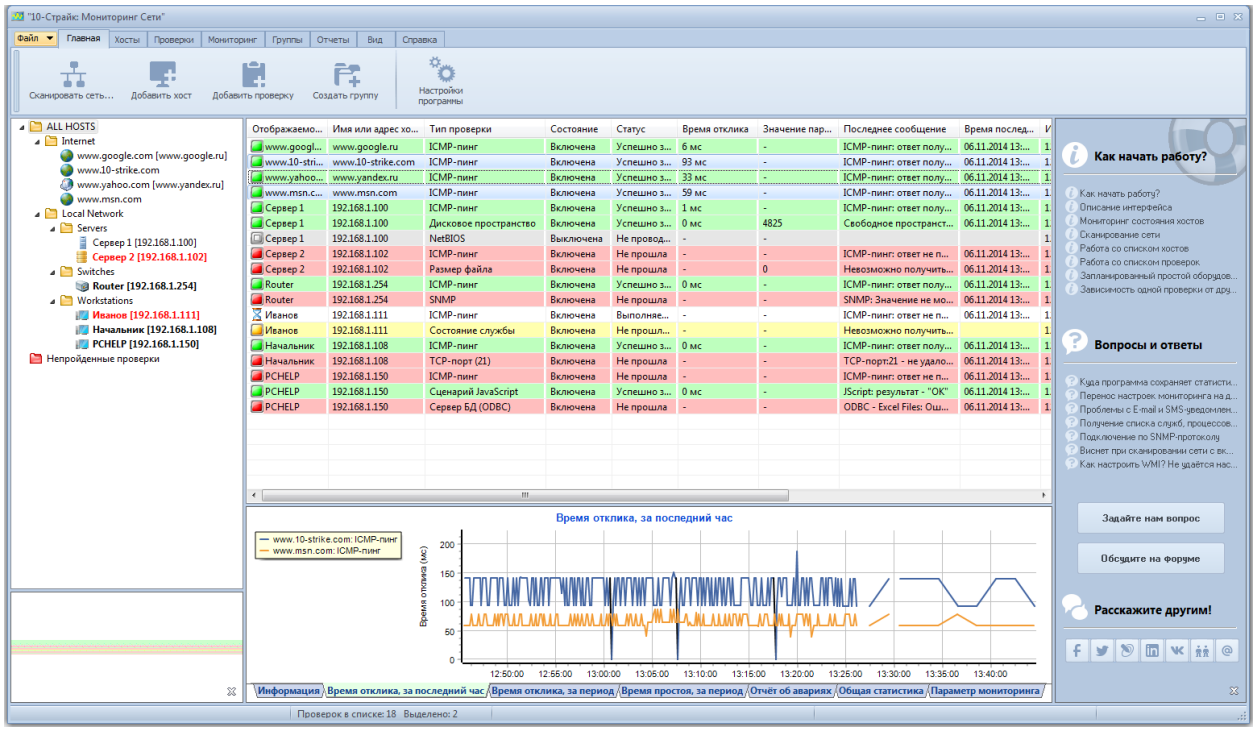


Фото 2.9

Плюси	Мінуси
Зручний та простий інтерфейс	Відносно дорога
	Вимоглива до системи

Табл. 2.9

Total Network Monitor 2

Total Network Monitor 2 - доступне і ефективне програмне забезпечення для мережевого моніторингу, яке має ідеальний баланс між зручністю (в більшості безкоштовних рішень відсутній GUI) та функціоналом. Одними з основних компонентів TNM 2 є перевірки, які виконуються з необхідною вам періодичністю. Вони дозволяють відстежити буквально будь-який параметр, починаючи від доступності серверів в мережі і закінчуючи перевіркою стану сервісів.

Ці об'єкти здатні самостійно усувати первинні наслідки помилок (тобто, відбувається все це без безпосередньої участі системного адміністратора) - наприклад, перезавантажувати окремі служби або користувальницькі пристрої, активувати антивірус, доповнювати журнал подій новими записами, тощо - в загальному, все те, що від початку системний адміністратор виконував своїми руками.

Що стосується звітів, то в ній зберігається вся інформація, пов'язана з кожної перевіркою, яка була проведена обраним монітором.

Див. Фото 2.10, Табл. 2.10

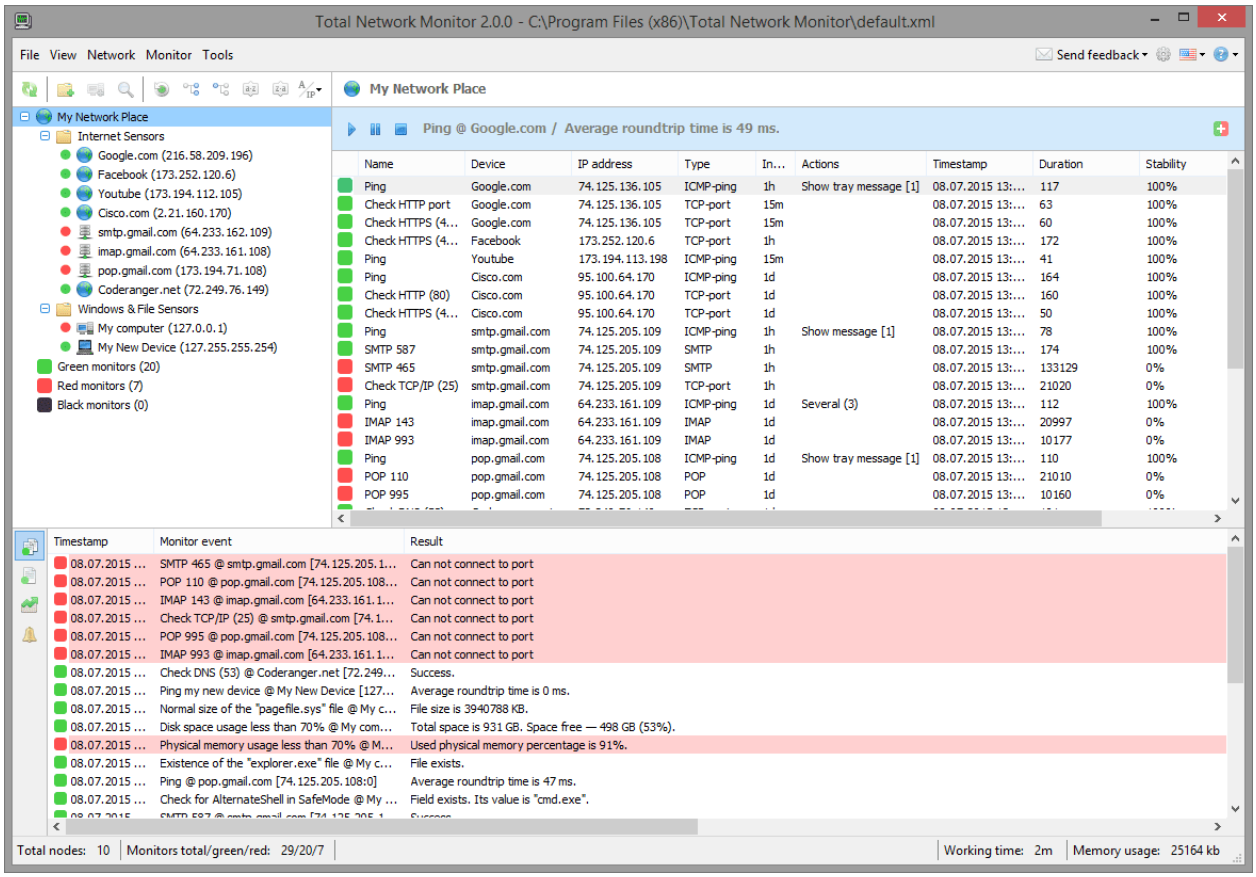


Фото 2.10

Плюси	Мінуси
Низька ціна	Немає дашбордів
Легко встановити	Немає багатопотоковості
Дружній інтерфейс	Не оновлюється

Табл. 2.10

2.2 Бази даних

2.2.1 Абстракція даних

Система баз даних — це сукупність взаємопов'язаних даних і набір програм, які дозволяють користувачам отримувати доступ до цих даних і змінювати їх. Основне призначення бази даних Система має надати користувачам абстрактний вигляд даних. Тобто система приховує певні деталі того, як дані зберігаються та обслуговуються.

Щоб система була придатною для використання, вона повинна ефективно отримувати дані. Потреба в ефективності змусила дизайнерів використовувати складні структури даних для представлення даних у базі даних. Оскільки багато користувачів системи баз даних не навчені комп'ютеру, розробники приховують складності від користувачів через кілька рівнів абстракції, щоб спростити взаємодії з системою:

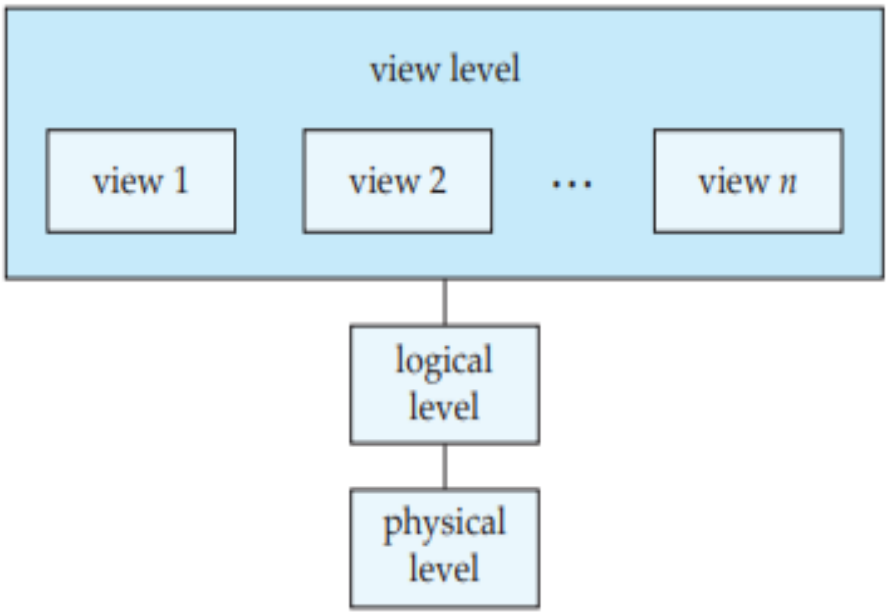
• Фізичний рівень. Найнижчий рівень абстракції описує, як фактично зберігаються дані. Фізичний рівень описує складні низько рівневі структури даних детально.

