Національний університет «Полтавська політехніка імені Юрія Кондратюка»
(повне найменування закладу вищої освіти)

Навчально-науковий інститут інформаційних технологій і робототехніки
(повне найменування інституту, назва факультету (відділення))

Кафедра автоматики, електроніки та телекомунікацій
(повна назва кафедри (предметної, циклової комісії))

Пояснювальна записка

до кваліфікаційної роботи

магістр

(ступінь вищої освіти)

на тему **Розроблення проекту інфокомунікаційної мережі комерційного підприємства**

Виконав: студент 6 курсу, групи 601ТТ
спеціальності 172 «Телекомунікації та
(шифр і назва напряму підготовки, спеціальності)
радіотехніка

Терьошкін М.Ю.
(прізвище та ініціали)
Керівник Сокол Г.В.
(прізвище та ініціали)

(прізвище та ініціали)

Полтава - 2022 рік

Національний університет «Полтавська політехніка імені Юрія Кондратюка»
Інституту Навчально-науковий інститут інформаційних технологій і робототехніки
Кафедра Автоматики, електроніки та телекомунікацій
Ступінь вищої освіти Магістр
Спеціальність 172 «Телекомунікації та радіотехніка»

ЗАТВЕРДЖУЮ

Завідувач кафедри автоматики, електроніки та телекомунікацій

_____ О.В. Шефер
“ ___ ” _____ 2022 р.

З А В Д А Н Н Я

НА МАГІСТЕРСЬКУ РОБОТУ СТУДЕНТУ

Терьошкіну Максиму Юрійовичу

1. Тема проекту (роботи) **«Розроблення проекту інфокомунікаційної мережі комерційного підприємства»**
керівник проекту (роботи) Сокол Галина Вікторівна, к.т.н., доцент
затверджена наказом вищого навчального закладу від “12 ” 08 2022 року № 544 фа
2. Строк подання студентом проекту (роботи) 07.12.2022 р.
3. Вихідні дані до проекту (роботи) телекомунікаційних мереж та захисту інформації.
4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити) Аналіз та проблеми постановки задач проекту. Проектування локальної мережі. Практична частина розробки інформаційного захисту. Розроблення проекту інфокомунікаційної мережі комерційного підприємства.
5. Перелік графічного матеріалу (на рисунках):
   1) Структура сучасних мереж.
   2) Вінформаційна система.
   3) Передавання даних з використанням семирівневої моделі відкритих систем;
   4) Обробка фреймів пакету на різних рівнях чотирирівневої моделі зв’язку відкритих систем DoD.
   5) План будівлі.
   6) Функціональна схема розташування комп’ютерів.
   7) Кручена пара.
   8) Конектор RJ-45.
   9) Схема мережі в програмі Packet Tracer.
   Дата видачі завдання 01.09.2022 р.

КАЛЕНДАРНИЙ ПЛАН

Пор. №	Назва етапів магістерської роботи	Термін виконання етапів роботи			Примітка (плакати)
1	Принципи роботи локальної мережі	13.09.22		15%	Пл. 1
2	Системи захисту інформації та організаційний і технічний захист	27.09.22	I	30%	Пл. 2
3	Дослідження локальної мережі ПП «Молтехпром»	10.10.22		40%	Пл. 3
4	Аналіз діючої локальної мережі підприємства	17.10.22		50%	Пл. 4
5	Проектування локальної мережі підприємства	24.10.22	II	60%	Пл. 5
6	Створення програми шифрування даних з використанням алгоритму шифрування RSA	08.11.22		70%	Пл. 6
7	Розрахунок споживаної потужності та вартості системи	07.12.22	III	100%	Пл. 7

Магістрант _____ Терьошкін М.Ю.
(підпис) (прізвище та ініціали)

Керівник роботи _____ Сокол Г.В.
(підпис) (прізвище та ініціали)

ЗМІСТ

ВСТУП

Спілкування є значущим аспектом соціальної взаємодії, однією з найзагальніших ознак будь-якої діяльності. Комунікацію можна визначити як форму спілкування, як один із проявів обміну інформацією між живими істотами, у процесі їх безпосереднього спілкування або за допомогою технічних засобів. Дослідження показують, що співробітники, які займаються проєктною діяльністю, витрачають на спілкування 50-80% свого часу. Це здається чимало, але стає зрозуміло, якщо врахувати, що працівник робить це постійно, щоб виконувати свої ролі в міжособистісних стосунках, обміні інформацією та процесах прийняття рішень. Тому важливо правильно організувати обмін інформацією на підприємстві, найпростішим і надійним способом виконання цього завдання є організація комп’ютерної мережі на підприємстві.

Найважливішою перевагою, яка забезпечила поширення комп'ютерних мереж, є можливість віртуальної роботи з інформацією. При цьому сама інформація може зберігатися в одній або декількох точках мережі і бути доступною з робочого місця будь-якого співробітника. Перевагою використання комп’ютерної мережі на підприємстві також є можливість спільного використання апаратних і програмних ресурсів, що значно скорочує витрати на забезпечення виробничого процесу.

Важливим аспектом взаємодії в комп'ютерній мережі є забезпечення інформаційної безпеки. Основні проблеми захисту інформації при роботі в комп'ютерних мережах можна умовно розділити на три типи: перехоплення, модифікація інформації та підміна авторства. Вирішення проблем захисту інформації в електронному вигляді базується на використанні таких методів захисту інформації, як: технічні, інженерні, організаційні та криптографічні.

Актуальність створення інфокомункаційної мережі. Це невід’ємна частина сучасного світу, бізнесу, політики, державних і регіональних органів. Головним завданням є забезпечення безпеки інформації, розмежування

доступу, авторизація користувача і даних. Найбільш актуальним є питання безпеки інформації для корпоративної організацій і підприємств.

Об’єкт дослідження: приватне підприємство «Молтехпром».

Предмет дослідження: інфокомунікаційна мережа підприємства.

Метою дипломної роботи є проектування локальної мережі, створення програми шифрування даних з використанням алгоритму шифрування RSA.

Постановка задачі. Задачею дослідження є проектування локальної мережі та вибір і обґрунтування використання алгоритму шифрування RSA.

Призначення інфокомунікаційної мережі. Розроблено проект мережі для одноповерхової будівлі комерційного піприємста «Молтехпром». Необхідне обладнання підібрано в залежності від потреб співробітників компанії, робочі станції різної продуктивності, а також веб і файлові сервери.

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ ТА СКОРОЧЕНЬ

КМ – комп’ютерна мережа

ЛОМ – локальна обчислювальна мережа

ЗХ – захист інформації

ОЗ – організаційний захист

ТЗ – технічний захист

ІС – інформаційна система

ОС – операційна система

ПК – персональний комп’ютер

ПЗ – програмне забезпечення

RSA – криптографічний алгоритм з відкритим ключем

РОЗДІЛ 1

АНАЛІЗ ПРОБЛЕМИ ТА ПОСТАНОВКА ЗАДАЧІ ПРОЕКТУ

1.1 Теоретичні відомості побудови комп’ютерних мереж

Комп'ютерна мережа - це система зв'язку між двома або більше комп'ютерами. У широкому розумінні комп'ютерна мережа - це система зв'язку через кабельне або повітряне середовище, комп'ютери різного функціонального призначення і мережеві пристрої. Як правило, для передачі інформації можуть бути використані різні фізичні явища - різного роду електричні сигнали або електромагнітне випромінювання.

Середовищем передачі в комп'ютерних мережах можуть бути телефонні кабелі та спеціальні мережеві кабелі: коаксіальні кабелі, кручені пари, оптоволоконні кабелі, радіохвилі, світлові сигнали.

Безпосередньою основою комп'ютерних мереж (КОМ) були телефонні і телеграфні мережі, в результаті розвитку мікроелектроніки з'явилися потужні електронно-обчислювальні машини, взаємодія яких вимагала швидкого і надійного каналу передачі даних.

Сучасні комп’ютерні мережі забезпечують:

• колективна обробка даних користувачами

• обмін файлами та іншими даними між користувачами

• ділитися програмами

• спільне використання принтерів, модемів тощо.

Для класифікації комп'ютерних мереж використовуються різні ознаки, вибір яких полягає в тому, щоб вибрати з існуючого різноманіття ті, які нададуть унікальні властивості цій класифікаційній схемі.

Комп'ютерні мережі класифікуються за такими ознаками:

• за територіальним розташуванням – локальні, регіональні, глобальні;

• за сферою використання – офісні, промислові, побутові;

• комплекс архітектурних рішень – Ethernet, Token Ring, Arcnet;

• за топологією - шина, кільце, зірка, дерево, повноз'єднані;

• за фізичним середовищем передачі - з симетричним кабелем, з коаксіальним кабелем, з кабелем "кручена пара", з оптичним кабелем, з інфрачервоним каналом, з мікрохвильовим каналом;

• за способом доступу до фізичного середовища передачі - з опитуванням, з маркерним доступом, з конкуренцією, з параметрами реєстру.

• за ознакою структурно-функціональної організації.

Певна неузгодженість вимог класифікації значно ускладнює завдання вибору раціональної схеми класифікації КМ. КМ класифікують переважно за структурно-функціональною організацією.

За призначенням КМ поділяються на:

• обчислювальна техніка;

• інформативний;

• змішаний (інформаційно-обчислювальний)

До недавнього часу побудова мережі обмежувалося лише територіальними масштабами, але сьогодні, з розвитком технологій, це змінилося, і тепер ці територіальні масштаби стали однаковими як для локальних мереж, так і для глобальних мереж, які відрізняються лише технологічними можливостями побудова мережі [1].

Комп'ютерні мережі поділяються на:

• локальна мережа;

• глобальна мережа.

Локальні мережі — це мережі з максимальною відстанню між вузлами не більше 1–2 км.

Глобальні мережі – мережі, що охоплюють територію країни або кількох країн з максимальною відстанню між окремими вузлами в тисячі кілометрів. Структура сучасних глобальних мереж наведена на рисунку 1.1.

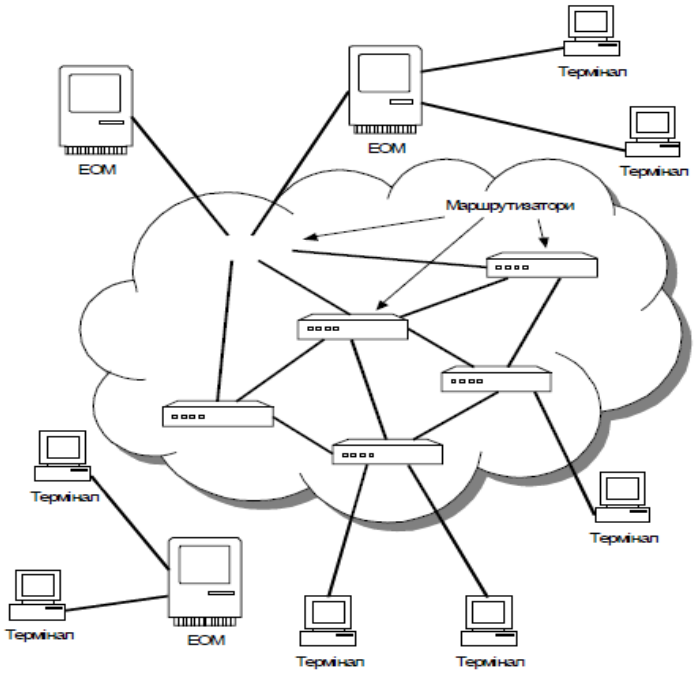

Рисунок 1.1 – Структура сучасних мереж

1.1.1 **Топологія локальних мереж**. Найбільш часто використовувана класифікація мереж базується на їх топології. Розрізняють фізичну та логічну топології. Фізична топологія визначає тип використовуваного кабелю, а також спосіб його прокладання. Логічна топологія описує шлях, по якому проходять сигнали в мережі. Незважаючи на зовнішню схожість цих термінів, насправді вони описують різні речі. Нижче наведено короткий список найпоширеніших топологій локальних мереж:

• шина;

• кільце;

• зірка.

Мережі шинної топології

Локальна мережа, побудована за шинною топологією, характеризується властивістю безпосередності. Мережевий кабель прокладається послідовно, від комп'ютера до комп'ютера.

У мережах такого типу використання термінатора (кінцеве навантаження шини) є обов'язковим. Цей пристрій характеризується можливістю дублювання сигналу, що заважає роботі мережі. Один кінець шини повинен бути заземлений.

Мережі з топологією шини зазвичай використовують тонкий або товстий коаксіальний кабель (Iobase2 або Iobase5). З'єднання такого кабелю з мережевими адаптерами, встановленими в комп'ютерах, здійснюється за допомогою Т-подібних перехідників.

У процесі роботи мережі повідомлення такого типу, що надсилаються кожним комп'ютером, приймаються всіма підключеними до шини комп'ютерами. Заголовки повідомлень аналізуються мережевими адаптерами. У процесі аналізу визначається комп’ютер одержувача цього повідомлення.

Мережі шинної топології мають такі переваги:

• Простота реалізації;

• відносна дешевизна.

Нижче наведено короткий опис недоліків мереж шинної топології:

• пасивний характер, що призводить до значного ослаблення сигналу;

• вразливість мережі (одна загальна шина).

Мережі з кільцевою топологією

Якщо ми з'єднаємо кінці шини, то отримаємо класичний приклад кільцевої мережі. Кожен комп'ютер підключається до двох сусідніх

комп'ютерів, в результаті чого сигнал йде «по колу». У цьому випадку термінатори не потрібні, оскільки немає ізольованого кінця мережі.

У кільцевій мережі також використовується коаксіальний кабель. Екранована вита пара (STP) використовується в спеціальній розгалуженій і кільцевій мережі (Token Ring, яка представляє собою логічне кільце відповідно до специфікації IEEE 802.5).

У кільцевій мережі передача сигналу відбувається в одному напрямку. Кожен комп’ютер отримує сигнал від свого сусіда ліворуч і посилає його сусідові справа. Такий тип топології називається активною, тому що під час передачі відбувається додаткове посилення сигналу.

Найчастіше кільцева топологія практично реалізується у вигляді архітектури Token Ring. У цьому випадку використовується концентратор Token Ring, також званий MSAU (Multistation Access Unit, Multistation Access Unit).

Переваги кільцевих топологічних мереж:

• простота фізичного виконання;

• легкість усунення різного роду проблем.

• Деякі недоліки, пов'язані з використанням круглих сіток:

• низький рівень надійності (при розриві кільця виходить з ладу вся мережа);

• труднощі з додаванням нових комп'ютерів.

Мережі з топологією зірки

Загальновідомо, що зірка є однією з найбільш часто використовуваних топологій у процесі побудови локальних мереж. У процесі створення такого типу мережі кожен комп'ютер підключається до центрального концентратора.

Використовуваний у цьому випадку концентратор може бути активним, пасивним або інтелектуальним. Пасивний концентратор використовується для реалізації фізичного підключення без споживання енергії. Найбільш поширеним є активний концентратор, який фактично є багатопортовим повторювачем. Цей тип концентраторів посилює сигнали, що передаються. Якщо активний хаб оснащений діагностичним пристроєм, він називається розумним хабом.

Неекранована кручена пара (Ethernet, lobaset або 100baset архітектура) використовується в процесі побудови мережі з топологією «зірка»[1].

У звичайній зіркоподібній мережі сигнал надходить від мережевих карт, встановлених у комп’ютерах, до концентраторів. Потім сигнал посилюється, а потім повертається до мережевих карток.

Переваги зіркоподібної топології:

• підвищена стабільність мережі;

• простота додавання/видалення нового комп'ютера в мережі;

• легко діагностувати та вирішувати проблеми.

Недоліки зіркоподібної топології:

• підвищений знос кабелю при прокладанні мережі;

• необхідність придбання дорогого концентратора.

1.1.2 **Стандартизація в комп'ютерних мережах**. Створення, розвиток і розширення використання комп'ютерних мереж були б неможливі без застосування однакових принципів передачі та обробки даних - стандартів.

Основними організаціями, які займаються розробкою та підтримкою впровадження стандартів у сфері комп'ютерних мереж, є:

1. Міжнародний союз електрозв'язку (ITU - International Telecommunication Union). Стандарти МТС розділені на серії. Норми кожної

серії, присвяченої одній темі, позначаються великою літерою латинського алфавіту. Після літери ставиться крапка і номер стандарту. Наприклад, літерою V позначені стандарти передачі даних по телефонних каналах, літерою X - стандарти мереж передачі даних, буквою Q - стандарти телефонної комутації і сигналізації.

2. Технічний комітет 97 - комітет Міжнародної організації зі стандартизації (ISO - International Standard Organisation). Технічний комітет розробляє стандарти обробки інформації за допомогою ЕОМ. Стандарти цієї організації ідентифікуються чотиризначним номером і суфіксом ISO. Наприклад, стандартом для стека протоколів TCP/IR є ISO 7498.

3. Рада з питань діяльності в Інтернеті (IAB - Internet Activities Board) - розробляє стандарти діяльності в Інтернеті.

Дотримання стандартів, розроблених цими організаціями, є обов'язковим при проектуванні, створенні та експлуатації будь-яких комп'ютерних мереж.

1.1.3 **Архітектурні принципи побудови комп'ютерних мереж**. Розглядаючи створення та функціонування комп’ютерних мереж, пам’ятайте про такі визначення та поняття:

• реальна система – набір з одного або кількох комп’ютерів, програмного забезпечення, периферійних пристроїв, терміналів і персоналу, який є повністю автономним і приймає і передає дані;

• реальна кінцева система (real end system) – реальна система, що виконує функції станцій даних у мережі, тобто є джерелом або одержувачем даних;

• відкрита система (відкрита система) – система, побудована та функціонуюча відповідно до вимог міжнародних стандартів;

• комунікаційна система – реальна відкрита система, що забезпечує обмін даними між системами, що беруть участь у відкритій інформаційній системі;

• система учасника (система користувача) – реальна відкрита система, яка є постачальником або споживачем мережевих ресурсів, забезпечує доступ до них користувачам і керує взаємозв'язком відкритих систем;

• прикладний процес (application process) – процес у реальній кінцевій системі, який обробляє дані для конкретних потреб користувача;

• середовище передачі – набір ліній для передачі даних і, можливо, інших пристроїв, що забезпечують передачу даних між абонентськими системами;

• середовище обміну відкритими системами – набір функцій, які дозволяють реальним відкритим системам обмінюватися даними відповідно до міжнародних стандартів.

Зв'язок деяких основних понять у відкритій інформаційній системі показано на схемі на рисунку 1.2.

Рисунок 1.2 – Відкрита інформаційна система

Структура комунікаційного середовища відкритих систем визначається стандартом ISO 7498. Середовище в цілому має складний набір функцій. При його створенні використовується ієрархічний підхід, який базується на таких принципах:

• оскільки функція передачі в середовищі дуже складна, її поділяють на рівні;

• на кожному рівні виконується певний кінцевий набір завдань;

• на межі рівня обмін даними повинен бути мінімальним;

• рівні повинні бути описані таким чином, щоб зміни одного з них не призводили до необхідності внесення змін до інших рівнів [1].

1.1.4 **Семирівнева модель взаємодії відкритих систем**. Відповідно до стандарту ISO 7498 обробка даних під час їх передачі в сеансі зв’язку відкритих систем поділяється на сім рівнів, які перераховані нижче:

• рівень 7 - використаний;

• рівень 6 - рефлексія;

• рівень 5 - сесія;

• рівень 4 - транспорт;

• рівень 3 - мережевий;

• рівень 2 - канал;

• рівень 1 - фізичний.

Передача даних за допомогою семирівневої моделі взаємозв’язку відкритої системи (OSI) здійснюється, як показано на малюнку 1.3.

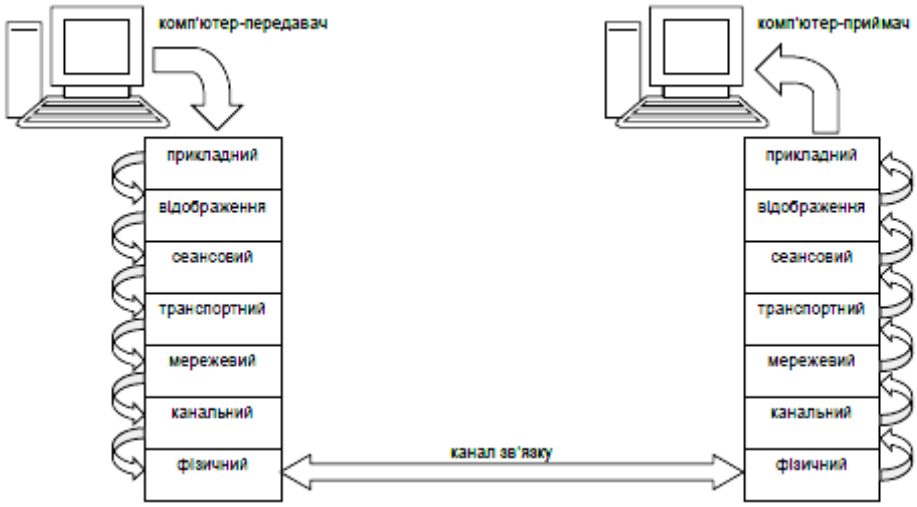

Рисунок 1.3 – Передавання даних з використанням семирівневої моделі взаємодії відкритих систем

Перед початком процесу передачі файл даних або потік даних розбивається на кадри розміром, наприклад, 64 КБ. Кожен кадр передається окремо (рис. 7).До недавнього часу в практиці комп'ютерної комунікації широко використовувалася модель взаємодії відкритих систем Міністерства оборони США DoD (Department of Defense). На рисунку 1.4 зображено схему процесу обробки даних за цією моделлю [2].

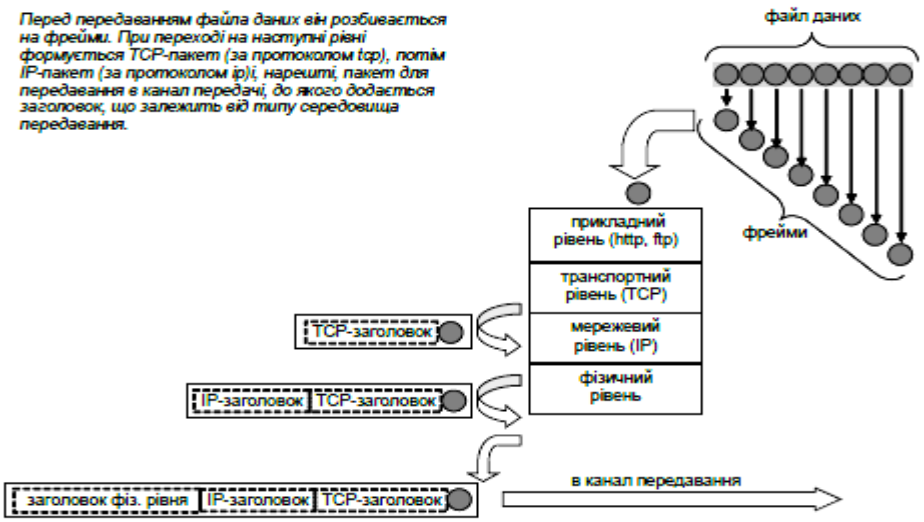

Рисунок 1.4 – Обробка фреймів пакету на різних рівнях чотирирівневої моделі зв’язку відкритих систем DoD

1.2 Захист інформації в локальній мережі

1.2.1 Законодавча база України. Існує Концепція технічного захисту інформації в Україні, затверджена постановою Кабінету Міністрів України від 8 жовтня 1997 р. N 1126.

Ця Концепція визначає основи державної політики у сфері захисту інформації інженерно–технічними заходами. Технічний захист інформації (далі – ТЗІ) є складовою частиною забезпечення національної безпеки України.

Концепція має забезпечити єдність принципів формування і проведення такої політики в усіх сферах життєдіяльності особи, суспільства та держави (соціальній, політичній, економічній, військовій, екологічній, науково-технологічній, інформаційній тощо) і служити підставою для створення програм розвитку сфери ТЗІ.

ТЗІ – це діяльність, спрямована на забезпечення інженерно-технічними заходами порядку доступу, цілісності та доступності (унеможливлення блокування) інформації з обмеженим доступом, а також цілісності та доступності відкритої інформації, важливої для особи, суспільства і держави.

Напрями розвитку ТЗІ обумовлюються необхідністю своєчасного вжиття заходів, адекватних масштабам загроз для інформації, і ґрунтуються на засадах правової демократичної держави відповідно до прав суб'єктів інформаційних відносин на доступ до інформації та її захист [3].

Приведення інформаційних відносин у сфері ТЗІ у відповідність з міжнародними стандартами сприятиме утвердженню України у світі як демократичної правової держави .

1992р. Держтехкомісія (ДТК) при Президенті Російської федерації опублікувала п'ять Керівних документів щодо захисту від несанкціонованого доступу до інформації. Ці документи чинні і в Україні. Ідейною основою цих документів є «Концепція захисту засобів обчислювальної техніки від

несанкціонованого доступу до інформації (НСД)». Документи ДТК встановлюють дев’ять класів захищеності автоматизованих систем (АС) від НСД (3Б, 3А, 2Б, 2А, 1Д, 1Г, 1В, 1Б, 1А), кожний з яких характеризується певною сукупністю вимог до засобів захисту.

В Україні в 1999р. Було вперше введено в дію вітчизняний пакет із п’яти нормативних документів НД ТЗІ з питань технічного захисту інформації комп’ютерних систем від несанкціонованого доступу.

Нижче наводиться список НПА щодо інформаційної безпеки (в значенні захисту інформації) в Україні.

2. Закони України:

- Закон України «Про інформацію» від 02.10.1992 № 2657;
- Закон України «Про захист інформації в інформаційно-телекомунікаційних системах» від 05.07.1994 № 80/94–ВР;
- Закон України «Про державну таємницю» від 21.01.1994 № 3855–XII;
- Закон України «Про захист персональних даних» від 01.06.2010 № 2297–VI.

2. Постанови КМУ:

- Постанова Кабінету міністрів України «Про затвердження Правил забезпечення захисту інформації в інформаційних, телекомунікаційних та інформаційно–телекомунікаційних системах» від 29.03.2006 №373;
- Постанова Кабінету міністрів України «Про затвердження Інструкції про порядок обліку, зберігання і використання документів, справ, видань та інших матеріальних носіїв інформації, які містять конфіденційну інформацію, що є власністю держави» від 27 листопада 1998 р. №1893[5].

1.2.2 **Міжнародні стандарти захисту інформації**. Серія стандартів ISO/IEC 27000 Серія стандартів управління інформаційною безпекою ISO/IEC 27000 була розроблена підкомітетом SC 27 Технічного комітету ISO/IEC JTC 1.

Система управління інформаційною безпекою (СУІБ) містить вимоги до впровадження та вдосконалення систем управління інформаційною безпекою та базується на моделі PDCA (Plan-Do-Check-Act):

• створення - ідентифікація активів, управління ризиками;

• реалізація - фаза реалізації відповідних заходів управління безпекою;

• верифікація - моніторинг та аналіз;

• експлуатація - підтримання в працездатному стані та поліпшення.

Виходячи з цього, можна побачити, що крім розробки правил управління та безпеки, не менш важливо забезпечити циклічність усіх процесів управління безпекою, щоб усі процедури поступово проходили через фази моделі PDCA. Це доводить, що система управління відповідає стандарту ISO 27001 і готова до сертифікації SMIB.

Виконання вимог стандарту ISO/IEC 27001 дозволяє, перш за все, мінімізувати ризик втрати майна компанії/організації, а отже, зменшити фінансові втрати.

Стандарт ISO/IEC 27001 призначений для сертифікації систем захисту інформації.

Сертифікація системи управління інформаційною безпекою (сертифікація SMIB) означає ефективне управління бізнес-процесами компанії/організації, інформаційними ризиками, а також сертифікат стійкої, що розвивається та надійної компанії, що в свою чергу дає позитивний підхід до бізнесу. підсилювач.

ISO/IEC 27001 Стандарт ISO/IEC 27001 є частиною загальної системи менеджменту компанії. Стандарт ISO/IEC 27001 (ISO 27001) містить описи найкращих світових практик у сфері управління інформаційною безпекою. ISO 27001 визначає вимоги до системи управління інформаційною безпекою,

щоб продемонструвати здатність організації захищати свої інформаційні активи. Цей стандарт було підготовлено як модель для розробки, впровадження, експлуатації, моніторингу, аналізу, обслуговування та вдосконалення EMS.

Сімейство стандартів ISO 27000 включає такі документи:

1. ISO/IEC 27001:2013 Системи управління інформаційною безпекою. Вимоги - Система управління інформаційною безпекою. Вимоги

• Інформаційна безпека - збереження конфіденційності, цілісності та доступності інформації; крім того, можуть бути включені інші властивості, такі як автентичність, незаперечність, автентичність.

• Конфіденційність – забезпечення доступності інформації лише особам, які мають відповідні дозволи (авторизованим користувачам).

• Цілісність – забезпечення точності та повноти інформації та способів її обробки.

• Доступність – надання доступу до інформації авторизованим користувачам у разі потреби (на вимогу).

Саме поняття «захист інформації» міжнародний стандарт трактує як забезпечення конфіденційності, цілісності та доступності інформації. Основою стандарту ISO 27001 є система управління інформаційними ризиками. Система управління ризиками дає відповіді на такі питання:

• на яку сферу інформаційної безпеки слід звернути увагу;

• скільки часу і грошей можна витратити на це рішення технічного захисту інформації.

2. ISO/IEC 27000:2014 Системи управління інформаційною безпекою. Огляд і словник - Система управління інформаційною безпекою. Огляд і термінологія.

3. ISO/IEC 27002:2013 Кодекс управління інформаційною безпекою – Принципи управління інформаційною безпекою [4].

Стандарт містить вказівки щодо передового досвіду управління інформаційною безпекою для тих, хто відповідає за розробку, впровадження або підтримку систем управління інформаційною безпекою. Стандарт визначає інформаційну безпеку як «збереження конфіденційності (забезпечення того, що інформація доступна лише особам, уповноваженим це робити), цілісності (забезпечення точності та повноти інформації та способів її обробки) та доступності (забезпечення того, що авторизовані користувачі мають доступ до інформації). та пов’язані ресурси).

4. ISO/IEC 27003:2010 Guidelines for implementing of information security management systems - Настанови щодо впровадження систем менеджменту інформаційної безпеки.

Він описує специфікацію та процес проекту СУІБ від початку до планів реалізації проекту, включаючи підготовчу та планову діяльність, до фактичного впровадження, і включає ключові елементи, такі як:

1. Схвалення керівництва та остаточне рішення про початок проекту;

2. Покриття та демаркація з точки зору ІКТ та фізичного розташування;

3. Оцінка загроз інформаційній безпеці та планування відповідної обробки загроз, якщо необхідно визначити вимоги до управління інформаційною безпекою;

4-й проект SMIB;

5. Планування проекту впровадження.

6. ISO/IEC 27005:2011 Управління ризиками інформаційної безпеки - Управління ризиками інформаційної безпеки.

7. ISO/IEC 27006:2011 Вимоги до суб’єктів аудиту та сертифікації систем управління інформаційною безпекою – Вимоги до суб’єктів аудиту та сертифікації систем менеджменту інформаційної безпеки.

8. ISO/IEC 27007:2011 Настанова щодо аудиту систем управління інформаційною безпекою (FCD).

9. ISO/IEC 27008:2011 Керівні принципи для аудиторів щодо засобів контролю СУІБ (ПРОЕКТ) - Керівні принципи аудиту засобів контролю СУІБ.

10. ISO/IEC 27011:2008 Настанови щодо управління інформаційною безпекою для телекомунікаційних організацій на основі ISO/IEC 27002 - Настанови щодо управління інформаційною безпекою в телекомунікаціях на основі ISO/IEC 27002.

11. ISO/IEC 27799:2008 Управління інформаційною безпекою в охороні здоров’я з використанням ISO/IEC 27002 – Настанови щодо управління інформаційною безпекою для організацій охорони здоров’я на основі ISO/IEC 27002 [3].

1.2.3 **Класифікація загроз і атак на IT-систему.** IT-система – це сукупність організаційних і технічних засобів, що використовуються для зберігання та обробки інформації з метою задоволення інформаційних потреб користувачів. Таке визначення може бути задовільним лише з найбільш загальної та неофіційної точки зору та потребує подальшого уточнення. IT-системи працюють в Україні під назвою «Автоматичні системи (АС)». Існує два типи загроз IT-системам – пасивні та активні.

Пасивні загрози полягають у перегляді та/або реєстрації даних, що передаються по лініях зв'язку. Пасивні загрози полягають у порушенні конфіденційності даних, що циркулюють у мережі)

Активні загрози — це несанкціоноване використання мережевих пристроїв для зміни окремих повідомлень або потоків повідомлень. Активні загрози небезпечні, оскільки вони порушують цілісність або доступність мережевих ресурсів і компонентів.

У найширшому розумінні загрозою для інформаційних ресурсів можна вважати потенційно можливі випадки природного, технічного або антропогенного характеру, які можуть спричинити небажаний вплив на ІТ-систему, а також на інформацію, що в ній зберігається. Виникнення загрози, тобто знаходження джерела оновлення конкретних подій у загрозі, характеризує елемент як вразливість. Саме наявність уразливості як певної особливості системи є причиною активації загроз. Загрози теорії множин самі по собі не є вичерпними і тому не можуть бути повністю описані.

Вважаємо, що шляхом інтеграції різних підходів, а також пропозиції вирішення цієї проблеми можна виділити такі типи загроз інформаційній безпеці: розкриття джерел інформації; порушення їх цілісності; вихід з ладу самого пристрою.

Види загроз в інформаційній безпеці. Загроза розкриття джерел інформації полягає в тому, що дані, інформація та знання стануть відомі тим, хто не повинен їх знати. В рамках нашої роботи під загрозою розголошення ми будемо розуміти ситуацію, коли є несанкціонований доступ до системних ресурсів, і мова йде про відкриті ресурси та ті, до яких доступ обмежений. Ці ресурси мають передаватися один одному та зберігатися в одній ІТ-системі [4].

2. Загроза порушення цілісності інформаційних ресурсів полягає в умисному антропогенному впливі (модифікації, видаленні, скороченні) даних, що зберігаються в ІТ-системі суб’єкта господарювання, а також передаються з цієї інформаційної системи іншим.

3. Ризик виходу з ладу самого пристрою може виникнути у разі блокування доступу до одного або кількох ресурсів IT-системи. Насправді блокування може бути постійним, так що запитуваний ресурс ніколи не буде отримано, або це може спричинити затримку в отриманні запитуваного ресурсу, достатню для того, щоб зробити його непридатним для використання.

Відповідно до вищесказаного розглянемо наступні загрози інформаційній безпеці. Врахування цих загроз має на меті продемонструвати, що знання загроз і вразливостей дозволить організувати відповідну систему управління інформаційною безпекою.

Найбільш поширеними і небезпечними є ненавмисні помилки користувачів, операторів, системних адміністраторів та інших осіб, які керують IT-системами. Іноді такі помилки несуть загрозу (некоректно введені дані, помилка в програмі, що спричиняє системний збій), іноді вони створюють ситуації, якими не тільки можуть скористатися зловмисники, але й самі несуть пряму загрозу об’єкту. Загалом, згідно з дослідженнями експертів з інформаційної безпеки, понад 65% шкоди, завданої інформаційним активам, є результатом ненавмисних помилок. Набагато рідше трапляються пожежі та землетруси, тобто стихійні лиха. Тому слід зосередитися на більш широкому впровадженні IT-систем, які забезпечують безпеку.

Крім того, за розміром збитку можна виділити крадіжку та шахрайство, суб’єктами яких у більшості випадків були штатні працівники цих організацій, які добре володіють IT-системою, а також забезпеченість коштів.

У цьому аспекті дуже небезпечні співробітники, які незадоволені або не поділяють цінності організації, в якій вони працюють. Показовим прикладом є дії колишнього генерала СБУ, одного з керівників ГУР України Валерія Кравченка, який 18 лютого 2004 року, маючи обмежений доступ до своїх матеріалів, незаконно надав доступ іншим особам, зокрема Deutsche Журналісти Welle.

Загалом дії постраждалих співробітників мотивовані спробою завдати шкоди організації, в якій вони працювали, яка, на їхню думку, їх образила. Таке правопорушення може проявлятися в таких діях:

• пошкодження приладу;

• вставлення логічної бомби, яка з часом знищує програми та дані;

• введення невірних даних;

• знищення даних;

• зміна даних;

• модифікація даних;

• надання доступу до службових даних тощо.

Окрім антропогенних загроз, слід виділити й природні загрози. Природні небезпеки характеризуються широким спектром. В першу чергу можна вказати порушення інфраструктури: перебої з електроенергією, тимчасова відсутність зв’язку, перебої з водопостачанням і т. д. Також небезпечними є стихійні лиха, землетруси, урагани, торнадо, шторми, тайфуни і т. д. Природні становлять близько 14 відсотків. всього.

Класифікація небезпеки

Розглянемо класифікацію загроз докладніше. Отже, предметом розкрадання є:

• апаратне забезпечення (блоки, вузли та готові продукти), яким оснащені комп'ютери та мережі;

• програмне забезпечення та носії інформації;

• паперові копії з друкованою інформацією.

Крадіжка може бути організована:

• робочі місця користувачів;

• під час транспортування;

• зі складських приміщень.

Джерелами помилок в програмному забезпеченні (ПО) можуть бути:

• логічні помилки розробників програмного забезпечення;

• непередбачені ситуації, що виникають під час оновлення, заміни або додавання нового обладнання, встановлення нових додатків, доступу до нових режимів роботи програмного забезпечення, поява раніше не зафіксованих нештатних ситуацій;

• віруси, що заражають програми;

• спеціальні програмні компоненти, що надаються розробниками програмного забезпечення для різних цілей.

Самі віруси також небезпечні і можуть виводити повідомлення на екран монітора; стирання інформації на дисках; переміщення файлів в інші папки; уповільнення комп'ютера; збір інформації про роботу організації тощо.

Враховуючи сферу діяльності органів державного управління, на нашу думку, загрозу їхнім ІТ-системам можливо з метою:

• забезпечення доступу до інформації з обмеженим доступом;

• викрадення ключів, паролів, ідентифікаторів, списків користувачів;

• виключення частини або всієї системи органів державного управління.

Типи загроз також різноманітні відповідно. Завдяки їх кількості ми спробували виділити загрози інформаційній безпеці з урахуванням сучасного розвитку класифікації загроз національній безпеці.

За джерелом походження:

• природного походження - включають небезпечні геологічні, метеорологічні, гідрологічні морські та прісноводні явища, деградацію ґрунту чи надр, природні пожежі, масове загибель сільськогосподарських рослин і тварин хворобами чи шкідниками, зміни стану водних ресурсів і біосфери тощо:

• антропогенного походження - дорожньо-транспортні пригоди (катастрофи), пожежі, неспровоковані вибухи або їх загроза, раптове руйнування каналів зв'язку, збої в роботі інженерних мереж і пристроїв життєзабезпечення, збої в роботі основних серверів органів державного управління тощо;

• антропогенного походження – різноманітна діяльність людини, спрямована на руйнування ІТ-систем, ресурсів, програмного обладнання об’єкта тощо. До цієї групи дій відносяться: ненавмисні, викликані неправильними чи ненавмисними діями особи (наприклад, це може бути помилкова запуск програми, ненавмисне встановлення закладок тощо); навмисні (інспіровані), що є наслідком навмисних дій людей (наприклад, навмисне встановлення програм, які передають інформацію на інші комп’ютери, навмисне впровадження вірусів тощо).