• Логічний рівень. Наступний вищий рівень абстракції описує, що таке дані зберігаються в базі даних і які зв'язки існують між цими даними.

Таким чином, логічний рівень описує всю базу даних у термінах невеликої кількості відносно прості конструкції. Хоча реалізація простих структур на логічному рівні може включати складні структури фізичного рівня, Користувачеві логічного рівня не потрібно усвідомлювати цю складність. Це називається фізичною незалежністю даних. Адміністратори баз даних, які повинні вирішити, яку інформацію зберігати в базі даних, використовувати логічний рівень абстракції.

• Рівень перегляду. Найвищий рівень абстракції описує лише частину цілого бази даних. Незважаючи на те, що логічний рівень використовує простіші структури, складність залишається через різноманітність інформації, що зберігається у великій базі даних. Багатьом користувачам системи баз даних

не потрібна вся ця інформація; замість цього, їм потрібен доступ лише до частини бази даних. Рівень перегляду абстракції існує для спрощення їх взаємодії з системою. Система може забезпечити багато переглядів для однієї бази даних.



Мал. 2.11

2.2.2 Примірники та схеми

Бази даних змінюються з часом, оскільки інформація вставляється та видаляється. Зібрання інформація, що зберігається в базі даних у певний момент, називається екземпляром бази даних. Загальний дизайн бази даних називається схемою бази даних.

Схеми змінюються рідко, якщо взагалі змінюються. Концепцію схем та екземплярів баз даних можна зрозуміти за аналогією до програми, написаної мовою програмування. Відповідає схема бази даних до оголошень змінних (разом із пов’язаними визначеннями типів) у програмі. Кожна змінна має певне значення в даний момент. Значення змінних у програмі в певний момент часу відповідають екземпляру схеми бази даних. Системи баз даних мають кілька схем, розділених за рівнями

абстракції. Фізична схема описує дизайн бази даних у фізичному рівень, тоді як логічна схема описує дизайн бази даних на логічному рівні. База даних також може мати кілька схем на рівні перегляду, які іноді називаються підсхеми, які описують різні види бази даних.

З них логічна схема є, безумовно, найважливішою з точки зору її ефекту на прикладних програмах, оскільки програмісти створюють програми за допомогою логічна схема. Фізична схема прихована під логічною схемою, і може зазвичай можна легко змінити, не впливаючи на прикладні програми. Кажуть, що програми виявляють фізичну незалежність даних, якщо вони не залежать на фізичній схемі, тому не потрібно переписувати фізичну схему зміни.

2.2.3 Моделі даних

В основі структури бази даних лежить модель даних: сукупність концептуальних інструменти для опису даних, зв'язків даних, семантики даних і узгодженості обмеження. Модель даних забезпечує спосіб опису дизайну бази даних на фізичний, логічний і переглядовий рівні.

Існує кілька різних моделей даних, які ми розглянемо в тексті. Моделі даних можна класифікувати на чотири різні категорії:

• Реляційна модель. Реляційна модель використовує набір таблиць для представлення як даних, так і зв’язків між цими даними. Кожна таблиця має кілька стовпців, і кожен стовпець має унікальну назву. Відомі також столи як відносини. Реляційна модель є прикладом моделі на основі записів.

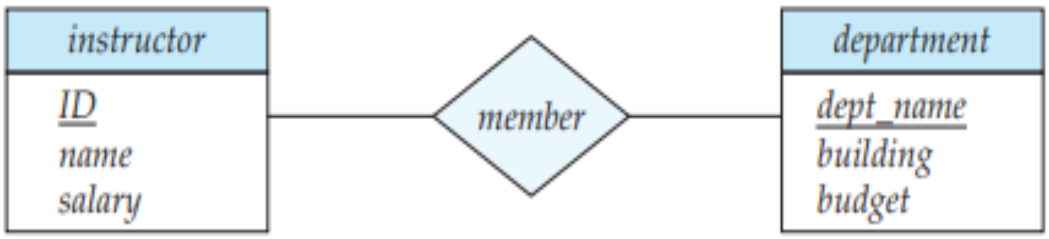
Моделі на основі записів називаються так тому, що база даних структурована записи фіксованого формату кількох типів. Кожна таблиця містить записи певного типу. Кожен тип запису визначає фіксовану кількість полів або атрибутів. Стовпці таблиці відповідають атрибутам типу запису. The реляційна модель даних є найбільш широко використовуваною моделлю даних, і переважна більшість сучасних систем баз даних засновані на реляційній моделі.

• Модель «сутність-зв’язок». Модель даних «сутність-зв’язок» (E-R) використовує сукупність базових об’єктів, які називаються сутностями, і зв’язків між цими об’єктами.

Сутність — це «річ» або «об’єкт» у реальному світі, який можна розрізнити від інших об'єктів. Модель сутність-зв'язок широко використовується в базі даних дизайну. **Див Мал. 2.12**

• Об'єктно-орієнтована модель даних. Об'єктно-орієнтоване програмування (особливо на Java, C++ або C#) став домінуючою методологією розробки програмного забезпечення. Це призвело до розробки об’єктно-орієнтованої моделі даних, яка може бути розглядається як розширення моделі E-R за допомогою понять інкапсуляції, методів (функції) та ідентичність об’єкта. Об’єктно-реляційна модель даних об’єднує особливості об'єктно-орієнтованої моделі даних і реляційної моделі даних.

• Напівструктурована модель даних. Напівструктурована модель даних дозволяє специфікація даних, де можуть бути окремі елементи даних одного типу різні набори атрибутів. Це відрізняється від згаданих моделей даних раніше, де кожен елемент даних певного типу повинен мати однаковий набір атрибутів. Широко використовується розширювана мова розмітки (XML). представляють напівструктуровані дані.



Мал. 2.12

2.2.4 SQL запити

Запит — це метод опитування бази даних SQL. Він використовується, щоб повідомити СУБД, що інформацію, яку ви хочете отримати з бази даних, а також те, як ви хочете отримати дані з'являтися. Якщо ви подумаєте про це, єдина причина для зберігання та підтримки бази даних інформації полягає в тому, щоб полегшити отримання інформації, яка вам потрібна, коли вам це потрібно це. Однією з найважливіших функцій будь-якої мови запитів є здійснення пошуку інформацію якомога простішу та потужнішу для користувача. Потреби в пошуку даних бути легким, оскільки більшість часу люди, які запитують базу даних, не є ті самі люди, які програмували базу даних. Користувачі не зацікавлені в технічні особливості того, як організована база даних або як нею керують. Запит мова повинна мати прості для розуміння (бажано простою англійською) команди, які користувачі можуть використовувати інтуїтивно. Мова запитів також має бути потужною, оскільки вона має бути здатним надавати користувачам всю необхідну інформацію. СУБД заздалегідь не знає, якими будуть запити користувача, мовні конструкції повинні бути достатньо потужними, щоб впоратися з усіма запитами користувача ймовірно зробити.

Інструкція SELECT дозволяє вказати дані, які ви хочете отримати, які порядок упорядкування даних, які обчислення виконувати з отриманими даними та багато іншого, багато інших операцій. Оскільки це єдине дієслово SQL, яке дозволяє запитувати база даних і SQL є мовою запитів, вона обов’язково є найскладнішою з усіх SQL команди. ANSI/ISO SQL допускає до шести різних пунктів у операторі SELECT з яких перші два обов'язкові. Показано синтаксис повного оператора SELECT на **Мал. 2.13**.

Простий оператор SELECT, як випливає з назви, є найпростішою формою запит, який використовує лише обов’язкові пропозиції повного SELECT. Це лише вимагає від вас надати дві відомості. По-перше, стовпці, які ви хочете бачити, а по-друге, ім’я таблиці, у якій знаходяться стовпці. Наприклад, цей запит отримує всі рядки в таблиці DEPARTMENTS:

```
SELECT DEPT_NO, DEPT_NAME, HEAD, BUDGET, P_BUDGET
  FROM DEPARTMENTS ;


DEPT_NO DEPT_NAME              HEAD      BUDGET       P_BUDGET
------- -------------------- --------- ------------ -----------
   1    Engineering             59      5780000      6200000
   2    Arts & Humanities       23       753000       643000
   3    Management Studies       3      2510000      1220000
   4    Industrial Law          12        78000       210000
```

Мал. 2.13

Записи додаються до таблиць за допомогою команди INSERT. По суті, їх двоє варіації цієї команди. По-перше, оператори INSERT, які додають записи по рядку, по-друге, оператори INSERT, які додають декілька рядків одночасно.

Синтаксис однорядкового оператора INSERT показано на **Мал. 2.14**. Наприклад, до вставте перший рядок у таблицю STUDENTS:

```
INSERT INTO STUDENTS
  (SURNAME, FIRST_NAME, D_O_B, STUDENT_NO, DEPT_NO, YEAR)
   VALUES ('Duke', 'Fitzroy', '26-NOV-70', 1, 4, 2);


1 row successfully inserted.
```

Мал. 2.14

Очевидно, щоб мати можливість додавати дані до таблиці, таблиця повинна вже бути створений за допомогою команди CREATE TABLE. Команда INSERT цього не робить створювати будь-які вихідні дані. Однак у більшості інтерактивних систем SQL СУБД повідомляє вам
чи були додані рядки, і якщо так, то скільки.