• Залежно від ступеня гіпотетичного пошкодження:

• загрози – це очевидні або потенційні дії, які перешкоджають або перешкоджають реалізації національних інтересів в інформаційній сфері та становлять загрозу для системи державного управління та управління її системоутворюючими елементами;

• загроза – негайна дестабілізація функціонування системи державного управління.

За ознакою повторності злочину:

• повторювані - загрози, які вже мали місце;

• безперервне – повторне розгортання загроз, що складається з серії однакових загроз, які мають спільну мету.

За регіоном походження:

• екзогенний – джерело дестабілізації системи знаходиться поза її межами;

• ендогенний – алгоритм дестабілізації системи знаходиться в самій системі.

За ймовірністю виконання:

• вірогідні – ті загрози, які обов’язково виникнуть при виконанні певного набору умов. Прикладом може бути повідомлення про атаку на інформаційні ресурси суб’єкта національної безпеки, яке передує самій атаці;

• неможливо – загрози, які ніколи не виникнуть, якщо буде виконано певний набір умов. Такі погрози, як правило, мають декларативний характер, не підкріплені реальною чи навіть потенційною можливістю реалізації задекларованих намірів і, як правило, мають залякувальний характер;

• випадкові – загрози, які виглядають по-різному кожного разу, коли виконується певний набір умов. Загрози такого рівня слід аналізувати з використанням оперативних методів дослідження, особливо теорії ймовірностей та теорії ігор, які вивчають закономірності випадкових явищ.

За рівнем детермінованості:

• регулярні – ті загрози, які є постійними, повторюваними, зумовленими об’єктивними умовами існування та розвитку системи захисту інформації. Наприклад, кожен суб'єкт NS буде піддаватися інформаційним атакам, якщо його система захисту інформації не працює або працює не на належному рівні;

• випадкові - загрози, які можуть відбутися або не відбутися. До таких загроз також відноситься загроза з боку хакерів, що дестабілізують ІТ-системи суб’єктів НС, НС СР.

За значенням:

• прийнятні – такі загрози, які не можуть призвести до збою системи. Приклади включають віруси, які не руйнують програми шляхом їх знищення;

• неприйнятні – такі загрози, які: 1) у разі їх реалізації можуть призвести до збою та системної дестабілізації системи; 2) може призвести до змін, несумісних із подальшим існуванням Служби державної охорони. Наприклад, вірус «I love you» пошкодив комп’ютерні системи в багатьох містах світу та завдав загальних збитків приблизно на 100 мільйонів доларів США [4].

1.2.4 **Методи захисту**. Немає повного захисту від усіх зловмисних програм та їх проявів, але щоб зменшити ризик втрати через шкідливі програми, дотримуйтесь цих вказівок:

• використовувати сучасні операційні системи з високим рівнем захисту від шкідливих програм;

• використовувати лише ліцензійне програмне забезпечення (операційну систему та додатки);

• працювати на персональному комп’ютері лише з правами користувача, а не адміністратора, що унеможливить установку на персональний комп’ютер більшості шкідливих програм;

• використовувати спеціалізоване програмне забезпечення, яке включає евристичні (поведінкові) аналізатори;

• використовувати антивірусні програми відомих виробників з автоматичним оновленням баз сигнатур;

• використовувати персональний мережевий екран, який контролює доступ до Інтернету з персонального комп’ютера на основі правил, встановлених користувачем;

• постійно оновлювати програмне забезпечення.

1.2.5 **Криптографія**. Захист даних за допомогою шифрування є одним із можливих рішень проблеми їх безпеки. Зашифровані дані будуть доступні

лише тим, хто знає, як їх розшифрувати, тому крадіжка зашифрованих даних абсолютно марна для неавторизованих користувачів.

Коди та шифри використовувалися задовго до появи комп’ютерів. З теоретичної точки зору чіткої різниці між кодами та шифрами немає. Однак у сучасній практиці різниця між ними зазвичай досить чітка. Коди працюють над мовними елементами та розкладають зашифрований текст на такі семантичні елементи, як слова та склади. У шифрі завжди є дві складові: алгоритм і ключ.

Алгоритм дозволяє використовувати відносно короткий ключ для шифрування будь-якого великого тексту. Шифри в основному використовуються для захисту даних в ІБ, тому ми поговоримо про них пізніше. У цьому розділі будуть представлені основні концепції криптографічного захисту інформації, корисні на практиці, а також переваги та недоліки найпоширеніших заходів захисту.

Давайте визначимо деякі терміни, які використовуються в криптографії.

Криптологія - це наука, що складається з двох областей: криптографії та криптоаналізу.

Криптографія — це наука про методи перетворення (шифрування) інформації для її захисту від зловмисників. Криптографія займається методами перетворення інформації, які повинні запобігти зловмиснику отримати її з перехоплених повідомлень. При цьому по каналу зв'язку передається не сама захищена інформація, а результат її перетворення за допомогою шифру або коду, і перед зловмисником стоїть складне завдання розшифрувати шифр або код.

Злом шифру — це процес вилучення захищеної інформації (відкритого тексту) із зашифрованого повідомлення (зашифрованого тексту) без знання використовуваного шифру.

Шифрування — це процес використання шифру та захищеної інформації, тобто перетворення захищеної інформації в зашифроване повідомлення за допомогою певних правил, які містяться в шифрі.

Дешифрування — процес, зворотний до шифрування, який полягає в перетворенні зашифрованого повідомлення в інформацію, захищену за допомогою певних правил, що містяться в шифрі.

Ключ у криптографії - це змінний елемент шифрування, який використовується для шифрування певних повідомлень.

Одне з центральних місць у понятійному апараті криптографії займає поняття **стійкості шифру**. Під міцністю шифру розуміють здатність шифру протистояти всім методам розтину. Якісно це досить легко зрозуміти, але отримання точних оцінок міцності будь-якого конкретного шифру все ще залишається відкритою проблемою. Це пояснюється тим, що досі немає математичних результатів, необхідних для вирішення такої задачі. Таким чином, про стабільність конкретного шифру можна судити лише за різними спробами зламати його та залежить від кваліфікації криптоаналітиків, які зламують шифр. Ця процедура називається перевіркою на криптостійкість.

Криптоаналіз - це дослідження (і практика його застосування) методів і засобів розшифровки шифрів. Зв'язок між криптографією і криптоаналізом очевидна: криптографія - це захист, тобто розробка шифрів, а криптоаналіз - це аналіз шифрів. Однак ці дві науки споріднені, тому що стійкість розробленого шифру можна довести різними спробами зламати шифр, поставивши себе на місце зловмисника[3].

Висновки до першого розділу

В ході проведеного аналізу теоретичних та методологічних основ створення систем кіберзахисту на основі стандартів ЄС. Розглянуто принципи побудови систем відеонагляду та сигналізації і попередження несанкціонованого доступу та витоку інформації з метою їх подальшої реалізації. Були виявлені такі проблеми, як необхідність суворого контролю та управління серверами. Названі проблеми посилюються ще тим фактором, що розміщуючи інформацію на сервері, користувачі не завжди можуть контролювати рівень безпеки. Також розглянуті можливі атаки на ресурси системи та виявлені елементи, котрі найбільш вразливі. До них можна віднести канал зв’язку, Web-сервер, комп’ютери користувачів, сервери баз даних, корпоративні сервери.

Структурно інтегрована система кіберзахисту може включати в себе спільно функціонуючі системи відеоспостереження, системи сигналізації, контролю та управлінням доступом, охоронну та пожежну систем, а також ряд додаткових систем, що забезпечують захист від різного рівня загроз.

Для досягнення необхідного рівня системи кіберзахисту необхідно забезпечити протидію різноманітним технічним загрозам та мінімізувати можливий вплив «людського фактору». В даному випадку доцільно передбачити можливість подальшого розвитку системи, шляхом розширення та вдосконалення окремих елементів її частин, а також додавання до існуючих систем у вигляді підсистеми.

Таким чином можна зробити висновок про досягнення мети і вирішення завдань, теоретичної та методологічної побудови системи кіберзахисту та приступити до проектування.

РОЗДІЛ 2
ПРОЕКТУВАННЯ ЛОКАЛЬНОЇ МЕРЕЖІ

2.1 Проектування локальної мережі виробничого підприємства

2.1.1 **Опис об'єкта та план будівництва**. Приватна виробничо-торговельна компанія «Молтехпром» заснована в травні 1996 року відповідно до Господарського і Цивільного кодексу та інших нормативних актів України. ПП «Молтехпром» знаходиться за адресою - м. Полтава, вулиця Поштова, 1Б. Спеціалізується на розробці та виробництві автомобільних запчастин і запчастин на спеціальні замовлення. У будівлі розташовані 5 будівельних відділів компанії, кабінет директора відділу та серверна. Тоді структура компанії чітко визначена:

• Відділення № 1 – «Комп’ютерне моделювання»

• відділення № 2 – «Торгівля»;

• відділення № 3 – «Бухгалтерський облік»;

• відділення № 4 - «ІТ відділ»;

• Відділення № 5 - «Відділ кадрів»;

Відділ 1 - Комп'ютерне моделювання. Виконання розрахунків, створення креслень і моделей Має 1 кімнату та 10 робочих місць.

Відділ 2 - Продажі. Підвищення вартості та прибутковості укладених договорів, збільшення кількості великих замовлень та контрактів, робота з клієнтами, пошук потенційних рекламодавців. Має 1 кімнату та 8 робочих місць.

Відділ 3 - Бухгалтерія. Керівництво роботою організації в галузі обліку та оплати праці працівників, організації планово-економічних робіт. Має 1 кімнату та 5 робочих місць.

Відділ 4 - відділ ІТ. Обслуговування комп'ютерів та іншого електронного обладнання, що належить компанії. Має 1 кімнату та 4 робочі місця.

Відділ 5 - відділ кадрів. Визначення кадрової потреби підприємства та набір працівників, аналіз плинності кадрів, пошук методів боротьби з плинністю кадрів, складання розкладу зайнятості на підприємстві, формування особових справ працівників. Має 1 кімнату та 3 робочі місця. Поруч знаходиться також кабінет керівника підприємства, в якому є 1 комп.

Під час доопрацювання дипломної роботи буде розроблено локальну мережу для одноповерхової будівлі Молтехпрому. Загальна площа 230 м2. Вся будівля розділена на 7 кімнат:

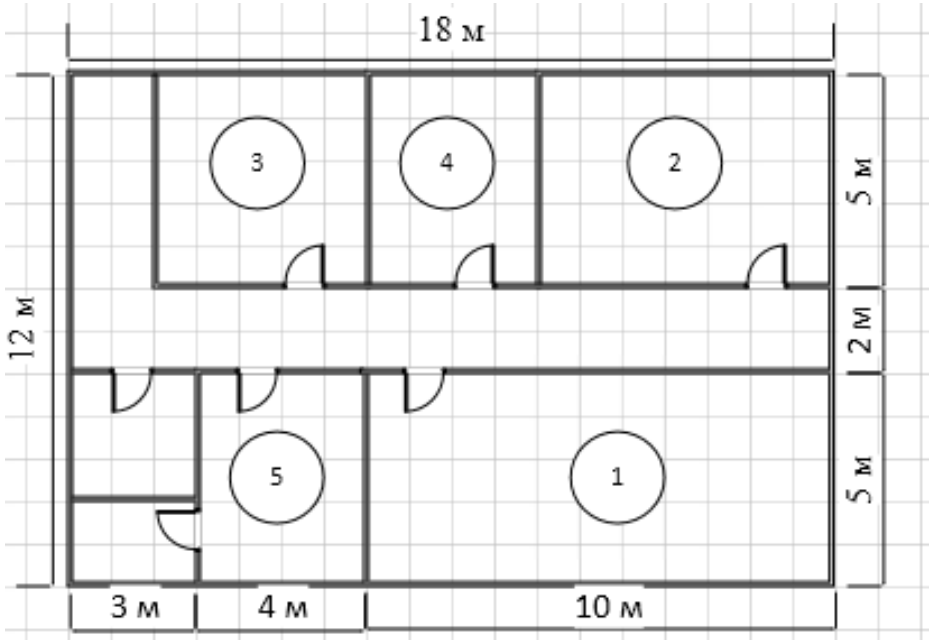

Рисунок 2.1 – План будівлі

Для функціонування закладу необхідно забезпечити швидкий та якісний доступ до необхідної інформації. Через кількість робочих станцій і необхідність швидкого обміну великими обсягами даних локальна мережа повинна забезпечувати швидкість 1 Гбіт/с. Доступ до глобальної мережі Інтернет також матиме швидкість 1 Гбіт/с, щоб усі необхідні комп’ютери були підключені до мережі одночасно.

Комп’ютери розподіляються з розрахунку 4,5 кв. м на 1 робоче місце відповідно до ДНАОП 0.00-1.31-99 (НПАОП 0.00-1.31-99) «Основи охорони праці при експлуатації електронно-обчислювальних засобів» та ДСанПІН 3.3.2.007-98 «Державні правила і стандарти гігієни праці з наочні посібники, термінали для відображення електронно-обчислювальних машин. Так, відділ комп’ютерного моделювання має 10 комп’ютерів, відділ продажу – 8 комп’ютерів, бухгалтерія – 4 комп’ютери, відділ інформаційних технологій – 5 комп’ютерів, відділ кадрів – 3 комп’ютери, один комп’ютер у кабінеті керівника відділу. Загальна кількість приміщень – 7, загальна кількість робочих місць – 31. Для підключення всіх комп’ютерів до мережі будуть використовуватися 4 мережеві комутатори та один маршрутизатор, мережа також використовуватиметься для доступу до файлового сервера та Інтернету. для доступу співробітників до мережі буде реалізована можливість бездротового підключення.

2.1.2 **Побудова функціональної схеми**. Функціональна схема - це документ, що пояснює процеси, що відбуваються в окремих функціональних вузлах продукту або в продукті в цілому. Функціональна схема дозволяє зрозуміти всю логіку роботи пристрою, всі відмінності від інших подібних пристроїв. За такою схемою можна визначити, як здійснюються перетворення і які функціональні елементи для цього потрібні. Кожен функціональний елемент містить лише ті входи та виходи, які необхідні для його належної роботи. На ієрархічній функціональній схемі вказується кількість робочих станцій, мережевого обладнання (комутаторів) і технологія комутаційних елементів. У цій локальній мережі ми будемо використовувати найбільш використовувану на даний момент технологію Ethernet.

Через велику кількість робочих станцій, необхідність постійного доступу до мережі Інтернет, а також зростаючі потреби співробітників і студентів у швидкості передачі, неможливо використовувати комутаційні пристрої 10 Мбіт/с. Використання швидкості 100 Мбіт є більш раціональним,

але не гарантує високу швидкість з'єднання для всіх робочих станцій і підключених мобільних пристроїв. Тому в цьому випадку ми будемо використовувати елементи комутації 1000 Мбіт/с.

Після представлення ієрархічної схеми необхідно скласти графічне зображення розташування робочих місць щодо проекції будівлі. Система зв’язку показана на рисунку 2.3.

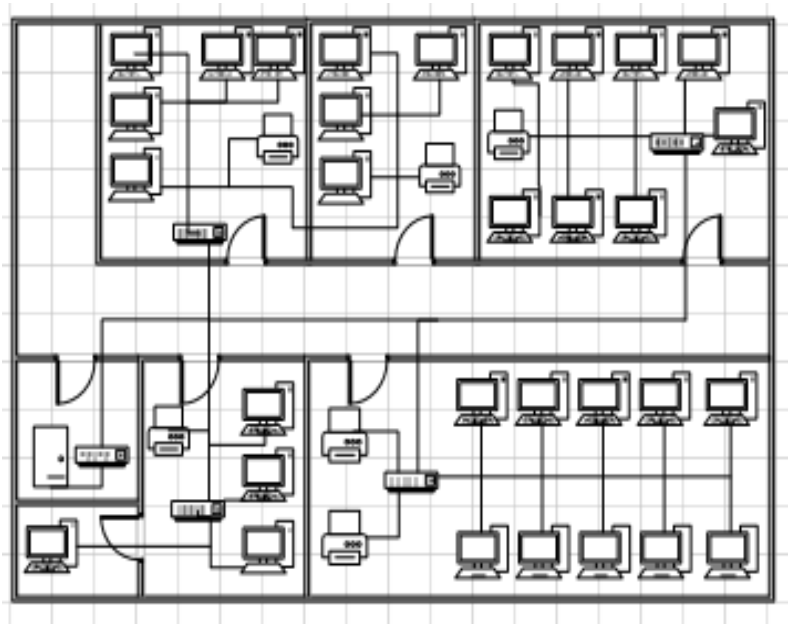

Рисунок 2.2 – Функціональна схема розташування комп’ютерів

2.1.3 Вибір і загальна характеристика приладів. Основним завданням дипломної роботи є проектування локальної мережі та забезпечення її функціонування. Мережа буде побудована з використанням 2 точок доступу Wi-Fi, 1 роутера, 4 комутаторів та FTP-кабелю категорії 5 – 100,3 м. В наявності 31 робоча станція та файловий та WWW сервер. Враховуючи, що обов'язки співробітників компанії умовно поділяються на: вимогливу продуктивність роботи з персональним комп'ютером (робота з графікою, робота з кресленнями, дизайн) і невимогливу продуктивність роботи з персональним комп'ютером (робота з текстом, робота в Інтернеті) , прості розрахункові операції) будуть правильно підібрані персональні комп’ютери –

10 ефективних комп’ютерів для відділу моделювання та 21 комп’ютер для інших відділів компанії. Розробимо конфігурацію робочих станцій [1].

Конфігурація робочих станцій

Робоча станція - це комп'ютер або термінал з необхідним програмним забезпеченням і допоміжними пристроями.

Таблиця 2.1 – Робочі станції для відділу моделювання

№	Комплектуючі	Найменування	Характеристика
1	Корпус	GreenVision GV–CS F01	ATX,mATX; 2 x USB 2.0, 2 x аудіороз’єма
2	Материнська плата	MSI H61M–P31/W8	MicroATX, Socket 1155, Intel Core i3 / i5 / i7 / Pentium / Celeron, Intel H61, максимальний об’єм оперативної пам’яті 16 Гб
3	Блок живлення	Aerocool VX–500	500 Вт
4	Процесор	Intel Core i7–6700	Socket 1151, 4 ГГц, 4 ядра
5	Оперативна пам’ять	Kingston DDR4–2400 8192MB PC4–19200	DDR3 SDRAM, 16 ГБ, 1600 МГц
6	Відеокарта	Asus GeForce GTX 1050	4GB GDDR5 (128bit) (1290/7008) (DVI, HDMI, DisplayPort)
7	HDD	Western Digital Blue 500GB	500 ГБ, SATAIII
8	Монітор	Acer V196HQLAb	18.5", 16:9, 1366x768, VGA
9	Маніпулятор	A4Tech N–330 Glossy Grey	USB, 1000 dpi, 3 клавіш
10	Клавіатура	A4 Tech KL–40 USB X–Slim	USB, 117 клавіш
11	Ціна	21410 грн	

В інших відділах співробітники працюють в основному з базами даних, інтернет-контентом тощо, тому робочі станції мають слабший процесор і блок живлення, характеристики операторських посад представлені в таблиці.