Дані можна вставляти лише в таблиці, якими володіє користувач або має привілей INSERT

на. На практиці це означає, що ви повинні створити таблицю або людину хто його створив, повинен надати вам дозвіл на вставлення даних за допомогою команди GRANT.

Список стовпців є необов'язковим у операторі INSERT. Якщо вказано список стовпців, тоді список значень повинен містити однакову кількість елементів і в тому самому порядку. Тип даних кожної пари стовпець/значення також має бути сумісним.

Команда UPDATE використовується для зміни існуючих значень стовпців. У ньому найпростіша форма UPDATE вимагає лише трьох частин інформації: імені таблиці там, де потрібні оновлення, назви стовпців, які потрібно оновити, і значення встановити стовпці на. Ви, мабуть, здогадалися за всіма літаючими навколо цього оновлення може змінити значення більш ніж одного стовпця в одному операторі.

У таблиці LECTURERS, наприклад, таке оновлення встановить зарплату для всіх викладачів до 25 тис **Мал. 2.14**:

```
UPDATE LECTURERS
  SET PAY = 25000 ;


6 rows updated.
```

Мал. 2.14

Рано чи пізно ви захочете видалити частину даних зі своїх таблиць. Це може бути неправильна інформація або зайві дані. SQL дозволяє видаляти дані за допомогою оператора DELETE.

DELETE дозволяє видалити один або кілька рядків із таблиці. Ця команда діє на цілі ряди. Це не дозволяє видаляти окремі значення полів. ви необхідно видалити цілий рядок або не видалити взагалі. Якщо використовується без предиката, DELETE видаляє всі рядки з таблиці. Очистити таблицю **Мал. 2.15, Мал. 2.16**:

```
DELETE FROM ELITE_EXAMS ;


2 rows deleted.
```

Мал. 2.15

```
DELETE FROM STUDENTS
  WHERE SURNAME = 'Wickes' AND FIRST_NAME = 'Wendy Y Y W' ;


1 row deleted.
```

Мал. 2.16

2.2.5 SQL server

Для збереження даних інформаційної системи було вибрано середовище MS SQL server.

SQL Server є однією з найпопулярніших систем управління базами даних (СУБД) у світі. Ця СУБД підходить для різних проектів: від невеликих додатків до великих високонавантажених проектів.

SQL Server характеризується такими особливостями як:

Продуктивність. SQL Server працює дуже швидко. Надійність та безпека. SQL Server надає шифрування даних.Простота. З цієї СУБД щодо легко працювати та вести адміністрування.

Центральним аспектом у MS SQL Server, як і будь-який СУБД, є база даних. База даних представляє сховище даних, організованих певним способом. Нерідко фізично база даних представляє файл на жорсткому диску, хоча така відповідність необов'язково. Для зберігання та адміністрування баз даних застосовуються системи управління базами даних (database management system) або СУБД (DBMS). І саме MS SQL Server є однією з таких СУБД.

Для організації баз даних MS SQL Server використовує реляційну модель. Ця модель на сьогодні вона фактично є стандартом для організації баз даних.

Виділяються два різновиди мови SQL: PL-SQL та T-SQL. PL-SQL використовується у таких СУБД як Oracle та MySQL. T-SQL (Transact-SQL) застосовується у SQL Server.

2.3 Віртуальні машини

Для тестування програмного продукту будуть використовуватися віртуальні машини.

Віртуальна машина — модель обчислювальної машини, створеної шляхом віртуалізації обчислювальних ресурсів: процесора, оперативної пам'яті, пристроїв зберігання та вводу і виводу інформації.

Віртуальна машина, на відміну від програми емуляції конкретного пристрою, забезпечує повну емуляцію фізичної машини чи середовища виконання (для програми).

Системні віртуальні машини дозволяють розподіл апаратних ресурсів фізичної машини між різними копіями віртуальних машин, на кожній з яких може бути встановлена своя операційна система. Пласт програмного забезпечення, що виконує віртуалізацію, називається гіпервізором.

Основні переваги системних ВМ:

- різні операційні системи можуть співіснувати на одному комп'ютері, і при цьому знаходитися в строгій ізоляції одна від одної
- ВМ можуть забезпечувати розширений набір машинних інструкцій, адже при моделюванні абстрактної обчислювальної машини набір інструкцій процесора віртуальної машини може бути довільним.
- широкі можливості контролю за програмами
- легкість модифікацій та відновлення

Основний недолік:

- віртуальна машина не така ефективна як реальна, тому що доступ до апаратури в ній відбувається опосередковано.

2.4 Висновок до другого розділу

У цьому розділі ми детально розібрали аналогові програмні продукти, їх плюси та недоліки і в цілому функціонал цих інформаційних систем.

Також дізналися про основні принципи роботи, конструювання та моделювання та функціоналу СУБД, яка є невід'ємною і дуже важливою частиною інформаційної системи.

Зробили коротке введення у середовище MS SQL server та технологію віртуальних машин, які допоможуть в розробці та тестуванні програмного продукту.