Таблиця 2.2 – Робочі станції працівників відділів

№	Комплектуючі	Найменування	Характеристика
1	Корпус	GreenVision GV–CS F01	ATX,mATX; 2 x USB 2.0, 2 x аудіороз’єма
2	Материнська плата	MSI H61M–P31/W8	MicroATX, Socket 1155, Intel Core i3 / i5 / i7 / Pentium / Celeron, Intel H61, максимальний об’єм оперативної пам’яті 16 Гб,
3	Блок живлення	Aerocool VX–500	504.0 Вт
4	Процесор	Intel Core i3–7100	Socket 1151, 3,9 ГГц, 2 ядра
5	Оперативна пам’ять	2 x Crucial Micron DDR3–1600 2048MB PC3–12800	DDR3 SDRAM, 4 ГБ, 1600 МГц
6	HDD	Western Digital Blue 500GB	500 ГБ, SATAIII
7	Монітор	Acer V196HQLAb	18.5", 16:9, 1366x768, VGA
8	Маніпулятор	A4Tech N–330 Glossy Grey	USB, 1000 dpi, 3 клавіш
9	Клавіатура	A4 Tech KL–40 USB X–Slim Black	USB, 117 клавіш
10	Ціна	9700 грн	

2.1.4 **Конфігурація сервера**. Сервером може бути комп’ютер, який може працювати без постійного втручання людини, основною функцією сервера є робота з клієнтами (робочими станціями): надання доступу до мережі Інтернет, надання доступу до баз даних тощо.

У будівлі буде встановлено 2 сервери: Інтернет (таблиця 2.6) та файловий (таблиця 2.7)

Для роботи з сервером використовуються додатки файлового сервера, ними називають додатки, які за структурою схожі на локальні додатки, але використовують мережевий ресурс для зберігання даних у вигляді окремих файлів. У цьому випадку функції сервера зазвичай зводяться до зберігання даних (можливо також зберігання виконуваних файлів), а обробка даних здійснюється тільки на стороні клієнта [2].

Таблиця 2.3 – Інтернет сервер

№	Комплектуючі	Характеристика
1	Процесор	2 x Intel Xeon E5–2680 (2.7 Ггц, 8 ядер, LGA2011 Socket)
2	Оперативна пам’ять	DDR3 SDRAM, 16 Гб встановлено, можна встановити до 384GB, 1600МГц
3	Контролер пам’яті	LSI MegaRAID SAS 9266CV–8i
4	HDD	Western Digital Red (1 ТБ, SATAIII)
5	Відеокарта	Matrox G200e (32 Мбайт SGRAM, 250 МГц, інтерфейс VGA)
6	Інтерфейси	1 x SATAl; 2 x USB 2.0; 1 x management – Ethernet 10Base–T/100Base–TX/1000Base–T; 1 x VGA; 4 x LAN (Gigabit Ethernet); 1 x KVM
7	Мережеві порти	4 x Gigabit Ethernet
8	Ціна	25800 грн

Таблиця 2.4 – Файловий сервер

№	Комплектуючі	Характеристика
1	2	3
1	Процесор	2 x Intel Xeon E5–2680 (2.7 Ггц, 8 ядер, LGA2011 Socket)
2	Оперативна пам’ять	DDR3 SDRAM, 16 Гб встановлено, можна встановити до 384GB, 1600МГц
3	Контролер пам’яті	LSI MegaRAID SAS 9266CV–8i
4	HDD	3 x Western Digital Red (1 ТБ, SATAIII)
5	Відеокарта	Matrox G200e (32 Мбайт SGRAM, 250 МГц, інтерфейс VGA)
6	Інтерфейси	1 x SATAl; 2 x USB 2.0; 1 x management – Ethernet 10Base–T/100Base–TX/1000Base–T; 1 x VGA; 4 x LAN (Gigabit Ethernet); 1 x KVM
7	Мережеві порти	4x Gigabit Ethernet
8	Ціна	29100 грн

2.1.5 **Активні мережеві пристрої.** Мережеве обладнання - обладнання, необхідне для роботи комп'ютерної мережі, наприклад, маршрутизатор, комутатор, концентратор, комутаційна панель тощо. Зазвичай розрізняють активні та пасивні мережеві пристрої.

Назва активного мережевого обладнання стосується деяких «розумних» функцій мережевого обладнання. Це такі пристрої, як маршрутизатори, комутатори і т. д. Активне обладнання має одну особливість, яка відрізняє його від пасивного мережевого обладнання - воно споживає електроенергію, яка є необхідним джерелом енергії.

У цій роботі необхідно вибрати комутатори, сервери і точку доступу.

Комутатор — це пристрій, який використовується для створення хостів у комп’ютерній мережі. Його можна використовувати для підключення одного або кількох сегментів мережі.

Таблиця 2.5 – Мережеве обладнання

№	Найменування обладнання	Характеристика
1	Коммутатор D–Link DGS–1100–10/ME x4	2 x combo10/100/1000BASE–T/SFP 8 x Gigabit Ethernet (10/100/1000 Мбит/с) Port Security, до 64 MAC–адресов на порт Сегментация трафика D–Link Safeguard Engine
2	Маршрутизатор MikroTik RB2011UiAS–IN x1	5 x LAN 10/100, 5 x LAN 1000, 1 x SFP, 1 x Серийный порт RJ45,1 x microUSB, RouterOS 5 Atheros AR9344, 600 МГц, ОЗУ 128 МБ, ПЗУ 128 МБ
3	Точка доступа Ubiquiti UniFi AP AC Lite x2	Speed up to 867 Mb/,Channel 2.4 ГГц–5 ГГц, AirOS
4	Загальна ціна	16270 грн

2.1.6 **Вибір пасивних мережевих пристроїв**. Пасивний мережевий пристрій – це пристрій, який не споживає електроенергію. Пасивні мережеві пристрої – це власне пристрої, основною функцією яких є забезпечення передачі сигналу, тобто дроти та пристрої для їх організації та захисту: кабель, з’єднувальні кабелі, розетки, роз’єми, з’єднувальні панелі.

Сьогодні більшість комп’ютерних мереж використовують для з’єднання дроти або кабелі. Вони діють як середовище для передачі сигналу між комп’ютерами.

У даній роботі буде використовуватися кабель вита пара (рис. 2.4). Цей тип кабелю використовується для монтажу простих і найдешевших локальних мереж, а відстань між сусідніми комп'ютерами в цьому випадку не перевищує 100 м. Кабель такого типу зазвичай містить дві (або чотири) пари скручених проводів.

Рисунок 2.3 – Кручена пара

У цьому проекті, згідно з функціональною схемою, має бути використано 210,3 м крученої пари. Тому що в 1 кабельній бухті 305 метрів ми займемо 1 бухту.

Коннектор.

З'єднувач (рис. 2.5) - це з'єднувач, розташований на кінці плашки. Роз'єм RJ 45 в стандарті вита пара (UTP або STP) обжимається за кольоровою схемою.

Рисунок 2.4 – Конектор RJ-45

Захист мережевого кабелю від пошкоджень і негативного впливу зовнішнього середовища здійснюється за допомогою лотків або лотків. Бувають пластикові і металеві для укладання всередині приміщень і зовні. Для економії коштів для прокладки кабелю були обрані пластикові лотки [1].

Мережеві розетки є кінцевою точкою, куди досягає кабельний канал або кабель. Основне завдання мережевих розеток - організувати мережеві кабелі в приміщенні і забезпечити надійне кріплення роз'ємів з'єднувального кабелю. У кожній кімнаті підприємства буде встановлена мережева розетка, що дозволить у разі потреби проводити подальшу реконфігурацію мережі.

2.1.7 Розрахунок вартості розвитку мережі. Результати розрахунку вартості розвитку мережі представлені в таблиці

Таблиця 2.6 – Розрахунок витрат на розробку мережі

Витрати	Сума витрат, грн
Активне мережеве обладнання	16270
Устаткування	472700
Пасивне мережеве обладнання	4380
Всього	493350

2.1.7 **Моделювання комп'ютерних мереж у Packet Tracer**. Packet Tracer — це емулятор мережі передачі даних, створений Cisco Systems. Він дозволяє створювати робочі моделі мережі, налаштовувати (за допомогою команд Cisco IOS) маршрутизатори та комутатори, спілкуватися між кількома користувачами (через Інтернет). Включає маршрутизатори серії Cisco 1800, 2600, 2800 і комутатори 2950, 2960, 3560. Також доступні сервери DHCP, HTTP, TFTP, FTP, робочі станції, різні модулі ПК і маршрутизаторів, пристрої WiFi, різні кабелі.

Програмні засоби дозволяють збирати актуальні дані про існуючу мережу без зупинки її роботи, створювати проект цієї мережі та проводити необхідні експерименти для визначення граничних характеристик, розширюваності, змінювати топологію та модифікувати обладнання мережі для її подальшого вдосконалення та розвитку. .

За допомогою Packet Tracer можна проектувати комп’ютерні мережі різного масштабу та призначення: від локальних мереж з кількома десятками комп’ютерів до міжнародних глобальних мереж, побудованих із застосуванням супутникового зв’язку. Програмне забезпечення містить потужну базу даних мережевих пристроїв провідних виробників: робочі станції, сервери, середовища передачі для різних типів мереж і мережевих технологій.

Переходимо до створення з'єднань між пристроями. Виберіть елемент зв’язку, виберіть пристрої, виберіть тип підключення (рис. 2.6).

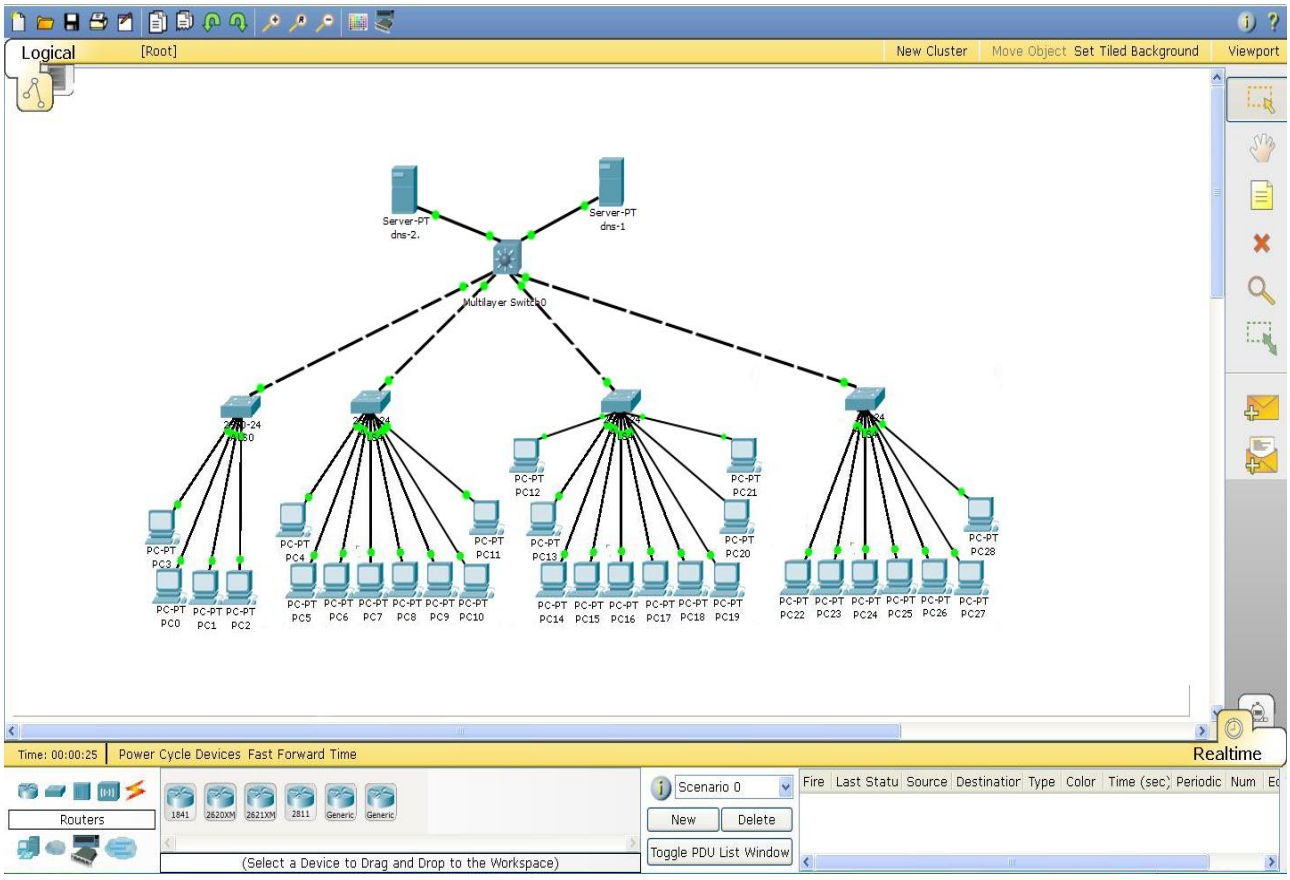

Рисунок 2.5 – Схема мережі в програмі Packet Tracer

2.2 Захист інформації

2.2.1 **Характеристика даних у корпоративній мережі**. Інформація, що циркулює в корпоративній мережі, поділяється на:

1. Відкрити:

• Інформація як об'єкт цивільних прав (патенти, авторські свідоцтва тощо);

• Масова інформація

• Інформація про вибори

• Офіційні документи

2. Конфіденційно:

• Виробнича та комерційна таємниця

• Ділова інформація та контакти

• Інформація про проект виробництва

3. Секрет

• Персональна інформація

• Професійна таємниця

Крім того, в таблиці представлено опис інформації, що передається по мережі, з точки зору поділу компанії на відділи [3].

Таблиця 2.6 –Характеристика інформації в мережі за ступенем секретності

Назва відділу	**Характеристика інформації**
Відділ Моделювання	Конфіденційна,
Відділ Продаж	Конфіденційна, відкрита
Бухгалтерія	Конфіденційна, секретна, відкрита
IT відділ	Конфіденційна, секретна
Відділ кадрів	Конфіденційна, відкрита, секретна

2.2.2 **Інженерно-технічний захист**. Інженерно-технічний захист - це сукупність спеціальних органів, технічних засобів і засобів їх використання для захисту конфіденційної інформації.

Інженерна - запобігає руйнуванню середовища в результаті навмисної дії або природного впливу інженерно-технічними засобами (сюди входять захисні споруди, системи охоронної та пожежної сигналізації).

Серед основних напрямків захисту інформації, крім організаційного захисту, виділяють правовий та інженерно-технічний захист інформації.

Для інженерно-технічного захисту на підприємстві рекомендується встановити:

• пожежні датчики;

• Відеоспостереження;

• датчики сигналізації;

• огорожі та грати на вікнах.

Для криптографічного захисту підприємства рекомендується:

• Шифрування конфіденційної інформації

• Шифрування паролів і логінів

Крім інженерно-технічного захисту комп'ютерного та мережевого обладнання:

• серверна шафа: 46U 600x800 мм 19 з додатковими замками безпеки;

• замки, що захищають від переміщення робочих комп'ютерів і носіїв інформації [4].

2.2.3 **Організаційний захист. Організаційний захист** — це регламентація виробничої діяльності та відносин між виконавцями на нормативно-правовій основі, яка виключає або істотно обмежує незаконне

володіння конфіденційною інформацією та прояви внутрішніх і зовнішніх загроз.

Організаційна інформаційна безпека – це організаційний початок, т. зв «ядро» в загальній системі захисту конфіденційної інформації компанії. Від повноти та якості вирішення організаційних завдань керівництвом та посадовими особами підприємства залежить ефективність функціонування системи захисту інформації в цілому. Роль і місце захисту організаційної інформації в загальній системі заходів захисту конфіденційної інформації підприємства визначається особливою важливістю прийняття своєчасних і правильних управлінських рішень з урахуванням сил, засобів, способів і методів. захист інформації, що здійснюється і базується на діючих нормативно-методичних заходах

Для створення організаційного захисту на підприємстві рекомендується виконати такі дії:

• режим і організація охорони;

• організація роботи з працівниками;

• організація роботи з документами та документованою інформацією;

• організація роботи з аналізу внутрішніх і зовнішніх загроз;

• організація систематичної контрольної роботи.

Використовуйте висококласний маршрутизатор Cisco ISR серії 3800 як шлюз VPN головного офісу компанії для агрегування трафіку від філій, партнерів і домашніх офісів.

Різноманітні програми для персоналу, бізнес-мандрівників і мобільних телефонів VPN.

У філіях маршрутизатори Cisco ISR серії 1800, серії 890c або серії 880c можна вибрати відповідно до масштабу та умов застосування організації:

Як VPN-шлюз.

Люди, які працюють вдома, можуть вибрати маршрутизатор Cisco ISR серії 880c як шлюз VPN.

1. Безпека

Хоча існує багато технологій і методів реалізації VPN, усі VPN повинні гарантувати специфічність і безпеку передачі даних через платформу загальнодоступної мережі.

З точки зору безпеки, оскільки VPN безпосередньо побудований на загальнодоступній мережі, він простий, зручний і гнучкий у реалізації, але в той же час проблеми безпеки є більш помітними.

Підприємства повинні гарантувати, що зловмисники не можуть підглянути та підробити дані, що передаються через їх VPN, і запобігати доступу незаконних користувачів до мережевих ресурсів або приватної інформації.

2. Якість обслуговування (QoS).

Мережа VPN повинна забезпечувати різні рівні гарантії якості обслуговування корпоративних даних. Різні користувачі та підприємства мають різні вимоги до забезпечення якості послуг.

великий. З точки зору оптимізації мережі, ще однією важливою вимогою для створення VPN є повне й ефективне використання обмежених ресурсів WAN і надання надійних даних для важливих даних.

пропускна здатність. Невизначеність трафіку WAN робить коефіцієнт використання пропускної здатності дуже низьким, спричиняючи перевантаження мережі під час піків трафіку, що робить дані з високими вимогами до реального часу.

Дані не можуть бути надіслані вчасно; велика кількість пропускної здатності мережі не використовується, коли трафік низький. QoS може використовувати стратегії прогнозування та контролю трафіку

Управління смугою пропускання реалізовано відповідно до пріоритету, щоб різні дані могли надсилатися в прийнятному порядку та запобігати виникненню перевантаження.

3. Масштабованість і гнучкість

VPN має підтримувати будь-який тип потоку даних через інтранет і екстранет, сприяти додаванню нових вузлів і підтримувати кілька типів передачі.

Середовище передачі може задовольнити вимоги щодо високоякісної передачі та збільшення пропускної здатності нових додатків, таких як одночасна передача голосу, зображення та даних.

4. Керованість.

Ним має бути легко керувати та підтримувати з точки зору користувачів і операторів. Цілями управління VPN є: зменшити мережевий ризик, мати високе розширення

Переваги безпеки, економічності та високої надійності. Насправді управління VPN в основному включає управління безпекою, керування пристроями, керування конфігурацією, список контролю доступу

Керування, управління QoS тощо.

5. Кілька методів підключення VPN.

Рішення Cisco VPN, засноване на маршрутизаторах ISR, надає зростаючим корпоративним користувачам різноманітні методи підключення VPN на вибір.

Маршрутизатори Cisco ISR розгортаються у філіях для забезпечення безпечного доступу до Інтернету.

Пропонує розширений брандмауер, здатний контролювати електронну пошту, обмін миттєвими повідомленнями та HTTP-трафік.

Політика безпеки від хробаків може бути реалізована на локальних пристроях філій, зберігаючи пропускну здатність глобальної мережі.

Система запобігання вторгненням (IPS): Ця функція глибокої перевірки пакетів може ефективно запобігати різним мережевим атакам.