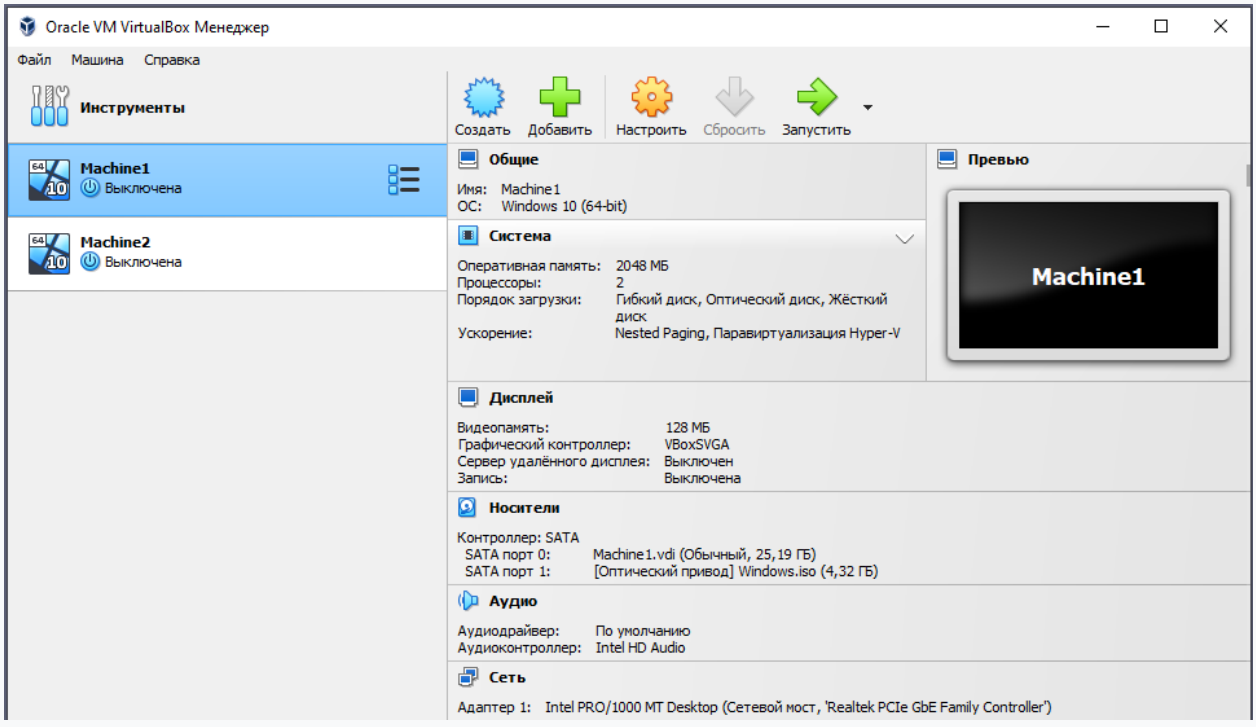
РОЗДІЛ 3. ПРОГРАМНА РЕАЛІЗАЦІЯ

3.1 Допоміжні програми та середовище розробки

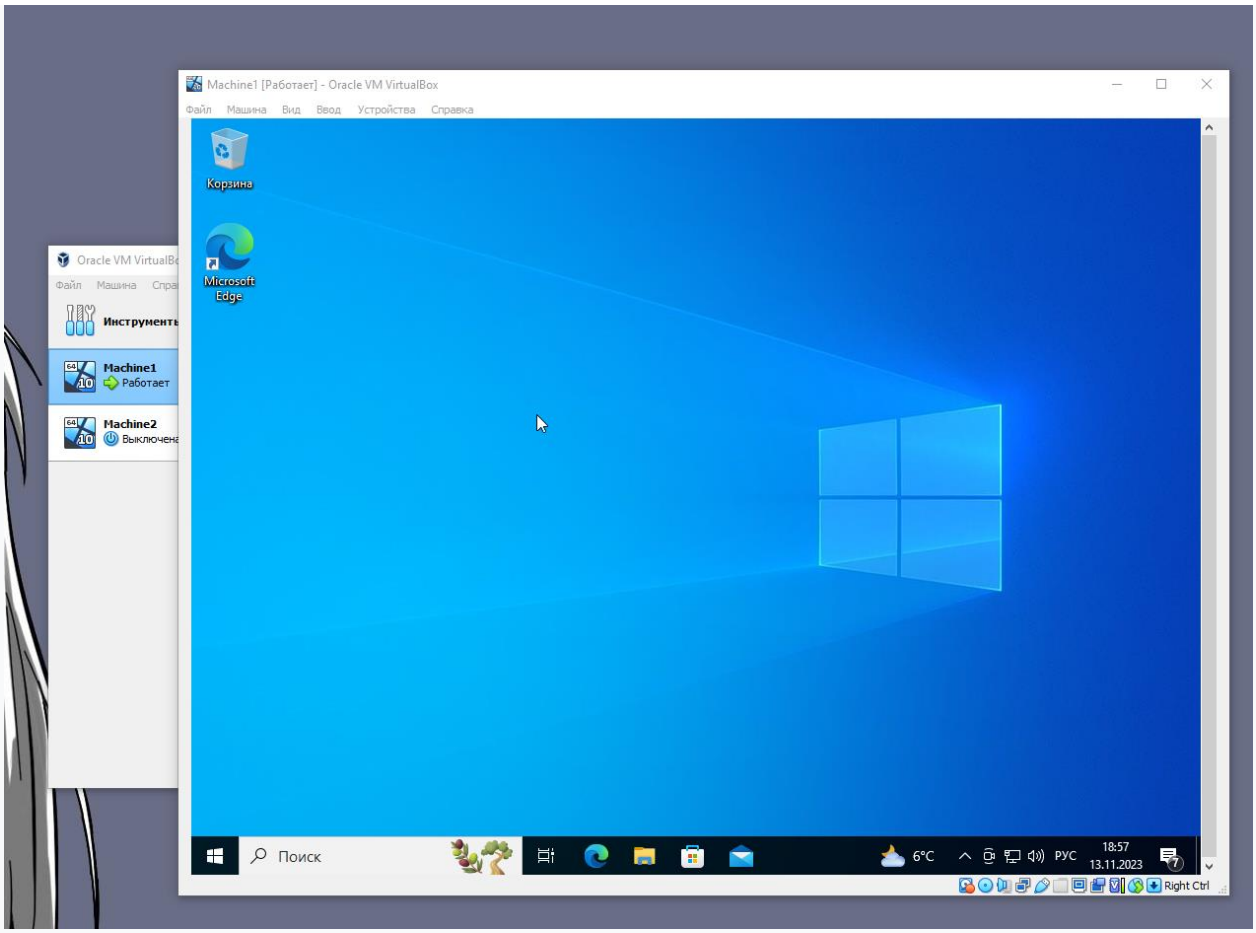
Virtual Box

VirtualBox — це потужний продукт віртуалізації x86 і AMD64/Intel64 для корпоративного та домашнього використання. VirtualBox — це не тільки надзвичайно багатофункціональний високопродуктивний продукт для корпоративних клієнтів, це також єдине професійне рішення, яке є у вільному доступі як програмне забезпечення з відкритим вихідним кодом згідно з умовами GNU General Public License (GPL) версії 3.

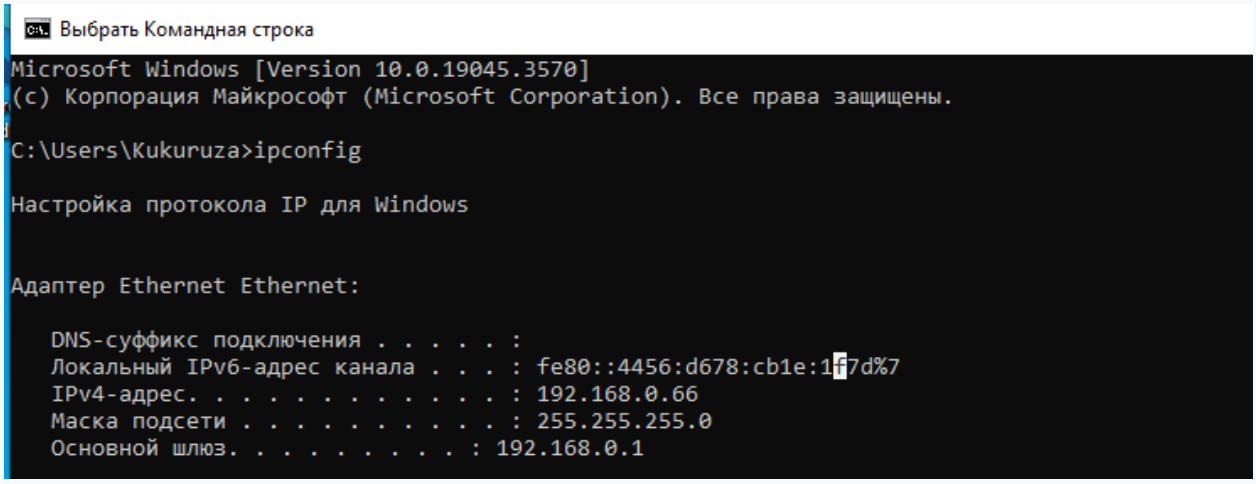
Використовувалось для емуляції машин у локальній мережі. На обидвах машинах була встановлена операційна система Windows 10 з встановленим IP адресом, ім’ям комп’ютера та підключенням до робочої групи **Див. Скрін.3.1, Скрін.3.2**, **Скрін.3.3, Скрін.3.4**.



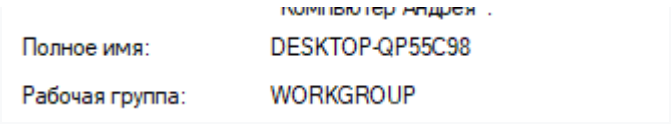
Скрін.3.1



Скрін.3.2



Скрін.3.3

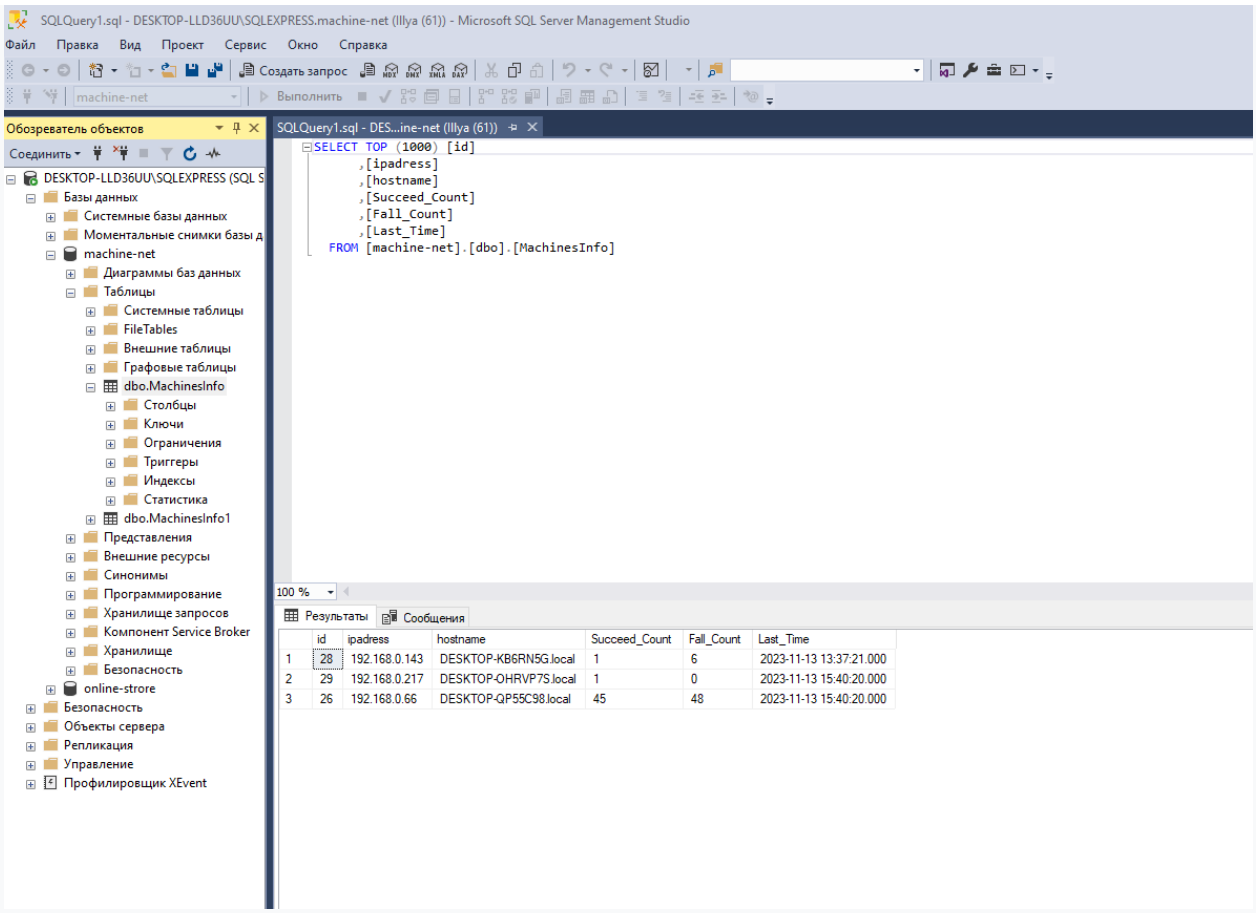


Скрін.3.4

SQL Server Management Studio

Для створення бази даних та таблиці у яких будуть зберігатися дані про пристрої у локальній мережі, використовувався.