Фільтрування вмісту: інтегроване рішення безпеки на основі підписки, яке надає рейтинги на основі категорій, блокування ключових слів і захист реклами

зловмисне програмне забезпечення, шкідливе програмне забезпечення, шпигунське програмне забезпечення та блокування URL-адрес:

У багатофілійному підприємстві головний офіс компанії вже розгорнув бездротову мережу, але чи може віддалена філія чи невеликий офіс використовувати ту саму бездротову мережу.

Розгорнути з тією самою бездротовою політикою? І дозвольте людям, які подорожують у відділеннях, використовувати бездротові мережеві з’єднання так, ніби вони все ще перебувають у своїх офісах.

Нове покоління Cisco ISR880c/890c забезпечує бездротове рішення для віддалених легких точок доступу Маршрутизатори ISR розгортаються у філіях підприємств.

Після встановлення захищеного VPN-з'єднання зі штаб-квартирою компанії. Бездротовий контролер у головному офісі підприємства може використовувати технологію Cisco Lightweight Access Point Protocol (LWAPP).

2.2.4 **Захист програмного забезпечення.** Антивірусний захист. Для захисту комп’ютерів від шкідливих програм на підприємстві слід встановити антивірус Microsoft Security Essentials.

Переваги:

• відомий розробник;

• великі, постійно оновлювані антивірусні бази;

• переважно те, що система не завантажена;

• простий і зрозумілий інтерфейс.

Антивірус абсолютно безкоштовний, що економить ще більше грошей. Також окремо встановимо антивірусну програму для захисту сервера. Безпека файлового сервера Avast для серверів Windows. Ціна цього антивіруса становить 400 доларів на рік.

Записи облікового запису Запис облікового запису - це запис, що містить інформацію, необхідну для ідентифікації користувача при підключенні до системи, а також інформацію для авторизації та розрахунків. Це ім’я користувача та пароль (або інший подібний метод автентифікації – наприклад, біометричні дані).

Пароль або його еквівалент зазвичай зберігається в зашифрованому або зашифрованому вигляді (для його безпеки).

Ідентифікація суб'єкта доступу полягає в тому, що суб'єкт повідомляє операційній системі про себе ідентифікаційні дані (ім'я, номер рахунку тощо) і таким чином ідентифікує себе.

Автентифікація суб'єкта доступу полягає в тому, що суб'єкт надає операційній системі, крім ідентифікаційної інформації, дані аутентифікації, які підтверджують, що він дійсно є суб'єктом доступу, якого стосується ідентифікаційна інформація.

Авторизація сутності доступу відбувається після успішної ідентифікації та автентифікації. Коли суб'єкт авторизований, операційна система виконує дії, необхідні суб'єкту для початку роботи в системі [3].

Одним із важливих елементів комплексного захисту є доступ на рівні операційної системи. Тому я вважаю за необхідне створити облікові записи користувачів і передати ідентифікацію, автентифікацію та авторизацію користувачів, які працюють у цій операційній системі.

2.2.5 **Криптологічний захист інформації**. Криптологія - це наука про захист інформації шляхом її перетворення. Криптологія об'єднує дві галузі - криптографію і криптоаналіз.

Криптографія займається пошуком і вивченням методів перетворення інформації з метою приховування її змісту. Основними сферами застосування криптографічних методів є передача конфіденційної інформації по каналах зв'язку, перевірка достовірності надісланих повідомлень, зберігання інформації (документів, баз даних) на носіях у зашифрованому вигляді.

Для криптологічного захисту інформації на цьому підприємстві пропонується шифрувати секретну інформацію, таку як паролі та логінні дані співробітників, секретне листування, що циркулює на підприємстві, тощо.

Обмеження доступу - це набір процедур, які реалізують перевірку та оцінку запитів доступу на основі правил обмеження доступу. Правила розмежування доступу є частиною політики безпеки, яка регулює доступ користувачів і процесів до пасивних об’єктів[4].

Коли користувачі або процеси намагаються отримати доступ до пасивних об'єктів, механізми, що реалізують контроль доступу, можуть "вирішити" легітимність запиту на основі політики безпеки та контролю атрибутів доступу. Використовуючи набір атрибутів доступу відповідно до прийнятої політики безпеки, можна реалізувати довірений контроль доступу,

адміністративний контроль доступу, контроль цілісності та інші види контролю доступу.

Поняття матриці доступу використовується для відображення функціональності комп'ютерної системи. Матриця доступу - це таблиця, по якій розміщені ідентифікатори об'єктів комп'ютерної системи, а елементи матриці мають включені або вимкнені режими доступу. Матриця доступу може бути двовимірною (наприклад, користувачі/пасивні об’єкти або процеси/пасивні об’єкти) або тривимірною (користувачі/процеси/пасивні об’єкти).

Таблиця 2.7 – Матриця доступу

Посада	Дані на сервері		
	Відкрита інформація	Конфіденційна інформація	Секретна інформація
Начальник підприємства	Зчитування, запис (власні дані)	Зчитування, запис (власні дані)	Зчитування, запис (власні дані)
Працівник IT відділу	Зчитування, редагування, запис (власні дані)	Зчитування, редагування	Зчитування, запис (власні дані)
Користувач	Зчитування, запис (власні дані)	Зчитування, запис (власні дані)	Запис (власні дані)

Висновки до другого розділу

Даний набір пристроїв призначений тільки для одного конкретного приміщення, щоб система кіберзахисту могла забезпечити повну безпеку.

Система кібербезпеки досить складна в обладнанні, адже в ній є маса датчиків, контролерів та сенсорів, які завжди передають інформацію на керуючий пристрій, котрий в подальшому після обробки інформації передає

кінцеві команди на пристрої. Система може працювати в автоматичному режимі, але завжди є інформація, яку потрібно повідомити користувачеві. Для цього використовують панель керування, або контрольну панель, використання якої дозволить користувачеві вручну керувати різними пристроями.

Будь-яка система комплексного кібербезпеки складається з різних компонентів, кожен з яких виконує властиві йому функції. В теперішній час ринок послуг кіберзахисту надає широкий спектр, як і самих технічних засобів, так і їх моделей, що мають різні технічні характеристики. В таких умовах вибір оптимальних захисних пристроїв є непростим завданням. Для вирішення якої пропонується використовувати комплексний метод визначення рівня якості, який дозволяє за чисельним значенням технічних характеристик різних моделей одного й того ж виду технічного засобу захисту, визначати у відносних одиницях їх рівень якості та порівнювати моделі між собою. Отримані результати дозволяють рекомендувати вище зазначені пристрої для проектування комплексної системи кіберзахисту.

РОЗДІЛ 3
ПРАКТИЧНА ЧАСТИНА РОЗРОБКИ ІНФОРМАЦІЙНОГО ЗАХИСТУ

3.1 Вибір та обґрунтування використання алгоритму шифрування RSA

RSA – криптографічна система з відкритим ключем. RSA став першим алгоритмом такого типу, придатним і для шифрування і для цифрового підпису. Алгоритм використовується у великій кількості криптографічних застосунків.

RSA схему шифрування було запропоновано у 1978 році та названо іменами трьох його винахідників: Роном Рівестом (Ron Rivest), Аді Шаміром (Adi Shamir) та Леонардом Адлеманом (Leonard Adleman). RSA належить до класу алгоритмів кодування з відкритим ключем.

У 80-х роках криптосистема переважно використовувалася для забезпечення секретності та достовірності цифрових даних. У сучасному світі RSA використовується в web-серверах та браузерах для зберігання таємності даних що передаються по мережі[7].

Одним із найнадійніших методів захисту інформаційних ресурсів інформаційно-комунікаційних систем є використання криптографічних засобів. Для забезпечення конфіденційного передавання інформації сучасна криптографія передбачає можливість використання значного розмаїття симетричних алгоритмів шифрування. До типових симетричних алгоритмів, призначених для шифрування даних, можна віднести алгоритми DES, 3DES, IDEA, AES, Twofish, Blowfish, CAST–5 (CAST–128) та інші, які можуть бути використані як самостійно, так і у режимах типу ECB, CBC, OFB та CFB. Типовою областю їх застосування є передавання даних. Проблемою, яка виникає під час передавання інформації, є надійність алгоритму, яка визначається рядом критеріїв:

- довжиною ключа,
- кількістю раундів шифрування,

- довжиною блока даних відкритого тексту,
- математичною складністю реалізації раунду шифрування тощо.

Аналіз показав, що серед найпоширеніших алгоритмів шифрування оптимальними з погляду специфіки їх роботи, рівня захисту та простоти імплементації є алгоритми AES та RSA[3,4].

Мережні рішення для бізнесу, що розвивається Cisco. Мережні рішення для бізнесу, що розвивається.

Мережні рішення Cisco для зростаючого бізнесу.

На даний момент основні елементи мережевого рішення для зростаючих підприємств включають: локальну мережу, підключення до глобальної мережі, управління мережею та безпеку. Зокрема, підприємство

Вимоги до мережі:

Встановіть захищену мережеву архітектуру та мережеві з’єднання між штаб-квартирою та філіями;Безпечне розгортання мережі для забезпечення нормальної роботи підприємства.Надання IPSec або SSL VPN для ділових мандрівників;

Забезпечувати функції інтелектуального керування та підтримувати керування графікою браузера;Конструкцію мережі легко оновити та вона сприяє захисту інвестицій.Рішення Cisco враховує ці елементи. Загалом, рішення мають такі спільні характеристики;

Він використовує високопродуктивне рішення з повним перемиканням, яке повністю відповідає потребам користувачів.Управління мережею просте, за допомогою простого у використанні браузера для керування мережею з інтуїтивно зрозумілим графічним інтерфейсом, тому мережеві адміністратори не потребують спеціального навчання.

Користувачі можуть використовувати різноманітні методи підключення WAN, тим самим зменшуючи вартість WAN-з’єднань.

Технологія стиснення пропускної здатності та застосування вдосконаленого QoS можуть ефективно зменшити трафік WAN-зв’язку.

З розвитком бізнесу компанії все мережеве обладнання може продовжувати використовуватися після оновлення початкової мережі, ефективно реалізуючи захист інвестицій.Система є безпечною та конфіденційною, і застосовано недороге рішення безпеки мережі, яке підходить для зростаючих підприємств.

Cisco ISR Router Growing Enterprise Router Growing Enterprise VPN рішення.

VPN-рішення маршрутизатора Cisco ISR для підприємств, що розвиваються.Технологія віртуальної приватної мережі (VPN) є надзвичайно економічно ефективною, порівняно з іншими рішеннями підключення до WAN, VPN може допомогти.Ви «глобалізуєте» свій бізнес швидше та економічно ефективніше, значно скорочуючи накладні витрати та отримуючи швидке повернення інвестицій. Завдяки технології VPN користувачі можуть.

Критично важливі програми поширюються на віддалені офіси, партнерські мережі, мандрівників і домашніх робітників, що робить бізнес більш конкурентоспроможним і вдосконалюється.Висока якість обслуговування клієнтів. Можна сказати, що VPN є розширенням внутрішньої мережі в публічній мережі, яка може забезпечити таку ж безпеку, керованість і передачу, як і приватна мережа.

Транспортна продуктивність, тоді як робота зі створення, експлуатації та підтримки мережі відокремлена від IT-відділу підприємства та передана спеціальному постачальнику VPN. Для плану Сюнь.

Для підприємств, які швидко розвивають глобальну електронну комерцію, вибір VPN, безсумнівно, буде розумним кроком.

Вимоги до застосування VPN для підприємств, що розвиваються.

Зараз багато підприємств, що розвиваються в Китаї, стикаються з таким викликом: філії, дистриб'ютори, партнери, клієнти та ділові мандрівники повинні.Прагнучи отримати доступ до ресурсів компанії через загальнодоступну мережу в будь-який час, ці ресурси включають: внутрішню інформацію компанії, офіс OA, систему ERP, систему CRM, електронні.Електронна пошта, система управління проектами тощо. У той же час при доступі до мережевих ресурсів має бути встановлено високонадійний канал безпеки.

Щоб задовольнити потреби додатків зростаючих підприємств, VPN, Cisco запустила комплексний корпоративний маршрутизатор з інтегрованими послугами (ISR).Комплексне рішення для рівня віртуальної приватної мережі (EVPN) з точки зору п'яти елементів реалізації VPN: масштабованої платформи, безпеки, сервісу, програми та керування.

Він має стандартну відкриту архітектуру, масштабоване та наскрізне підключення до мережі. За допомогою вдосконалених маршрутизаторів Cisco ISR для.Корпоративні користувачі, які зростають, надають різноманітні методи підключення VPN та безпечні та надійні методи шифрування, захищаючи таким чином інформаційні ресурси підприємства.

Щоб задовольнити потреби підприємств, що розвиваються, у додатках VPN, Cisco пропонує різноманітні методи підключення VPN за допомогою маршрутизаторів ISR, наприклад: безпека IP.VPN (IPSec), Secure Sockets VPN (SSL), Dynamic Multipoint VPN (DMVPN) і VPN для доступу з мобільного телефону.Рішення Cisco забезпечують високоінтелектуальну мережу, яка поєднує безпеку, бездротовий доступ і уніфіковані комунікації. Він характеризується повним використанням мережевого обладнання.Завдяки вдосконаленню модульності та відкриттю багатьох функцій програмного забезпечення пристрою, IP-телефон і бездротова точка доступу налаштовані одночасно для реалізації безпечної багатофункціональної служби мережа

підтримки. Крім того, Cisco спеціально покращила дизайн доступу до глобальної мережі та оптимізацію з’єднання WAN.

Комплексний захист безпеки перетворює IT-центр із високозатратного центру на відділ підтримки бізнесу, спрощуючи технічне обслуговування та керування та знижуючи експлуатаційні витрати.Гнучка та потужна бездротова точка доступу має широке покриття та гнучку конфігурацію для досягнення більш надійного захисту, ніж дротові точки доступу.

Високоінтелектуальна, зручна і проста уніфікована комунікаційна система легко реалізує інтерактивне робоче середовище. Це не тільки просте управління, але й витрати на інвестиції, розгортання та обслуговування голосової мережі, які можна контролювати, і її можна гнучко додавати відповідно до змін у попиті.Простий метод віддаленого доступу з високим рівнем безпеки, клієнту потрібен лише стандартний браузер, для простих користувачів віддаленого доступу (потрібно лише ввести.доступ до внутрішнього WEB підприємства, FTP-сайту або зв’язку електронною поштою), а також надавати послуги віддаленого доступу дуже економічно, що зручно для централізованого керування.

Високоефективна передача каналів WAN WAAS забезпечує прискорене використання програм для філій, інтегрує розосереджену IT-інфраструктуру в центри обробки даних і забезпечує доступПродуктивність додатків поблизу локальної мережі та ефективний контроль постійного зростання попиту на пропускну здатність, покращення захисту даних, резервного копіювання та реплікації.

Крім того, застосування мережевих рішень Cisco для малих і середніх підприємств має такі аспекти: по-перше, спільне використання ресурсів, кожен робочий стіл у мережі.Користувачі можуть спільно використовувати бази даних і принтери для реалізації різних функцій в системі автоматизації офісу; по-друге, послуги зв’язку, кінцеві користувачі через глобальну мережу.

Підключення може надсилати та отримувати електронні листи, створювати веб-додатки, отримувати доступ до Інтернету та здійснювати безпечний доступ до WAN; нарешті, мультимедійні програми.Підтримка мультимедійної багатоадресної передачі з відмінною гарантією якості обслуговування;Cisco ISR Router Secure Branch Solution Маршрутизатор Secure Branch Solution.

Cisco ISR Router Secure Branch Office Solution.

У багатофілійному підприємстві головний офіс розгорнув безпечну та надійну мережеву систему, тоді як віддалені філії або невеликі офіси.Як в офісі розгорнути відповідне мережеве обладнання для боротьби з різними поточними мережевими загрозами? Наприклад: хакерська атака, вірус-хробак, незаконна поведінка в Інтернеті тощо.Зачекайте. Ці мережеві загрози не тільки впливають на мережеву безпеку філій, але й споживають велику кількість мережевих ресурсів, серйозно впливаючи на нормальну роботу мережевих систем підприємства.

Маршрутизатори серії Cisco ISR800 забезпечують надійні мережеві рішення для малих і середніх філій і невеликих офісів для захисту різних мереж.

Використання криптосистеми RSA в даний час. Криптосистема RSA використовується в самих різних продуктах, на різних платформах і в багатьох галузях. В даний час криптосистема RSA вбудовується в різні комерційні продукти, число яких постійно збільшується. Також її використовують операційні системи Microsoft, Apple, Sun і Novell.

Технологію шифрування RSA BSAFE використовують близько 500 мільйонів користувачів усього світу. Так як в більшості випадків при цьому використовується алгоритм RSA, то його можна вважати найбільш поширеною криптосистемою загального (public) ключа в світі і ця кількість має явну тенденцію до збільшення в міру зростання Internet. RSA —

криптографічна система з відкритим ключем. RSA став першим алгоритмом такого роду, який підходить як для шифрування, так і для цифрового підпису. Алгоритм використовується в багатьох криптографічних програмах[8].

Схема шифрування RSA була розроблена в 1978 році та названа на честь трьох її винахідників: Рона Ріввеста, Аді Шаміра та Леонарда Адлемана. RSA належить до класу алгоритмів шифрування з відкритим ключем.

У 1980-х роках криптосистема в основному використовувалася для забезпечення конфіденційності та автентичності цифрових даних. У сучасному світі RSA використовується у веб-серверах і браузерах для збереження конфіденційності даних, що надсилаються через мережу,

Одним із найнадійніших методів захисту інформаційних ресурсів ІКТ-систем є використання криптографічних засобів. Для забезпечення конфіденційності передачі інформації сучасна криптографія дає можливість використовувати широкий спектр симетричних алгоритмів шифрування. Типові симетричні алгоритми шифрування даних включають DES, 3DES, IDEA, AES, Twofish, Blowfish, CAST-5 (CAST-128) та інші, які можна використовувати незалежно, а також режими ECB CBC, OFB і CFB. Типовою сферою застосування є передача даних. Проблемою, яка виникає при передачі інформації, є надійність алгоритму, яка визначається кількома критеріями:

• довжина ключа,

• кількість раундів шифрування,

• довжина блоку даних у вигляді звичайного тексту,

• математична складність реалізації раунду шифрування тощо.

Аналіз показав, що серед найпоширеніших алгоритмів шифрування оптимальними за специфікою роботи, рівнем захисту та простотою впровадження є алгоритми AES та RSA[9].

Використання криптосистеми RSA сьогодні. Криптосистема RSA використовується в багатьох різних продуктах, платформах і галузях. В даний час криптосистема RSA вбудована в різні комерційні продукти, кількість яких постійно зростає. Він також використовується операційними системами Microsoft, Apple, Sun і Novell.

Технологію шифрування RSA BSAFE використовують приблизно 500 мільйонів користувачів у всьому світі. Оскільки в більшості випадків використовується алгоритм RSA, його можна вважати найпопулярнішою криптосистемою з відкритим ключем у світі, і ця кількість явно зростає з розвитком Інтернету [8].

3.2 Вибір та обґрунтування використання мови С++ програмування для розроблення програмного продукту.

С++ — це мова програмування високого рівня, яка підтримує декілька парадигм програмування: об’єктно-орієнтовану, узагальнену та процедурну. Він був розроблений Б'ярном Страуструпом з AT&T Bell Laboratories (Мюррей-Хілл, Нью-Джерсі) у 1979 році та спочатку називався "Si with Classes". У 1983 році Страуструп перейменував мову на С++. Він заснований на мові С[10].

У 1990-х роках С++ стала однією з найпоширеніших мов програмування загального призначення. Мова використовується для системного програмування, розробки програмного забезпечення, написання драйверів, потужних серверних і клієнтських програм і створення розважальних програм, таких як відеоігри. С++ справила величезний вплив на інші популярні сьогодні мови програмування: C# і Java.

Синтаксис С++ подібний до С і Java. Мова має сувору статичну типізацію, підтримує поліморфізм, перевантаження операторів, покажчики на

функції-члени класу, атрибути, події, властивості, винятки. Перейнявши багато чого від свого попередника, мови Ci, Delphi, виходячи з практики їх використання, виключає деякі моделі, які виявилися проблематичними при розробці програмних систем [11].

3.3 **Впровадження програмного забезпечення** . Схема RSA базується на обчисленні виразів зі степенями. Відкритий текст зашифрований у вигляді блоків, кожен з яких має довжину менше певного числа n.

3.3.1 **Алгоритм генерації ключів**. Алгоритм RSA складається з 4 кроків: генерація ключа, шифрування, дешифрування та розподіл ключа.

Безпека алгоритму RSA побудована на складності цілочисельної факторизації. Алгоритм використовує два ключі - відкритий і закритий, разом відкритий і відповідний секретний ключі утворюють пари ключів. Відкритий ключ не обов'язково зберігати в секреті, він використовується для шифрування даних. Якщо повідомлення було зашифровано відкритим ключем, його можна розшифрувати лише відповідним секретним ключем[12].

3.3.2 **Розподіл ключів.** Щоб Боб міг надіслати свої секретні повідомлення, Аліса надсилає свій відкритий ключ (n, e) Бобу безпечним, але не обов’язково секретним шляхом. Секретний ключ d ніколи не розповсюджується.

3.3.3 **Генерація ключів.** Для створення пар ключів виконуються такі кроки:

1. вибираються два великих простих числа p і q, довжиною близько 512 біт кожне;

2. обчислюється їх добуток n=p*q;

3. обчислюється функція Ейлера φ(n)=(p–1)(q–1);

4. Вибрано ціле число e так, що 1<e<φ(n) і e є взаємно простими відносно φ(n);

5. За допомогою розширеного алгоритму Евкліда знаходимо таке число d, що ed=1(modφ(n)).

Число n називається модулем, а числа e і d — відкритим і таємним показниками відповідно. Пари чисел (n,e) — це відкрита частина ключа, а (n,d) — секретна частина. Числа p і q після генерації пари ключів можуть бути знищені, але за жодних обставин вони не повинні бути розкриті. [8]

3.3.4 **Шифрування.** Припустімо, Боб хоче надіслати повідомлення M Алісі. По-перше, він перетворює M на ціле число m таке, що $0 \leq m < n$, використовуючи узгоджений оборотний протокол, відомий як схема доповнення. Потім обчислює зашифрований текст c за допомогою відкритого ключа Аліси e за допомогою рівняння c=me mod n[12].

Це можна зробити досить швидко, навіть для 500-бітних чисел, використовуючи модульне піднесення до степеня. Потім Боб віддає його Алісі.Щоб розшифрувати повідомлення Боба m, Аліса повинна обчислити таку рівність: m=cd mod n.Легко переконатися, що оригінальне повідомлення відновлено під час дешифрування: cd=(me)d=med (mod n)

З умови ed=1 (mod φ(n))

випливає, що ed=kφ(n)+1 для деякого цілого k, отже med=mkφ(n)+1 (mod n)

Відповідно до теореми Ейлера:

$m\varphi(n)=1 \pmod n$, отже $mk\varphi(n)+1=m \pmod n$, $cd=m \pmod n$

Гіпотеза RSA – RSA є односторонньою перестановкою, тобто для будь-якого правильного алгоритму A ймовірність $Pr[A(n,e,c)=c1/e]$

дуже маленький, що означає, що RSA не може бути скасовано без секретної інформації d [10].

Таблиця 3.1 – Етапи роботи алгоритму шифрування

Етап	Опис операції	Результат операції
Генерація ключів	Обрати два простих різних числа	p=3557, q=2579
	Обчислити добуток	n=p*q=3557*2579=9173503
	Обчислити функцію Ейлера	$\varphi(n)=(p-1)(q-1)=9167368$
	Обрати відкриту експоненту	e=3
	Обчислити секретну експоненту	$d=e-1 \bmod \varphi(n)$ d=6111579
	Опублікувати відкритий ключ	{e,n}={3,9173503}
	Зберегти секретний ключ	{d,n}={6111579,9173503}
Шифрування	Обрати текст для шифрування	m=1111111
	Обчислити шифротекст	$c=E(m)$ $=m^e \bmod n$ $=11111^3 \bmod 9173503$ =4051753
Розшифрування	Обчислити вихідне повідомлення	$m=D(c)=$ $=c^d \bmod n$ $= 4051753^{6111579} \bmod 9173503$ =11111

В даний час RSA рекомендує для звичайних завдань ключі розміром 1024 біта, а для особливо важливих завдань – 2048 бітів

3.3.5 Блок–схема

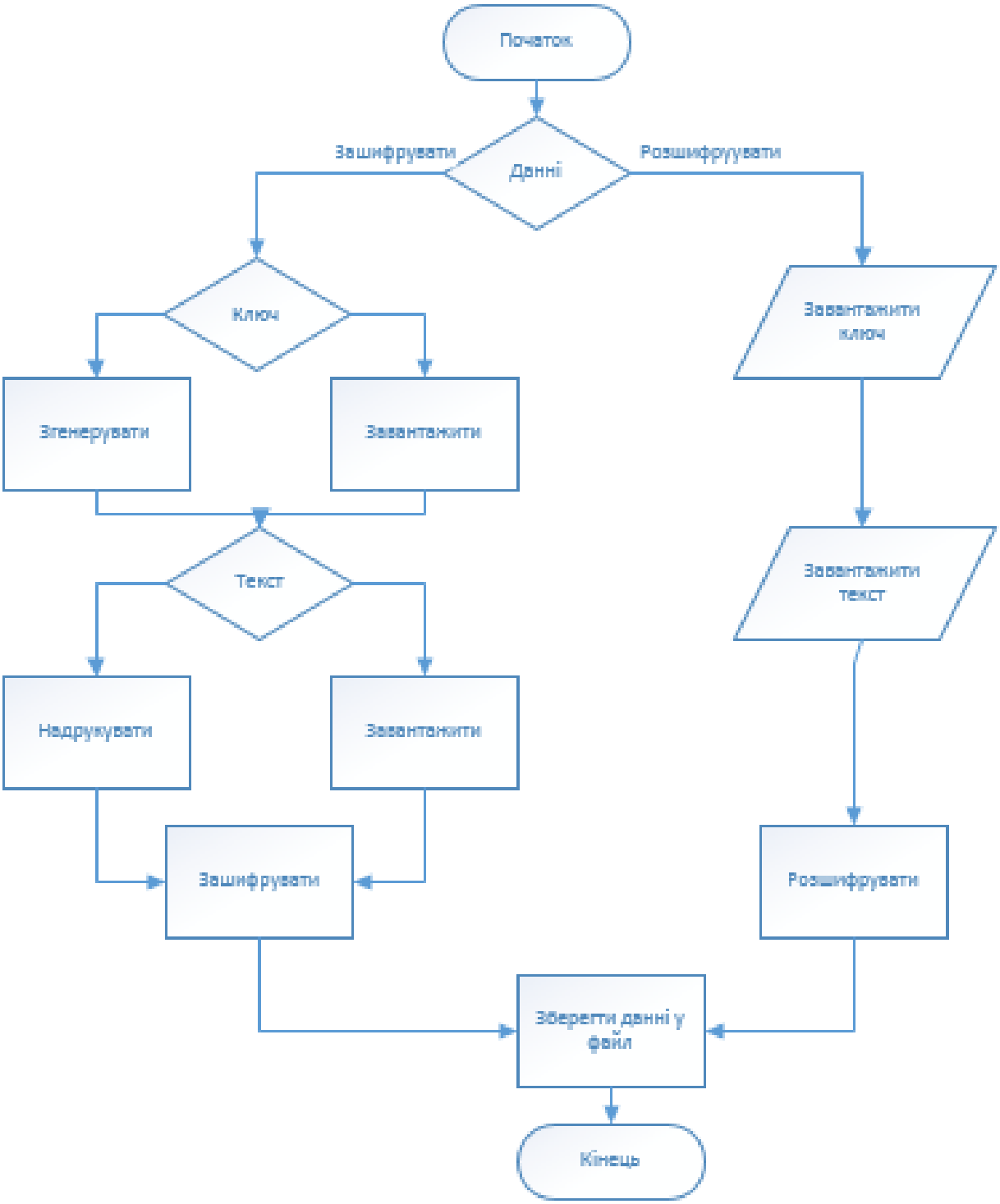

Рисунок 3.1 – Блок-схема роботи програми

3.3.6 **Опис та демонстрація програмної частини**. Розроблена програма може шифрувати та дешифрувати дані, генерувати ключі, необхідні для шифрування та дешифрування, імпортувати та експортувати вхідні та вихідні дані. Для безпечного зберігання цих даних використовуються алгоритми кодування. Використовуючи це програмне забезпечення, ви зможете надійно приховати від сторонніх осіб конфіденційні дані - банківські документи, особисті файли, пошту і т.д. Програма має 2 режими шифрування і дешифрування даних.

Далі розглянемо принцип роботи поетапно. Виконуємо програму.

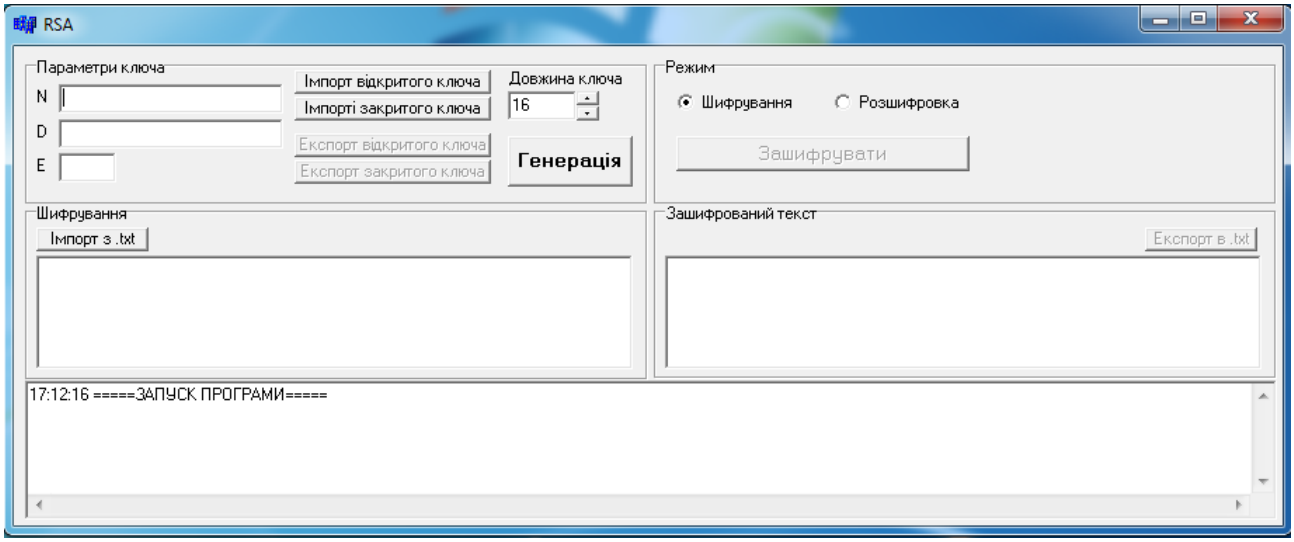

Рисунок 3.2 – Вікно програми

Далі ми повинні вибрати режим роботи програми, з якою ми хочемо працювати, натиснувши кнопку «Шифрування» або «Дешифрування» в полі «Режим» програми, або

Після вибору режиму роботи програми ми повинні згенерувати або завантажити ключ, який буде використовуватися для шифрування або дешифрування [12].

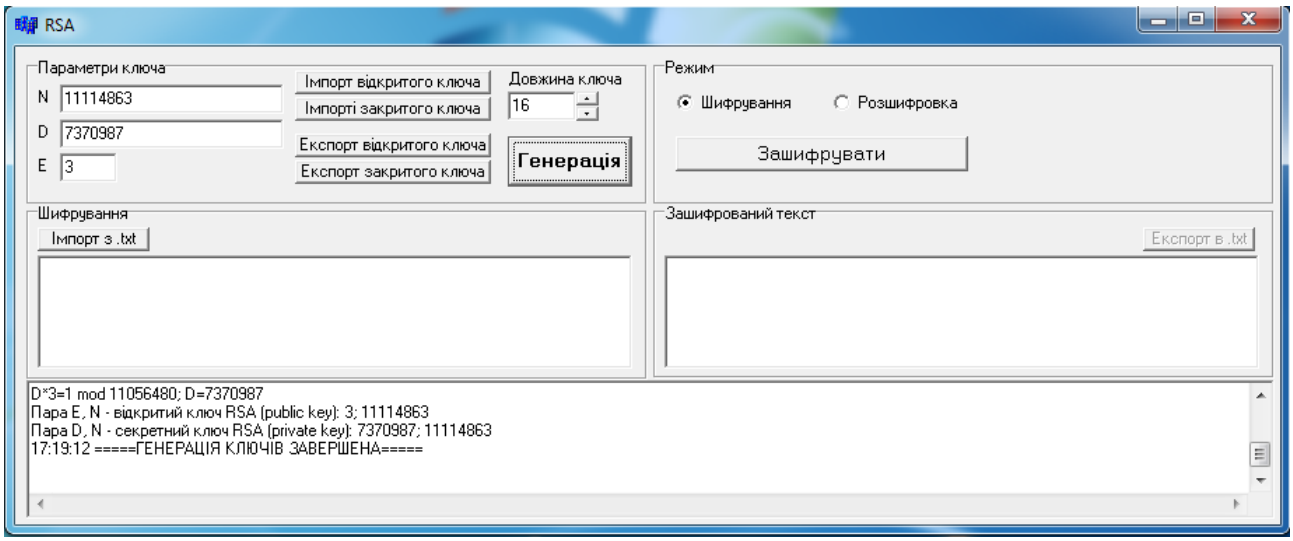

Рисунок 3.3 – Генерація ключів

Після генерації або завантаження ключів для подальшої роботи з алгоритмом шифрування даних ви можете ввести або завантажити текст для обробки, після чого натиснути кнопку шифрування:

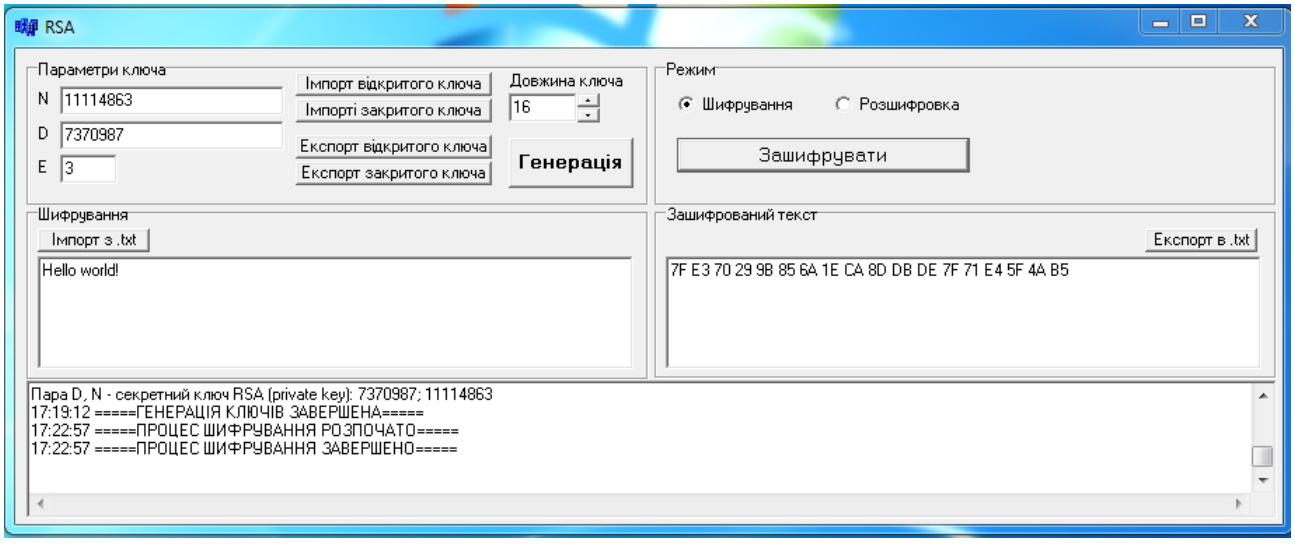

Рисунок 3.4 – Шифрування данних

Крім того, ви можете імпортувати ключ і дані, отримані під час шифрування, також можете змінити режим програми та розшифрувати зашифровані дані

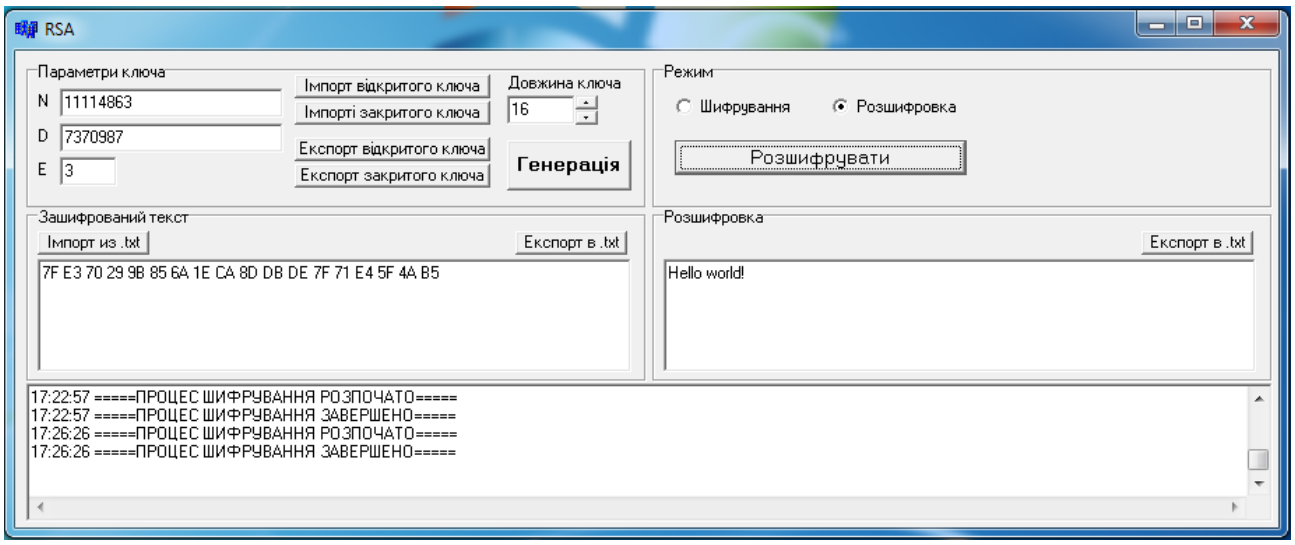

Рисунок 3.5 – Розшифровування данних

Довжину ключа можна вибирати порозрядно, крім того, програма оснащена чатом, що відображає історію програми, завдяки чому можна отримати детальну інформацію про роботу алгоритму шифрування, що особливо корисно в разі невдачі.

Висновки до третього розділу

Кожне підприємство, на якому проектується система кібербезпеки є унікальним, тому кожна проектована система є продукцією одиничного виробництва, що створюється наново для кожного конкретного підприємства. Процес проектування при розробленні системи відіграє найважливішу роль, саме на цьому етапі закладаються усі необхідні якісні характеристики системи. При проектуванні важливим питанням залишається вибір технічних засобів, з яких створюється система.

Прийняте технічне рішення ґрунтується на комплексному підході до захисту підприємства. Було розроблено структурну схему та запропоновано та обґрунтовано обладнання для її реалізації.