SQL Server Management Studio (SSMS) - це інтегроване середовище для управління будь-якою інфраструктурою SQL, від SQL Server до баз даних SQL Azure. SSMS надає засоби для налаштування, спостереження та адміністрування екземплярів SQL Server та баз даних. Використовуйте SSMS для розгортання, моніторингу та оновлення компонентів рівня даних, що використовуються програмами, та створення запитів та скриптів **Див. Скрін.3.5**



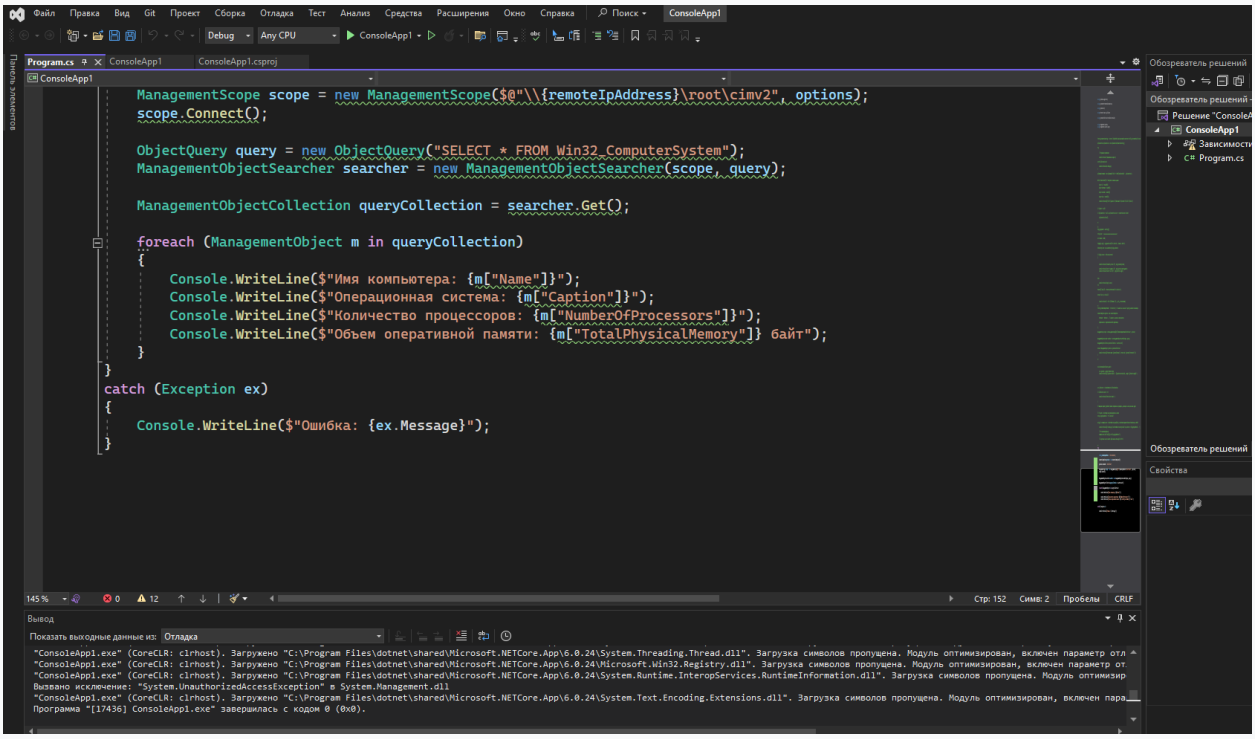
Скрін.3.5

Visual Studio

Було вибрано середовище Visual Studio для розробки програмного продукту.

Microsoft Visual Studio - лінійка продуктів компанії Microsoft, що включають інтегроване середовище розробки (IDE) програмного забезпечення та низку інших інструментів. Дані продукти дозволяють розробляти як консольні програми, так і ігри та програми з графічним інтерфейсом, у тому числі з підтримкою технології Windows Forms, UWP а також веб-сайти, веб-додатки, веб-служби як в рідному, так і в керованому коді всіх платформ, підтримуваних Windows, Windows Mobile, Windows CE, .NET Framework, .NET Core, .NET, MAUI, Xbox, Windows Phone .NET Compact Framework та Silverlight. Після покупки компанії Xamarin

корпорацією Microsoft з'явилася можливість розробки IOS та Android програм **Див. Скрін.3.6.**



Скрін.3.6

3.2 Розробка

При розробці було використано мову програмування C# та простори імен **System.Net.NetworkInformation, System.Data.SqlClient, System.Timer Див. Скрін.3.7.**

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading;
using System.Threading.Tasks;
using System.Windows.Forms;
using System.Net;
using System.Net.NetworkInformation;
using System.Net.Sockets;
using System.Xml.Linq;
using System.Data.SqlClient;
using static System.Windows.Forms.VisualStyles.VisualStyleElement;
using System.Xml;
using System.Timers;
using System.Runtime.Remoting.Contexts;
using static System.Net.WebRequestMethods;
```

Скрін.3.7

System.Net.NetworkInformation - надає доступ до даних мережного трафіку, відомостей про мережні адреси та сповіщень про зміну адрес для локального комп'ютера. Цей простір імен містить класи, що реалізують функціональність програми Ping. Клас Ping та інші пов'язані з ним класи можуть бути використані для перевірки доступності комп'ютера по мережі.

Ping - дозволяє застосунку визначити, чи доступний віддалений комп'ютер через мережу. Цей клас надає функціональні можливості, аналогічні засобу командного рядка Ping.exe. Методи Send і SendAsync надсилають повідомлення запиту на луну ICMP на віддалений комп'ютер і очікують відповіді на луну ICMP з цього комп'ютера.

ICMP - мережевий протокол, що входить у стек протоколів TCP/IP. В основному ICMP використовується для передачі повідомлень про помилки та

інші виключні ситуації, що виникли при передачі даних, наприклад, запитувана послуга недоступна або хост, або маршрутизатор не відповідають. Також ICMP покладаються деякі сервісні функції (services).

System.Data.SqlClient - простір імен є постачальником даних .NET платформи для SQL Server.

System.Timer - Створює подію після заданого інтервалу з можливістю створення подій, що повторюються.

Запит на підключення до бази даних та відкриття асинхронного підключення за допомогою функції OpenAsync() **Див Скрін.3.8**

```
private async void Form1_Load(object sender, EventArgs e)
{
    // Підключення до бази данних та відкриття підключення
    string connectionString = "Server=.\\SQLEXPRESS;" +
        "Database=machine-net;User Id=Illya;Password=half;" +
        "Trusted_Connection=True;TrustServerCertificate=True";

     sqlConnection = new SqlConnection(connectionString);

    try
    {
        await sqlConnection.OpenAsync(); // Відкриття асинхроного підключення

    }
    catch(SqlException ex)
    {
```

Див Скрін.3.8

Запит на зчитування з таблиці даних **Див Скрін.3.9**

```
// Зчитування данних з таблиці
    SqlCommand command = new SqlCommand("SELECT * FROM [MachinesInfo]",
        sqlConnection);
    SqlDataReader reader = await command.ExecuteReaderAsync();

// Відображеня данних машин у додатку
try
{

while (await reader.ReadAsync())
{
    object ip = reader.GetValue(1);
    object hostname = reader.GetValue(2);
    object succeed = reader.GetValue(3);
    object fall = reader.GetValue(4);
    object data = reader.GetValue(5);
```

Скрін.3.9

Запит на записування нових даних у таблицю **Див Скрін.3.10**

```
SqlCommand command = new SqlCommand("INSERT INTO MachinesInfo (ipadress, hostname) VALUES(

command.Parameters.AddWithValue("IP", row.Cells[0].Value.ToString());
command.Parameters.AddWithValue("Host", row.Cells[1].Value.ToString());
await command.ExecuteNonQueryAsync();
```

Скрін.3.10

Запит на видалення даних в таблиці **Див Скрін.3.11**

```
SqlCommand command = new SqlCommand("DELETE FROM MachinesInfo WHERE ipadress = @IP"
    , sqlConnection);

 command.Parameters.AddWithValue("IP", row.Cells[0].Value.ToString());

 await command.ExecuteNonQueryAsync();
```

Скрін.3.11

Запит Ping на перевірку підключення пристрою до мережі і відразу отримання відповіді, за допомогою класу PingReply **Див Скрін.3.12**

```
ping = new Ping();
pingReply = ping.Send(textBox1.Text);


if (pingReply.Status == IPStatus.Success)
{


    abbr = IPAddress.Parse(textBox1.Text);
    host = Dns.GetHostEntry(abbr);
    name = host.HostName;

    dataGridView1.Rows.Add(textBox1.Text, name, "Active");
```

Скрін.3.12

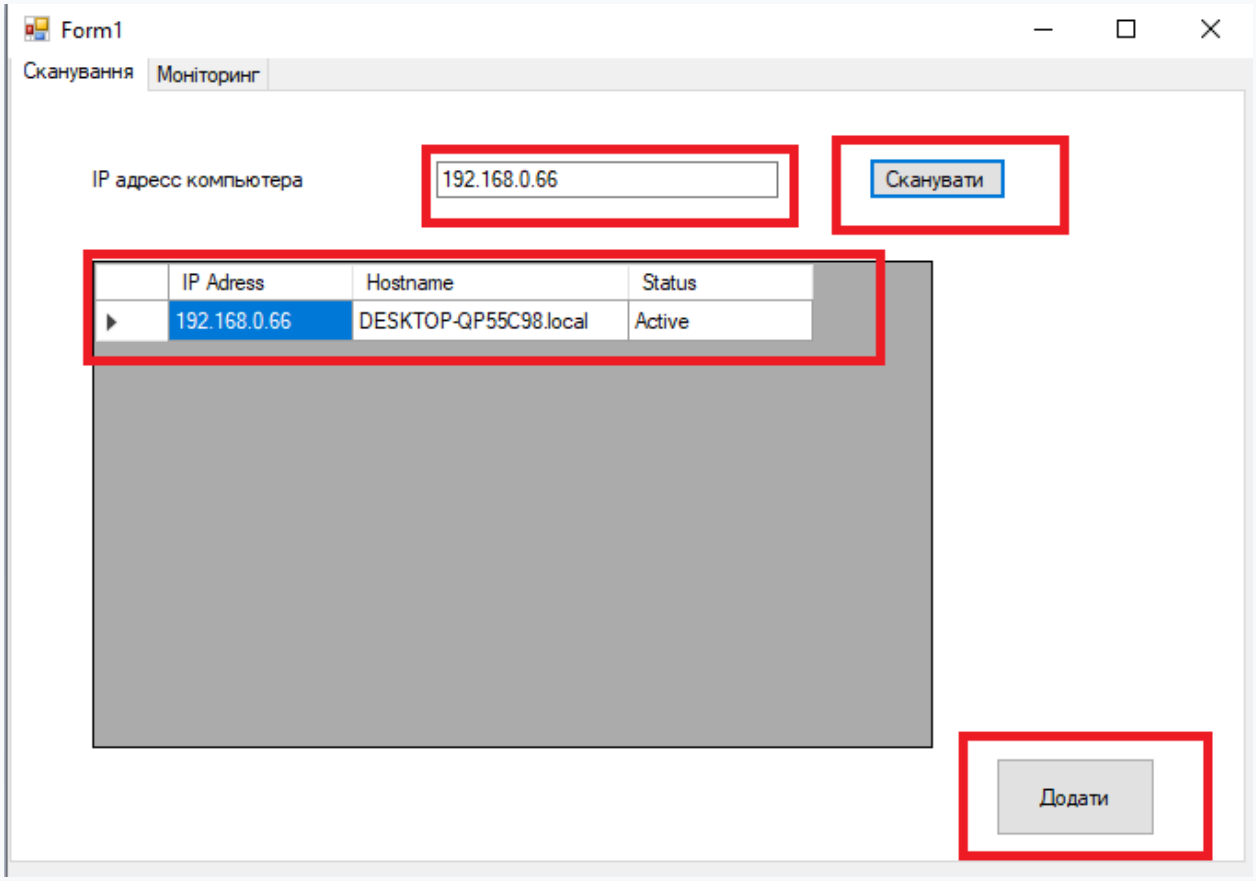
Написання таймеру, який по закінченню часу викликає функцію, яка оновлює інформацію у додатку **Див Скрін.3.13**