Спроектована охоронна система забезпечує захист від несанкціонованого проникнення об'єкт. Система включає датчики руху, датчики відкривання дверей і датчики розбиття скла, пожежний датчик. Було проведено розрахунок сигнальних рівнів та споживаної потужності. Розглянуто систему передачі сповіщень, розраховано основні параметри

сигналу. Запропоновано заходи щодо збільшення перешкодозахищеності системи. Усі підсистеми взаємодіють один з одним і можуть використовувати загальні датчики.

В будь-якому випадку система є складним технічним проектом і при її створенні потрібно використовувати різне обладнання, як по функціональному призначенню, так і обладнання від різних виробників.

ВИСНОВКИ

В рамках магістерської роботи було проаналізовано потреби підприємства та розроблено проект мережі для одноповерхової будівлі компанії «Молтехпром». Необхідне обладнання було обрано в залежності від потреб співробітників компанії, робочі станції різної продуктивності, а також веб і файлові сервери.

При побудові мережі була обрана та реалізована топологія «зірка» як найбільш економічна та оптимально заснована, активне обладнання мережі підібрано таким чином, щоб забезпечити її працездатність та оптимальний рівень продуктивності.

У другому розділі дипломної роботи були вказані витрати на запропоноване обладнання, а також усі витрати на: організацію комп’ютерної мережі, придбання та експлуатацію програмного забезпечення.

У третьому розділі дипломної роботи надано пропозиції щодо інженерно-технічного, організаційного та криптографічного захисту інформації. Крім того, в рамках роботи впроваджено рекомендації щодо криптографічного захисту, зокрема розроблено програмне забезпечення для шифрування та дешифрування даних за алгоритмом шифрування RSA. Для створення програми використовувалася мова програмування C++.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Браун С. "Мозаїка" та "Всесвітня павутина" для доступу до Internet: Пер. с англ. – М.: Світ: Маліп: СК Пресс, 2008. – 167с.

2. Джефф Форрестол, Грег Шиплі, Сканери для виявлення вад у корпоративній мережі // Мережі і системи зв'язку – # 7. – 2009. – С.114 – 124.

3. Гришина Н. В. Організація комплексної системи захисту інформації. – М. Геліос АРВ, 2007. – 256с.

4. Сьомкін С. Н., Бєляков А. В., Грибанов С. В., Козачок В. І. Основи організаційного забезпечення інформаційної безпеки об'єктів інформатизації: Навчальний посібник – М .: Геліос АРВ, 2005. – 192с.

5. Законодавча база України: за станом на 21 груд. 2017 р. / Верховна Рада України. – Офіц. вид. – К.: Парлам. вид–во, 2006. – 651 с. – (Бібліотека офіційних видань).

6. Войтенко В.В. Морозов А.В. Теорія та практика (мова С++). – Житомир, 2002.

7. Винник В. Ю. Основи програмування мовою Сі++ –. Житомир, 2008. – 311 с.

8. Павловская Т.А. С/С++. Программирование на языке высокого уровня. – СПб.: Питер, 2003. – 461с.

9. Липпман С., Лажойе Ж. Язык программирования С++. Вводный курс. – С-Пб, Невский проспект, 2006. – 1406 с.

10. Х.Дейтел, П. Дейтел. Как программировать на С++. Пятое издание – М.: Бином -Пресс, 2008. – 1456 с.

11. Стивен Прат. С++ Лекции и упражнения.6-е издание. – Вилямс, 2012. – 1298 с.

12. Скот Маэрс Эффективное использование С++. 55 верных советов улучшить структуру и код ваших программ. – третье издании е. – М.: ДМК-Прес, 2006. – 300с.

ДОДАТОК А
ПРИКЛАД ЛІСТИНГУ ПРОГРАМНОГО КОДУ

Код файлу RSA.cpp

```
//---------------------------------------------------------------------------
#include <vcl.h>
#pragma hdrstop
//---------------------------------------------------------------------------
USEFORM("UTForm.cpp", Form1);
//---------------------------------------------------------------------------
WINAPI WinMain(HINSTANCE, HINSTANCE, LPSTR, int)
{
	try
	{
		Application->Initialize();
		Application->Title = "RSA";
		Application->HelpFile = "C:\\Users\\1\\Desktop\\1352473611_icon.bmp";
		Application->CreateForm(__classid(TForm1), &Form1);
		Application->Run();
	}
	catch (Exception &exception)
	{
		Application->ShowException(&exception);
	}
	catch (...)
	{
		try
		{
			throw Exception("");
		}
		catch (Exception &exception)
		{
			Application->ShowException(&exception);
		}
	}
	return 0;
}
//---------------------------------------------------------------------------
```

Код файлу UTSimpleNum.cpp

```
//---------------------------------------------------------------------------
#pragma hdrstop
#include "UTSimpleNum.h"
#include <sysutils.hpp>
```

```
#include <math.hpp>
//---------------------------------------------------------------------------
#pragma package(smart_init)
//---------------------------------------------------------------------------
TSimpleNum::TSimpleNum(int bl)
{
BitLength = bl;

TNumber from = pow(TNumber(2),TNumber(bl - 1)) + TNumber(1),
       to   = pow(TNumber(2),TNumber(bl));
*this = rnd(from,to);
while (!SmallSNCheck() || !RabinMiller())
  *this = rnd(from,to);
}
//---------------------------------------------------------------------------
TSimpleNum::TSimpleNum(TNumber& newnum) : TNumber(newnum)
{
BitLength = 1;
while (newnum >= pow(TNumber(2),TNumber(BitLength)))
  BitLength++;
}
//---------------------------------------------------------------------------
bool TSimpleNum::SmallSNCheck()
{
for (int i = 0; i < SmallSNCount; i++)
  if ((*this % TNumber(SmallSN[i])) == TNumber(0) && *this != TNumber(SmallSN[i]))
    return false;
return true;
}
//---------------------------------------------------------------------------
bool TSimpleNum::RabinMiller()
{
TNumber p = *this;
TNumber p1 = p - TNumber(1);
TNumber b = 0;
while (p1%TNumber(2) == TNumber(0) && p1 > TNumber(0))
  {
  p1 = p1/TNumber(2);
  b++;
  }
TNumber m = p1;
AnsiString m_bin = "";
TNumber m_rest = m;
while (m_rest != TNumber(0))
  {
  if (m_rest%TNumber(2) == TNumber(1)) m_bin = "1" + m_bin;
               else m_bin = "0" + m_bin;
  m_rest = m_rest/TNumber(2);
  }
bool might_be_simple = true;
int iter = 0;
while (might_be_simple && iter < TSN_DEFAULT_RMTIMES)
```

```
  {
  iter++;
  might_be_simple = false;
  TNumber a;
  while (a >= p || a == TNumber(0))
    a = rnd(TNumber(1),pow(TNumber(2),TNumber(BitLength)));
  TNumber z = 1;
  for (int i = 1; i <= m_bin.Length(); i++)
    {
    z = (z*z)%p;
    if (m_bin[i] == '1')
      z = (z*a)%p;
    }
  if (z == TNumber(1) || z == p - TNumber(1)) might_be_simple = true;
  TNumber j = 0;
  while (b > TNumber(0) && j < b - TNumber(1))
    {
    j++;
    z = (z*z) % p;
    if (z == p - TNumber(1))
      {
      might_be_simple = true;
      break;
      }
    }
  }
return might_be_simple;
}
//---------------------------------------------------------------------------
bool TSimpleNum::FullCheck()
{
TNumber i;
for (i = 2; *this > i; i++)
  if ((*this % i) == TNumber(0))
    return false;
return true;
}
//---------------------------------------------------------------------------
AnsiString TSimpleNum::ToStr()
{
char* res_str = *this;
AnsiString res(res_str);
return res;
}
//---------------------------------------------------------------------------
```

Код файлу UTRSA.cpp

```
//---------------------------------------------------------------------------
#pragma hdrstop
```

```
#include "UTRSA.h"
#include "UTSimpleNum.h"
#include <sysutils.hpp>
//---------------------------------------------------------------------------
#pragma package(smart_init)
_INFO Info;
//---------------------------------------------------------------------------
TRSA::TRSA(const unsigned int bl_len = TRSA_DEFAULT_BLOCK_LEN) //ГЕНЕРАЦИЯ
КЛЮЧЕЙ
{
if (bl_len % 8 != 0)
  throw Exception("Довжина блоків повинна бути кратна 8");
BlockLength = bl_len;
HaveSecretKey = true;
TSimpleNum *p = new TSimpleNum(bl_len/2),
          *q = new TSimpleNum(bl_len/2 + 8);
TNumber p1 = *(TNumber*)p,    //генерація p
        q1 = *(TNumber*)q;    //генерація q
Info.InfoP=p1;
Info.InfoQ=q1;
n = p1 * q1;                  //добуток N=p*q
TNumber phi = (p1 - TNumber(1)) * (q1 - TNumber(1));  //функція Ейлера
Info.InfoPhi=phi;
e = TNumber(2);                    // Пошук найменшого числа, взаємно простого
while (phi % e == TNumber(0)) e++; // з фи.
d = eae(phi,e);                    // Пошук секретного ключа
if (d < TNumber(0)) d = phi + d;
}
//---------------------------------------------------------------------------
TRSA::TRSA(TNumber KeyN, TNumber EKey, int bl_len = TRSA_DEFAULT_BLOCK_LEN)
{
if (bl_len % 8 != 0)
  throw Exception("Довжина блоків повинна бути кратна 8");
BlockLength = bl_len;
HaveSecretKey = false;
n = KeyN;
e = EKey;
}
//---------------------------------------------------------------------------
TRSA::TRSA(TNumber KeyN, TNumber EKey, TNumber DKey, int bl_len =
TRSA_DEFAULT_BLOCK_LEN)
{
if (bl_len % 8 != 0)
  throw Exception("Довжина блоків повинна бути кратна 8");
BlockLength = bl_len;
HaveSecretKey = true;
n = KeyN;
e = EKey;
d = DKey;
}
//---------------------------------------------------------------------------
void TRSA::Encrypt()
```

```
{
int bl_len_bytes = BlockLength / 8;
while (FPlainText.Count % bl_len_bytes != 0)
  FPlainText.AsCharString = FPlainText.AsCharString + " ";
FCryptedText.AsCharString = "";
for (int i = 0; i < FPlainText.Count / bl_len_bytes; i++)
  for (int j = 0; j <= bl_len_bytes; j++)
    FCryptedText.AsCharString = FCryptedText.AsCharString + " ";
for (int i = 0; i < FPlainText.Count / bl_len_bytes; i++)
  {
  TNumber bl = 0;
  bl += TNumber(FPlainText[i * bl_len_bytes]);
  for (int j = 1; j < bl_len_bytes; j++)
    {
    bl *= TNumber(256);
    bl += TNumber(FPlainText[i * bl_len_bytes + j]);
    }
  if (bl >= n)
    throw Exception("Помилка шифрування! Довжина ключа не відповідає ключу N");
  AnsiString e_bin = "";
  TNumber e_rest = e;
  while (e_rest != TNumber(0))   //Перетворення E в X2 систему числення (e_bin)
    {
    if (e_rest%TNumber(2) == TNumber(1)) e_bin = "1" + e_bin;
                                 else e_bin = "0" + e_bin;
    e_rest = e_rest/TNumber(2);
    }
  TNumber ebl = 1;
  for (int j = 1; j <= e_bin.Length(); j++)
    {
    ebl = (ebl*ebl)%n;
    if (e_bin[j] == '1')
      ebl = (ebl*bl)%n;
    }
 for (int j = bl_len_bytes; j >= 0; j--)
    {
    FCryptedText[i * (bl_len_bytes + 1) + j] = (int) (ebl % TNumber(256));
    ebl = ebl / TNumber(256);
    }
  }
}
//---------------------------------------------------------------------------
void TRSA::Decrypt()
{
if (!HaveSecretKey)
  throw Exception("Декодування неможливо! Немає секретного ключа!");
int bl_len_bytes = BlockLength / 8 + 1;         // Довжина блоку код. тексту більше
if (FCryptedText.Count % (bl_len_bytes) != 0)
  throw Exception("Декодування неможливо! Довжина ключа не відповідає!");
FPlainText.AsCharString = "";
for (int i = 0; i < FCryptedText.Count / bl_len_bytes; i++)
  for (int j = 0; j < bl_len_bytes - 1; j++)
```

```
    FPlainText.AsCharString = FPlainText.AsCharString + " ";
for (int i = 0; i < FCryptedText.Count / bl_len_bytes; i++)
  {
  TNumber bl = 0;
  bl += TNumber(FCryptedText[i * bl_len_bytes]);
  for (int j = 1; j < bl_len_bytes; j++)
    {
    bl *= TNumber(256);
    bl += TNumber(FCryptedText[i * bl_len_bytes + j]);
    }
  AnsiString d_bin = "";
  TNumber d_rest = d;
  while (d_rest != TNumber(0))
    {
    TNumber mod = d_rest%TNumber(2);
    bool comp = mod == TNumber(1);
    if (comp) d_bin = "1" + d_bin;
                                 else d_bin = "0" + d_bin;
    d_rest = d_rest/TNumber(2);
    }
  TNumber dbl = 1;
  for (int j = 1; j <= d_bin.Length(); j++)
    {
    dbl = (dbl*dbl)%n;
    if (d_bin[j] == '1')
      dbl = (dbl*bl)%n;
    }
  for (int j = bl_len_bytes - 2; j >= 0; j--)
    {
    FPlainText[i * (bl_len_bytes-1) + j] = (int) (dbl % TNumber(256));
    dbl = dbl / TNumber(256);
    }
  }
}
//---------------------------------------------------------------------------
void TRSA::SetPlainText(THexString Source)
{
FPlainText = Source;
Encrypt();
}
//---------------------------------------------------------------------------
void TRSA::SetCryptedText(THexString Source)
{
if (!HaveSecretKey)
  throw Exception("Декодування неможливо! Секретний ключ неправильний!");
FCryptedText = Source;
Decrypt();
}
//---------------------------------------------------------------------------
THexString TRSA::GetPlainText()
{return FPlainText;}
//---------------------------------------------------------------------------
```

```
THexString TRSA::GetCryptedText()
{return FCryptedText;}
//---------------------------------------------------------------------------
TNumber TRSA::GetKeyModul()
{return n;}
//---------------------------------------------------------------------------
TNumber TRSA::GetOpenKey()
{return e;}
//---------------------------------------------------------------------------
TNumber TRSA::GetSecretKey()
{
if (!HaveSecretKey)
  throw Exception("Секретного ключа не існує!");
return d;
}
//---------------------------------------------------------------------------
```

Код файлу UTNumber.cpp

```
//---------------------------------------------------------------------------
#pragma hdrstop
#include "UTNumber.h"
#include <sysutils.hpp>
#pragma package(smart_init)
//---------------------------------------------------------------------------
TNumber::TNumber(const char *value)
{
if (value)
  {
  vlsign = (*value == '-') ? 1:0;
  if(ispunct(*value))
    {
    vlen  = strlen(value)-1;
    vlstr = new char[vlen + 1];
    strcpy(vlstr, value+1);
    }
  else
    {
    vlen = strlen(value);
    vlstr = new char[vlen + 1];
    strcpy(vlstr, value);
    }
  strrev(vlstr);
  }
else
  {
  vlstr = new char[2];
  *vlstr = '0';
  *(vlstr+1)= '\0';
  vlen = 1;
```

```
  vlsign = 0;
  }
}

//---------------------------------------------------------------------------
TNumber::TNumber(int n)
{
int i;
if (n < 0)
  {
  vlsign = 1;
  n = (-n);
  }
else vlsign = 0;
if (n)
  {
  i = (int)log10(n)+2;
  vlstr = new char[i];
  vlen = i-1;
  i = 0;
  while (n >= 1)
    {
    vlstr[i] = n%10 + '0';
    n /= 10;
    i++;
    }
  vlstr[i] = '\0';
  }
else
  {
  vlstr = new char[2];
  *vlstr = '0'; *(vlstr+1) = '\0';
  vlen = 1;
  }
}
//---------------------------------------------------------------------------
TNumber::TNumber(const TNumber& x):vlen(x.vlen),vlsign(x.vlsign)
{
vlstr = new char[x.vlen + 1];
strcpy(vlstr, x.vlstr);
}
//---------------------------------------------------------------------------
TNumber::~TNumber()
{delete [] vlstr;}
//---------------------------------------------------------------------------
TNumber::operator int() const
{
static TNumber max0(INT_MAX);
static TNumber min0(INT_MIN+1);
int number, factor = 1;
//if (*this > max0)
//  throw Exception("Error: convert TNumber->integer incredible");
```

```
//if (*this < min0)
//  throw Exception("Error: convert TNumber->integer incredible");
number = vlstr[0] - '0';
for (int j = 1; j < vlen; j++)
  {
  factor *= 10;
  number += (vlstr[j] - '0') * factor;
  }
if (vlsign) return -number;
return number;
}
//---------------------------------------------------------------------------
TNumber::operator long() const
{
static TNumber max0(INT_MAX);
static TNumber min0(INT_MIN+1);
long number, factor = 1;

if (*this > max0)
  throw Exception("Error: convert TNumber->integer incredible");
if (*this < min0)
  throw Exception("Error: convert TNumber->integer incredible");
number = vlstr[0] - '0';
for (long j = 1; j < vlen; j++)
  {
  factor *= 10;
  number += (vlstr[j] - '0') * factor;
  }
if (vlsign) return -number;
return number;
}
//---------------------------------------------------------------------------
TNumber::operator double() const
{
double sum, factor = 1.0;
sum = double(vlstr[0] - '0');
for (int i = 1; i < vlen; i++)
  {
  factor *= 10.0;
  sum += double(vlstr[i] - '0') * factor;
  }
return sum;
}
//---------------------------------------------------------------------------
TNumber::operator char * () const
{
char *temp = new char[vlen + 1];
char *s;
if (vlen > 0)
  {
  strcpy(temp, vlstr);
  if (vlsign)
```

```
    {
    s = new char[vlen + 2];
    strcpy(s,"-");
    }
  else
    {
    s = new char[vlen + 1];
    strcpy(s,"");
    }
  strcat(s,strrev(temp));
  }
else
  {
  s = new char[2];
  strcpy(s,"0");
  }
delete [] temp;
return s;
}
//---------------------------------------------------------------------------
const TNumber & TNumber::operator = (const TNumber &rhs)
{
if (this == &rhs) return *this;
delete [] vlstr;
vlstr = new char [rhs.vlen + 1];
strcpy(vlstr, rhs.vlstr);
vlen = rhs.vlen;
vlsign = rhs.vlsign;
return *this;
}
//---------------------------------------------------------------------------
TNumber TNumber::operator -() const
{
TNumber temp(*this);
if (temp != zero) temp.vlsign = !vlsign;
return temp;
}
//---------------------------------------------------------------------------
TNumber TNumber::operator ++()
{
return *this = *this + one;
}
//---------------------------------------------------------------------------
TNumber TNumber::operator ++ (int)
{
TNumber result(*this);
*this = *this + one;
return result;
}
//---------------------------------------------------------------------------
TNumber TNumber::operator -- ()
{
```

```
return *this = *this - one;
}
//---------------------------------------------------------------------------
TNumber TNumber::operator -- (int)
{
TNumber result(*this);
*this = *this - one;
return result;
}
//---------------------------------------------------------------------------
TNumber TNumber::operator += (const TNumber& v)
{
return *this = *this + v;
}
//---------------------------------------------------------------------------
TNumber TNumber::operator -= (const TNumber& v)
{
return *this = *this - v;
}
//---------------------------------------------------------------------------
TNumber TNumber::operator *= (const TNumber& v)
{
return *this = *this * v;
}
//---------------------------------------------------------------------------
TNumber TNumber::operator /= (const TNumber& v)
{
return *this = *this / v;
}
//---------------------------------------------------------------------------
TNumber TNumber::operator %= (const TNumber& v)
{
return *this = *this % v;
}
//---------------------------------------------------------------------------
TNumber operator + (const TNumber &u, const TNumber &v)
{
if (u.vlsign ^ v.vlsign)
  {
  if (u.vlsign == 0)
    {
    TNumber t1 = u - abs(v);
    return t1;
    }
  else
    {
    TNumber t2 = v - abs(u);
    return t2;
    }
  }
int j,
    d1,
```

```
    d2,
    digitsum,
    carry = 0,
    maxlen = (u.vlen > v.vlen) ? u.vlen:v.vlen;
char *temp = new char[maxlen + 2];
for (j = 0; j < maxlen; j++)
  {
  d1 = (j > u.vlen - 1) ? 0 : u.vlstr[j] - '0';
  d2 = (j > v.vlen - 1) ? 0 : v.vlstr[j] - '0';
  digitsum = d1 + d2 + carry;
  if (digitsum >= 10)
    {
    digitsum -= 10;
    carry = 1;
    }
  else
    carry = 0;
  temp[j] = digitsum + '0';
  }
if (carry) temp[j++] = '1';
if (u.vlsign) temp [j++] = '-';
temp[j] = '\0';
u.strrev(temp);
TNumber result(temp);
delete [] temp;
return result;
}
//---------------------------------------------------------------------------
TNumber operator - (const TNumber &u, const TNumber &v)
{
if (u.vlsign ^ v.vlsign)
  {
  if (u.vlsign == 0)
    {
    TNumber t1 = u + abs(v);
    return t1;
    }
  else
    {
    TNumber t2 = -(v + abs(u));
    return t2;
    }
  }
int maxlen = (u.vlen > v.vlen) ? u.vlen:v.vlen,
    d,
    d1,
    d2,
    i,
    negative,
    borrow = 0;
char *temp = new char[maxlen + 1];
TNumber w,y;
```

```
if (u.vlsign == 0)
  if (u < v)
    {
    w = v;
    y = u;
    negative = 1;
    }
  else
    {
    w = u;
    y = v;
    negative = 0;
    }
else
  if (u < v)
    {
    w = u;
    y = v;
    negative = 1;
    }
  else
    {
    w = v;
    y = u;
    negative = 0;
    }
for (i = 0; i < maxlen; i++)
  {
  d1 = (i > w.vlen - 1) ? 0 : w.vlstr[i] - '0';
  d2 = (i > y.vlen - 1) ? 0 : y.vlstr[i] - '0';
  d = d1 - d2 - borrow;
  if (d < 0)
    {
    d += 10;
    borrow = 1;
    }
  else
    borrow = 0;
  temp[i] = d + '0';
  }
while (i - 1 > 0 && temp[i - 1] == '0')
  --i;
if (negative) temp[i++] = '-';
temp[i] = '\0';
u.strrev(temp);
TNumber result(temp);
delete [] temp;
return result;
}
//---------------------------------------------------------------------------
TNumber operator * (const TNumber &u, const TNumber &v)
{
```

```
TNumber pprod("1"), tempsum("0");
for (int j = 0; j < v.vlen; j++)
  {
  int digit = v.vlstr[j] - '0';
  pprod = u.multdigit(digit);
  pprod = pprod.mult10(j);
  tempsum += pprod;
  }
tempsum.vlsign = u.vlsign ^ v.vlsign;
return tempsum;
}
//---------------------------------------------------------------------------
TNumber operator / (const TNumber &u, const TNumber &v)
{
TNumber w, y, b, c, d, quotient = TNumber("0");
int len = u.vlen - v.vlen;
if (v == TNumber("0"))
  throw Exception("Помилка! Ділення на 0!");
w = abs(u); y = abs(v);
if (w < y) return TNumber("0");

char *temp = new char[w.vlen+1];
strcpy(temp, w.vlstr + len);
b.strrev(temp);
c = TNumber(temp);
delete [] temp;
for (int i = 0; i <= len; i++)
  {
  quotient = quotient.mult10(1);
  b = d = TNumber("0");
  while (b < c)
    {
    b = b + y;
    d = d + TNumber("1");
    }
  if (c < b)
    {
    b = b - y;
    d = d - TNumber("1");
    }
  quotient = quotient + d;
  if (i < len)
    {
    c = (c-b).mult10(1);
    c = c + TNumber(w.vlstr[len-i-1]-'0');
    }
  }
quotient.vlsign = u.vlsign^v.vlsign;
return quotient;
}
//---------------------------------------------------------------------------
TNumber operator % (const TNumber &u, const TNumber &v)
```

```
{
return (u - v * (u / v));
}
//---------------------------------------------------------------------------
int operator == (const TNumber &u, const TNumber &v)
{
return (u.vlsign == v.vlsign && !strcmp(u.vlstr, v.vlstr));
}
//---------------------------------------------------------------------------
int operator != (const TNumber &u, const TNumber &v)
{
return !(u==v);
}
//---------------------------------------------------------------------------
int operator < (const TNumber &u, const TNumber &v)
{
if  (u.vlsign < v.vlsign) return 0;
else if  (u.vlsign > v.vlsign) return 1;
if  (u.vlen < v.vlen) return (1^u.vlsign);
else if  (u.vlen > v.vlen) return (0^u.vlsign);
int temp;
char *temp1 = new char[u.vlen + 1],
    *temp2 = new char[v.vlen + 1];
strcpy(temp1, u.vlstr);
strcpy(temp2, v.vlstr);
u.strrev(temp1);
u.strrev(temp2);
temp = strcmp(temp1, temp2);
delete [] temp1;
delete [] temp2;
if (temp < 0) return (1^u.vlsign);
else if (temp > 0) return (0^u.vlsign);
    else return 0;
}
//---------------------------------------------------------------------------
int operator <= (const TNumber &u, const TNumber &v)
{
return (u < v || u == v);
}
//---------------------------------------------------------------------------
int operator > (const TNumber &u, const TNumber &v)
{
return (!(u<v) && u!=v);
}
//---------------------------------------------------------------------------
int operator >= (const TNumber &u, const TNumber &v)
{
return (u>v || u==v);
}
//---------------------------------------------------------------------------
TNumber abs(const TNumber &v)
{
```

```
TNumber u(v);
if (u.vlsign) u.vlsign = 0;
return u;
}
//---------------------------------------------------------------------------
TNumber sqrt(const TNumber &v)
{
if (v.vlsign)
  throw Exception("NaN");
int j, k = v.vlen + 1, num = k >> 1;
TNumber y, z, sum, tempsum, digitsum;
char *temp = new char[num + 1],
     *w = new char[k];
strcpy(w, v.vlstr);
k = v.vlen - 1;
j = 1;
if (v.vlen % 2)
  digitsum = TNumber(w[k--] - '0');
else
  {
  digitsum = TNumber((w[k] - '0')*10 + w[k-1] - '0');
  k -= 2;
  }
sum = z = TNumber(int(sqrt(double(digitsum))));
temp[0] = int(z) + '0';
digitsum -= z*z;
for(; j < num; j++)
  {
  digitsum = digitsum.mult10(1) + TNumber(w[k--] - '0');
  y = z + z;
  z = digitsum/y;
  tempsum = digitsum.mult10(1) + TNumber(w[k] - '0');
  digitsum = -y*z.mult10(1) + tempsum - z*z;
  while (digitsum < zero)
    {
    --z;
    digitsum = -y*z.mult10(1) + z;
    }
  --k;
  temp[j] = int(z) + '0';
  z = sum = sum.mult10(1) + z;
  }
temp[num] = '\0';
TNumber result(temp);
delete [] temp; delete [] w;
return result;
}
//---------------------------------------------------------------------------
TNumber pow(const TNumber &X, const TNumber& degree)
{
TNumber N(degree), Y("1"), x(X);
if (N == TNumber("0")) return TNumber("1");
```

```
if (N < TNumber("0")) return TNumber("0");
while(1)
  {
  if (N%TNumber("2") != TNumber("0"))
    {
    Y = Y * x;
    N = N / TNumber("2");
    if (N == TNumber("0")) return Y;
    }
  else
    N = N / TNumber("2");
  x = x * x;
  }
}
//---------------------------------------------------------------------------
double div(const TNumber &u, const TNumber &v)
{
double qq=0.0, qqscale = 1.0;
TNumber w, y, b, c;
int d, count, decno = 16;
if (v == TNumber("0"))
  throw Exception("Помилка! Ділення на 0!");
if (u == TNumber("0")) return 0.0;
w = abs(u); y = abs(v);
while(w < y)
  {
  w = w.mult10(1);
  qqscale *= 0.1;
  }
int len = w.vlen - y.vlen;
char *temp = new char[w.vlen + 1];
strcpy(temp, w.vlstr + len);
w.strrev(temp);
c = TNumber(temp);
delete [] temp;
for (int i = 0; i <= len; i++)
  {
  qq *= 10.0;
  b = TNumber("0"); d = 0;
  while (b < c)
    {
    b += y;
    d += 1;
    }
  if (c < b)
    {
    b -= y;
    d -= 1;
    }
  qq += double(d);
  c = (c-b).mult10(1);
  if (i < len)
```

```
    c += TNumber(w.vlstr[len - i - 1] - '0');
  }
qq *= qqscale; count = 0;
while (c != TNumber("0") && count < decno)
  {
  qqscale *= 0.1;
  b = TNumber("0"); d = 0;
  while (b < c)
    {
    b += y;
    d += 1;
    }
  if (c < b)
    {
    b -= y;
    d -= 1;
    }
  qq += double(d)*qqscale;
  c = (c-b).mult10(1);
  count++;
  }
if (u.vlsign^v.vlsign) qq *= (-1.0);
  return qq;
}
//---------------------------------------------------------------------------
ostream & operator << (ostream &s, const TNumber &v)
{
char *temp = new char[v.vlen + 1];
if (v.vlen > 0)
  {
  strcpy(temp, v.vlstr);
  if (v.vlsign) s << "-";
  s << v.strrev(temp);
  }
else s << "0";
delete [] temp;
return s;
}
//---------------------------------------------------------------------------
istream & operator >> (istream &s, TNumber &v)
{
char* temp;
s >> temp;
delete [] v.vlstr;
v.vlen = strlen(temp);
v.strrev(temp);
v.vlstr = new char[v.vlen + 1];
strcpy(v.vlstr, temp);
return s;
}
//---------------------------------------------------------------------------
char * TNumber::strrev(char *s) const
```

```
{
int len = strlen(s),
    len1 = len - 1,
    index,
    limit = len >> 1;
char t;
for (int i = 0; i < limit; i++)
  {
  index = len1 - i;
  t = s[index];
  s[index] = s[i];
  s[i] = t;
  }
return s;
}
//---------------------------------------------------------------------------
TNumber TNumber::multdigit(int num) const
{
int j, carry = 0;
if (num)
  {
  char *temp = new char[vlen + 2];
  for(int j = 0; j < vlen; j++)
    {
    int d1 = vlstr[j] - '0',
        digitprod = d1*num + carry;
    if (digitprod >= 10)
      {
      carry = digitprod/10;
      digitprod -= carry*10;
      }
    else carry = 0;
    temp[j] = digitprod + '0';
    }
  j = vlen;
  if (carry)
    {
    temp[j] = carry + '0';
    j++;
    }
  temp[j] = '\0';
  strrev(temp);
  TNumber result(temp);
  delete [] temp;
  return result;
  }
else
  return zero;
}
//---------------------------------------------------------------------------
TNumber TNumber::mult10(int num) const
{
```

```
if (*this != zero)
  {
  int j, dd = vlen + num, bb = vlen - 1;
  char *temp = new char [dd + 1];
  for (j = 0; j < vlen; j++)
    temp[j] = vlstr[bb-j];
  for (j = vlen; j < dd; j++)
    temp[j] = '0';
  temp[dd] = '\0';
  TNumber result(temp);
  delete [] temp;
  return result;
  }
else
  return zero;
}
//---------------------------------------------------------------------------
TNumber rnd(const TNumber& from, const TNumber& to)
{
int res_length;

while ((res_length = random(to.vlen) + 1) < from.vlen);
char *res_str = new char[res_length + 1];
TNumber res;
while (1)
  {
  for (int i = 0; i < res_length; i++)
    {
    int digit = random(10);
    if (i == 0)
      while (digit == 0) digit = random(10);

    res_str[i] = '0' + digit;
    }
  res_str[res_length] = '\0';
  res = TNumber(res_str);
  if (res >= from && res <= to) break;
  }
return res;
}
//---------------------------------------------------------------------------
TNumber eae(TNumber u, TNumber v)            // Алгоритм Евкліда
{
TNumber u1, u2, u3,
        v1, v2, v3,
        t1, t2, t3,
        q;
u1 = TNumber(1); u2 = TNumber(0); u3 = u;
v1 = TNumber(0); v2 = TNumber(1); v3 = v;
while (v3 != TNumber(0))
  {
  q = u3/v3;
```

```
  t1 = (u1 - v1*q);  t2 = (u2 - v2*q);  t3 = (u3 - v3*q);
  u1 = v1;           u2 = v2;           u3 = v3;
  v1 = t1;           v2 = t2;           v3 = t3;
  }
return u2;
}
```

Код файлу UTHexString.cpp

```
//---------------------------------------------------------------------------
#pragma hdrstop
#include "UTHexString.h"
#include <sysutils.hpp>
#include <math.hpp>
#include <math.h>
//---------------------------------------------------------------------------
#pragma package(smart_init)
//---------------------------------------------------------------------------
THexString::THexString()                //
{
el = new unsigned int[1];
dim = 0;
}
//---------------------------------------------------------------------------
THexString::THexString(const AnsiString Source)
{                                       //
el = new unsigned int[1];
AsHexString = Source;
}
//---------------------------------------------------------------------------
THexString::THexString(const THexString &Source)
{                                       //
dim = Source.dim;
el = new unsigned int[dim];
for (int i = 0; i < dim; i++)
  el[i] = Source.el[i];
}
//---------------------------------------------------------------------------
THexString::~THexString()               //
{
delete[] el;
}
//---------------------------------------------------------------------------
int THexString::HexCharToInt(const char &h)
{                                       //
if (h == '0') return 0;
if (h == '1') return 1;
if (h == '2') return 2;
if (h == '3') return 3;
if (h == '4') return 4;
```

```
if (h == '5') return 5;
if (h == '6') return 6;
if (h == '7') return 7;
if (h == '8') return 8;
if (h == '9') return 9;
if (h == 'A' || h == 'a') return 10;
if (h == 'B' || h == 'b') return 11;
if (h == 'C' || h == 'c') return 12;
if (h == 'D' || h == 'd') return 13;
if (h == 'E' || h == 'e') return 14;
if (h == 'F' || h == 'f') return 15;
return -1;
}
//---------------------------------------------------------------------------
char THexString::IntToHexChar(unsigned int digit)
{                                       // 1
if (digit > 15)
throw Exception("hex must be [0:15]");
if (digit <=  9) return AnsiString(digit)[1];
if (digit == 10) return 'A';
if (digit == 11) return 'B';
if (digit == 12) return 'C';
if (digit == 13) return 'D';
if (digit == 14) return 'E';
return 'F';
}
//---------------------------------------------------------------------------
void THexString::SetHexString(AnsiString Source)
{                                       // Запис як 16-річная строка
Source = Source.Trim();
if ((Source.Length() + 1)%3 != 0)
  throw Exception("Довжина рядка не відповідає формату");
delete[] el;
dim = (Source.Length() + 1)/3;
el = new unsigned int[dim];
unsigned int digit1,digit2;
for (int i = 1; i <= dim; i++)
  {
  digit1 = HexCharToInt(Source[3*i-2]);
  digit2 = HexCharToInt(Source[3*i-1]);
  if (digit1 == -1 || digit2 == -1)
    throw Exception("'"+Source+"' is not string of hex.");
  if (i*3 <= Source.Length())
    if (Source[i*3] != ' ')
      throw Exception("'"+Source+"' is not string of hex.");
  el[i-1] = digit1*16 + digit2;
  }
}

//---------------------------------------------------------------------------
AnsiString THexString::GetHexString()      //Переклад вихідного тексту в HEX-код
{
AnsiString result;
```

```
result = "";
int digit1,digit2;
for (int i = 0; i < dim; i++)
  {
  digit2 = el[i]%16;
  digit1 = (el[i] - el[i]%16) / 16;
  result += IntToHexChar(digit1);
  result += IntToHexChar(digit2);
  result += " ";
  }
result = result.Trim();
return result;
}
//---------------------------------------------------------------------------
AnsiString THexString::GetCharString()     //
{
AnsiString result = "";
for (int i = 0; i < dim; i++)
  result += (char)el[i];
return result;
}
//---------------------------------------------------------------------------
void THexString::SetCharString(AnsiString Source)
{                                     //
dim = Source.Length();
delete[] el;
el = new unsigned int[dim];

for (int i = 0; i < dim; i++)
  el[i] = (unsigned char)Source[i+1];
}
//---------------------------------------------------------------------------
int THexString::GetDim()                  //
{return dim;}
//---------------------------------------------------------------------------
void THexString::XOR(const THexString &op) //
{
if (dim != op.dim)
  throw Exception("Помилка!");
for (int i = 0; i < dim; i++)
  el[i] = el[i]^op.el[i];
}
//---------------------------------------------------------------------------
void THexString::AddWOExtend(const THexString &op)
{                                     //
if (dim != op.dim)
  throw Exception("Помилка!");
int rem = 0;
for (int i = 0; i < dim; i++)
  {
  el[i] = op.el[i] + el[i] + rem;
  rem = (el[i] - el[i] % 256)/256;
```

```
  el[i] = el[i] % 256;
  }
}
//---------------------------------------------------------------------------
THexString& THexString::operator = (THexString &a)
{                                     //
AsHexString = a.AsHexString;
return *this;
}
//---------------------------------------------------------------------------
AnsiString  THexString::GetBinString()
{
AnsiString Result = "";
for (int i = 0; i < dim; i++)
  {
  AnsiString res;
  int todiv = el[i];
  int rem;
  for (int j = 0; j < 8; j++)
    {
    rem = todiv%2;
    todiv = Floor(todiv/2);
    res = AnsiString(rem) + res;
    }
  Result += res;
  }
return Result;
}
//---------------------------------------------------------------------------
void THexString::SetBinString(AnsiString Source)
{
if (Source.Length() % 8 != 0)
  throw Exception("Помилка!");
dim = Source.Length() / 8;
delete[] el;
el = new unsigned int[dim];
for (int i = 0; i < dim; i++)
  {
  unsigned int value = 0;
  for (int j = 0; j < 8; j++)
    {
    if (Source[i*8 + j + 1] != '0' && Source[i*8 + j + 1] != '1')
      throw Exception("Строка повинна складатися з 0 и 1");
    if (Source[i*8 + j + 1] == '0')
      value = value * 2;
    else
      value = value * 2 + 1;
    }
  el[i] = value;
  }
}
//---------------------------------------------------------------------------
```

```
THexString& THexString::SubString(int pos, int count)
{
AnsiString ResStr;
ResStr = this->AsCharString;
ResStr = ResStr.SubString(pos,count);
THexString *Result;
Result = new THexString();
Result->AsCharString = ResStr;
return *Result;
}
//---------------------------------------------------------------------------
THexString& THexString::operator + (THexString &a)
{
AnsiString res = this->AsCharString;
res += a.AsCharString;
THexString *Result;
Result = new THexString();
Result->AsCharString = res;
return *Result;
}
//---------------------------------------------------------------------------
unsigned int& THexString::operator[] (int index)
{
return el[index];
}
//---------------------------------------------------------------------------
```