```
    //Задаття таймеру та виклик функції яка виконується коли час інтерв
    System.Timers.Timer timer = new System.Timers.Timer(10000);
    timer.Enabled = true;
    timer.Elapsed += timer_Tick;
}
```

Скрін.3.13

3.3 Тестування

З відкриттям додатку користувач опиняється на першій вкладці “Сканування”. Тут ми маємо можливість сканувати мережу по введеному користувачеві IP адресу. Також після того як програма знайшла пристрій ми маємо можливість додати його до бази даних для подальшого моніторингу **Див. Скрін. 3.14.**



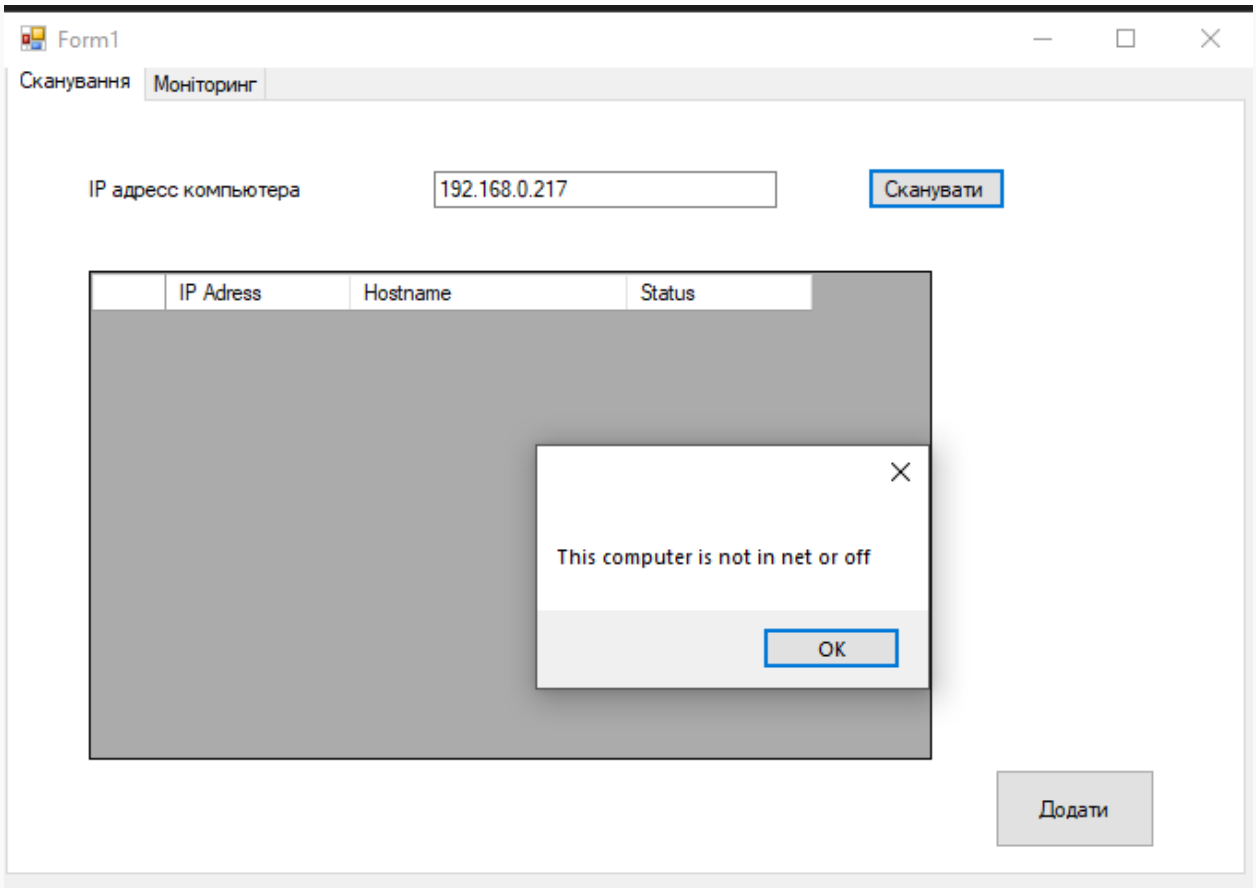
Скрін. 3.14

Також для перевірки заходимо у середовище бази даних і бачимо у таблиці, що вся інформація зберігається **Див. Скрін. 3.15.**

	id	ipadress	hostname	Succeed_Count	Fall_Count	Last_Time
1	28	192.168.0.143	DESKTOP-KB6RN5G.local	1	14	2023-11-13 13:37:21.000
2	30	192.168.0.217	DESKTOP-OHRVP7S.local	2	0	2023-11-13 21:24:56.000
3	26	192.168.0.66	DESKTOP-QP55C98.local	50	48	2023-11-13 21:24:56.000

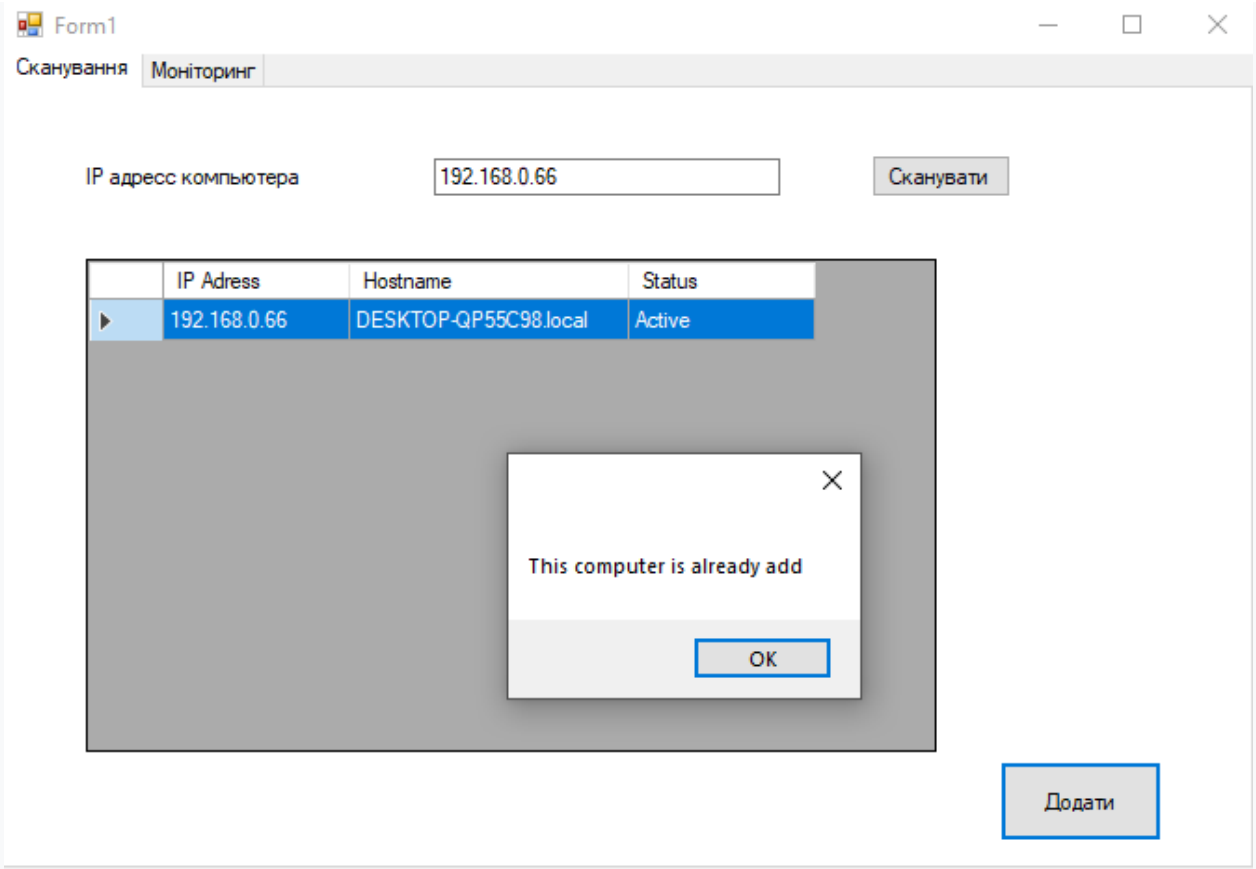
Скрін. 3.15

Якщо машина, IP адреса якої була введена, не відповідає по будь-яким причинам, програма нас про це сповіщає **Див. Скрін. 3.16** .



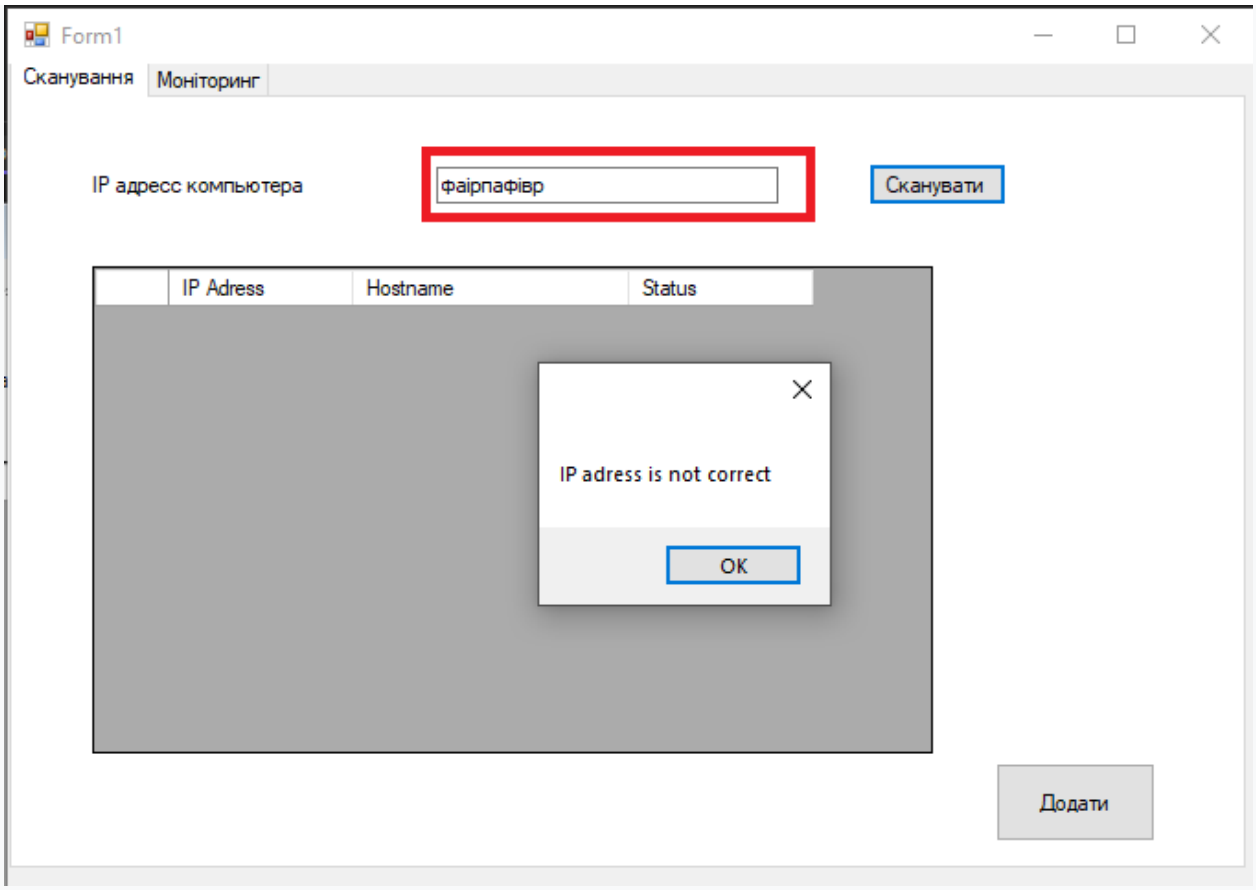
Скрін. 3.16

Якщо ми хочемо додати комп’ютер, який вже є у базі даних, алгоритм нам цього не дозволить **Див. Скрін. 3.17** .



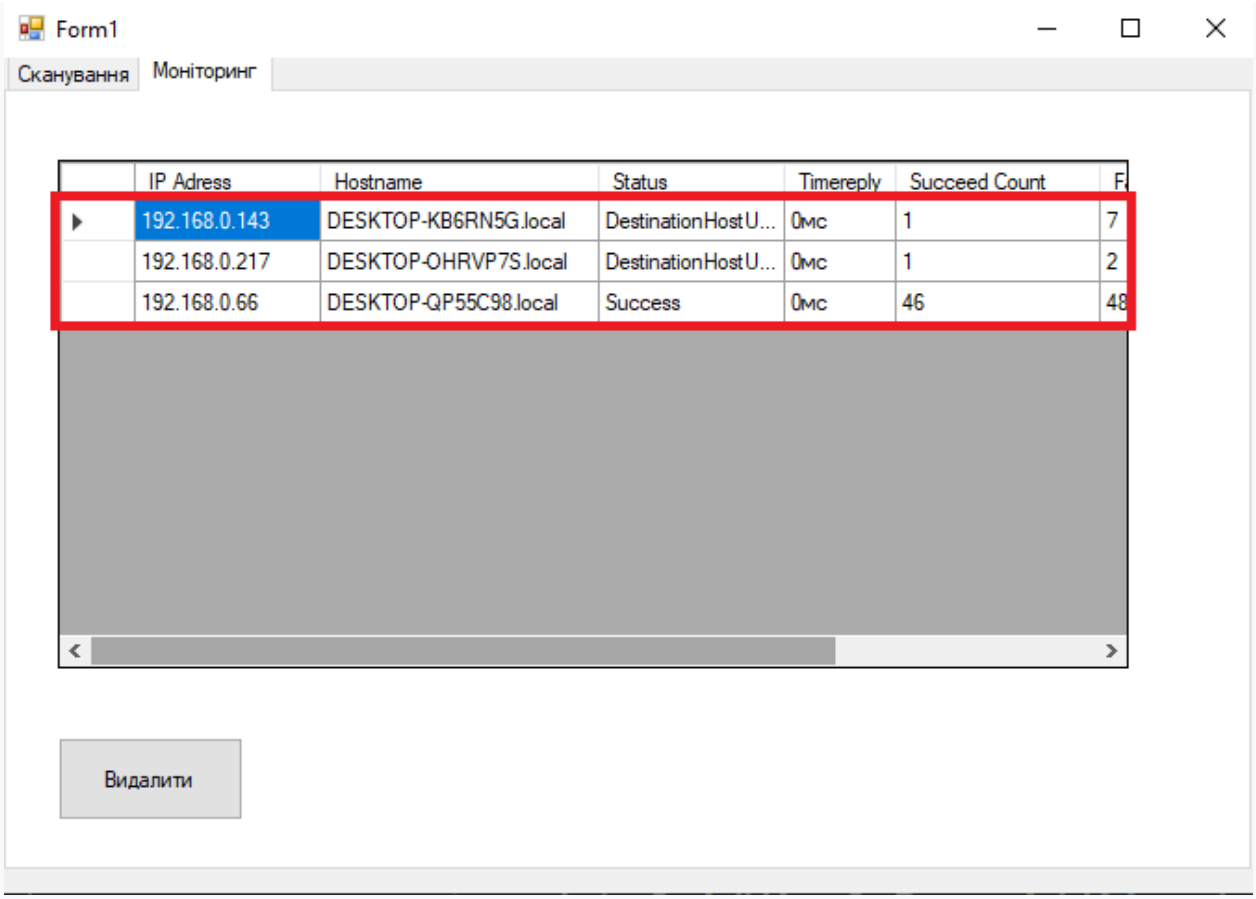
Скрін. 3.17

Якщо IP адреса введена некоректно, програма теж сповіщає про це користувача **Див. Скрін. 3.18.**

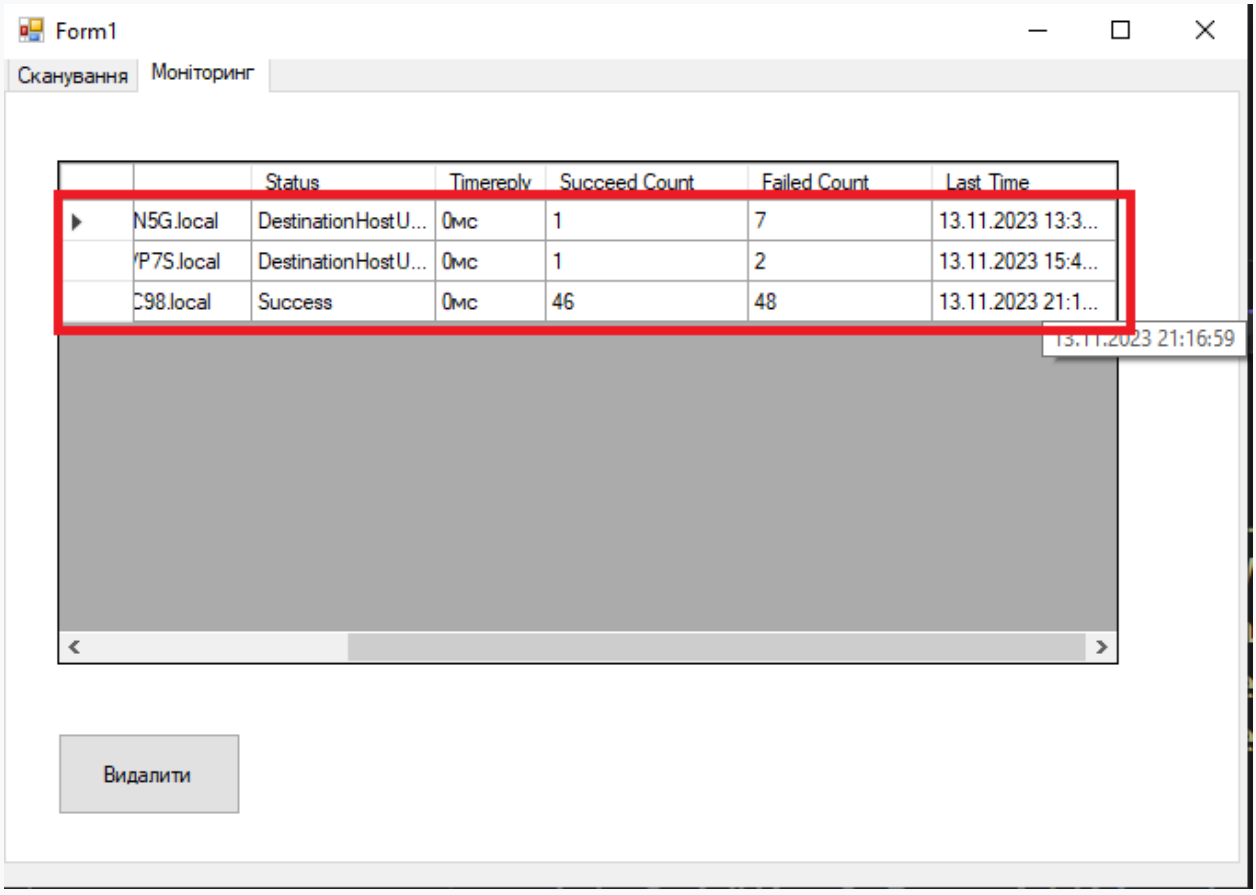


Скрін. 3.18

На другій вкладці “Моніторинг” ми маємо можливість спостерігати за машинами у мережі в режимі реального часу. Зокрема ми бачимо не тільки IP адресу і ім’я комп’ютера, а й статус підключення, час витрачений на посилання запиту і отримання відповіді, статистику успішних підключень і неуспішних та останній час використання **Див. Скрін. 3.19., Скрін. 3.20**.

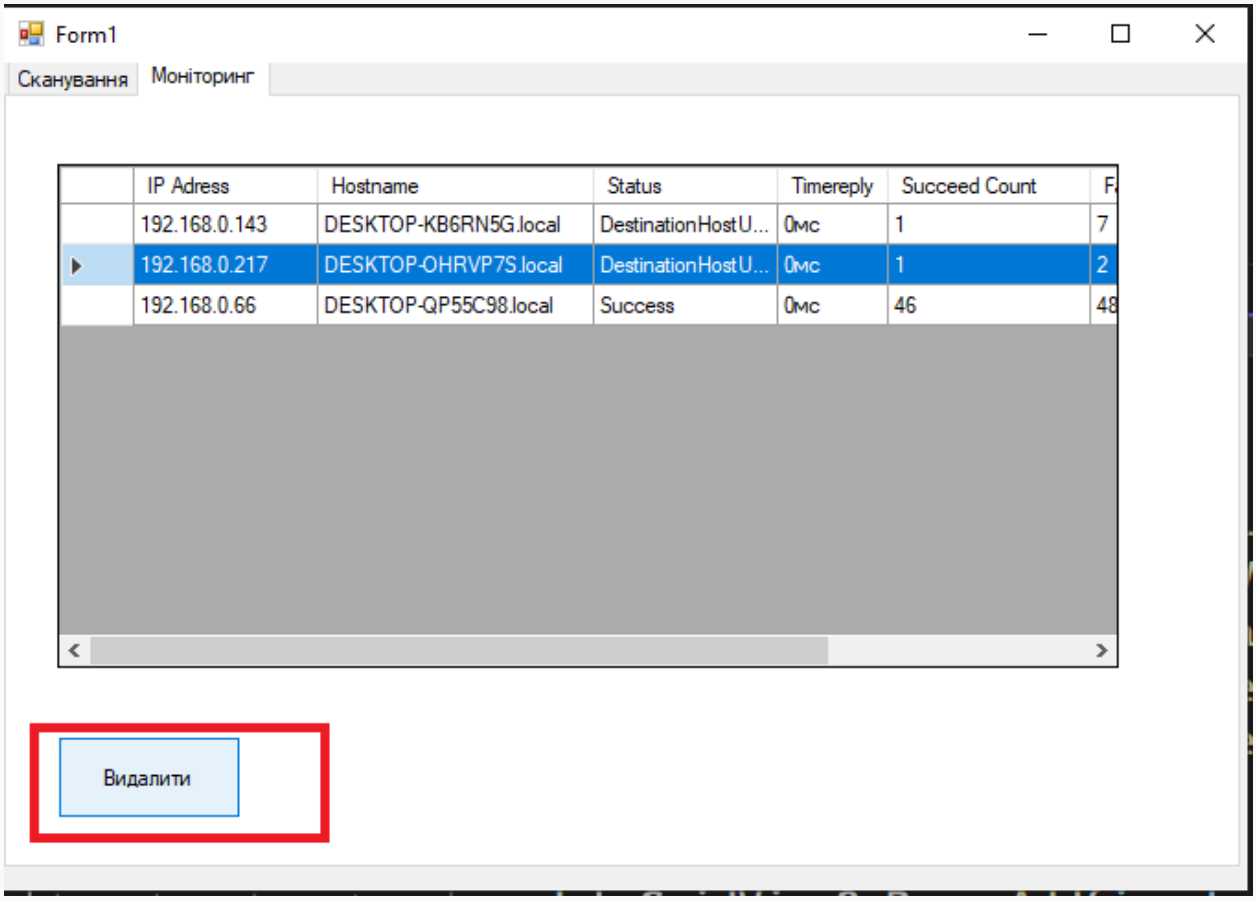


Скрін. 3.19



Скрін. 3.20

Також користувач має можливість видалення пристрою з вкладки “Моніторинг” і з бази даних **Див. Скрін. 3.21., Скрін. 3.22.**



Скрін. 3.21

	id	ipadress	hostname	Succeed_Count	Fall_Count	Last_Time
1	28	192.168.0.143	DESKTOP-KB6RN5G.local	1	8	2023-11-13 13:37:21.000
2	26	192.168.0.66	DESKTOP-QP55C98.local	47	48	2023-11-13 21:19:06.000

Скрін. 3.22

3.4 Висновок до третього розділу

У цьому розділі ми оглянули програми, за допомогою яких було розроблено інформаційну систему.

Подивилися та розібралися у програмному коді, написаному на C# та бібліотеках.

Також здійснили тестування і побачили, як програма працює і що вона працює коректно.

ВИСНОВКИ

За результатами виконання дипломної магістерської роботи було розглянуто чимало тем і частин з яких складається Інформаційна система.

Було проведено введення в тему і поняття Інформаційної системи. Розглядалися причини появи ІС, їх моделювання та основні принципи розробки і супроводження. Ще важливо було відмічене їх роль у підприємстві та витрати ресурсів на їх розробку і супровід.

Дізналися про комп'ютерні мережі. Досліджено принцип їх роботи та необхідність у роботі підприємств.

Також не менш важливу роль в ІС грають бази даних. Вміння правильно структурувати, моделювати та розробляти бази даних – дуже важлива частина для цього проекту.

Дослідили аналоги і побачили наскільки ІС з контролю мереж користуються попитом та продовжують розвиватись, хоча і мають свої недоліки.

У кінці демонстрація технологій для розробки і самої розробки програмного продукту.

Можна сказати що розробка інформаційної системи це в деяких аспектах складна проте важлива річ, до якої треба підходити з розумінням. А також що ІС з ухилом у комп'ютерні мережі є дуже зручним інструментом в організації робочого процесу на підприємствах та в компаніях.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Information Systems for Business An Experiential Approach France Bélanger, Craig Van Slyke, Robert E. Crossler
2. Computer Networks a systems approach Larry L. Peterson and Bruce S. Davie
3. TCP/IP Illustrated, Volume 1: The Protocols" by W. Richard Stevens.
4. Database System Concepts" by Abraham Silberschatz, Henry F. Korth, and S. Sudarshan.
5. SQL Performance Explained" by Markus Winand.
6. КОМП'ЮТЕРНІ МЕРЕЖІ,принципи, технології, протоколи В.Г. Оліфер, Н.А. Оліфер
7. Network Security Essentials Applications and Standards sixth edition William Stallings
8. https://uk.wikipedia.org/wiki/Інформаційні_системи
9. https://learn.microsoft.com/ru-ru/sql/ssms/download-sql-server-management-studio-ssms?view=sql-server-ver16
10. https://learn.microsoft.com/ru-ru/dotnet/api/system.timers.timer?view=net-7.0
11. https://metanit.com/sql/sqlserver/1.1.php
12. https://www.softinventive.com.ua/best-network-monitoring-tools
13. https://www.virtualbox.org/
14. https://timeweb.com/ru/community/articles/lokalnaya-set-windows-10-nastroyka
15. Clean Code: A Handbook of Agile Software Craftsmanship" by Robert C. Martin.
16. Head First Software Development: A Learner's Companion to Software Development" by Dan Pilone, Russ Miles.

17. Code Complete: A Practical Handbook of Software Construction" by Steve McConnell.

18. High-Performance Browser Networking" by Ilya Grigorik

19. Computer Networking: A Top-Down Approach" by James F. Kurose, Keith W. Ross.

20. Database Design for Mere Mortals: A Hands-On Guide to Relational Database Design" by Michael J. Hernandez.

21. MongoDB: The Definitive Guide" by Kristina Chodorow and Michael Dirolf.

22. Authentication: A Secure Way to Protect Network" by Asim Kumar Mandal.

23. ISO/IEC 27001:2013– Стандарт з управління інформаційною безпекою

24. Комп'ютерна безпека. Архітектура та програмні засоби" Майкл Е. Вітман, Херберт М. Матесон

25. The Web Application Hacker's Handbook: Finding and Exploiting Security Flaws" by Dafydd Stuttard, Marcus Pinto.

ДОДАТОК А

ДОДАТОК Б