Міністерство освіти і науки України
Запорізький національний університет

Є. М. Кісельов
С. Л. Шмалій

ЗАСОБИ ЗАХИСТУ В МЕРЕЖАХ ЕЛЕКТРОННО – ОБЧИСЛЮВАЛЬНИХ МАШИН

методичні вказівки до лабораторних робіт для здобувачів ступеня вищої освіти магістра спеціальності 171«Електроніка» освітньої програми «Електроніка»

Затверджено
вченою радою ЗНУ
Протокол № 1 від 05.09.2024 р.

Запоріжжя
2024

УДК 621.38

Кісельов Є. М., Шмалій С. Л. Засоби захисту в мережах електронно – обчислювальних машин : методичні вказівки до лабораторних робіт для здобувачів ступеня вищої освіти магістра спеціальності 171«Електроніка» освітньої програми «Електроніка». Запоріжжя : ЗНУ, 2024. 95с.

Методичні вказівки розроблено для студентів денної та заочної форми навчання спеціальності 171 «Електроніка». Розглянуті вимоги до виконання лабораторного практикуму з дисципліни «Засоби захисту в мережах електронно – обчислювальних машин».

Рецензент
В.Л. Коваленко, доктор технічних наук, професор кафедри електричної інженерії та кіберфізичних систем

Відповідальний за випуск
Т.В. Критська, доктор технічних наук, професор кафедри електроніки, інформаційних систем та програмного забезпечення

ЗМІСТ

Лабораторна робота №1 «Дослідження криптостійкості паролів» 4

Лабораторна робота №2 «Захист баз даних на прикладі MS ACCESS» 16

Лабораторна робота №3 «Використання програмних засобів шифрування» 25

Лабораторна робота №4 «Робота з антивірусними програмами різних типів» 35

Лабораторна робота №5 «Робота з програмою резервного копіювання даних» 48

Лабораторна робота №6 «Конфігурація міжмережевих екранів» 57

Лабораторна робота №7 «Конфігурація списків доступу маршрутизаторів» 69

Лабораторна робота №8 «Дослідження методів видалення надмірної інформації» 78

Лабораторна робота №9 «Вивчення принципів архівації даних методами Хаффмана і Лемпеля-Зіва» 81

Лабораторна робота №10 «Вивчення потокових шифрів» 87

Перелік рекомендованих джерел 94

ЛАБОРАТОРНА РОБОТА №1

«Дослідження криптостійкості паролів»

Мета роботи: визначення криптостійкості паролів різного ступеня складності з використанням методу прямого перебору

1 Короткі теоретичні відомості

Інформація з погляду інформаційної безпеки володіє наступними категоріями:

- конфіденційність – гарантія того, що конкретна інформація доступна тільки тому колу осіб, для кого вона призначена; порушення цієї категорії називається розкраданням або розкриттям інформації;
- цілісність – гарантія того, що інформація зараз існує в її початковому вигляді, тобто при її зберіганні або передачі не було проведено несанкціонованих змін; порушення цієї категорії називається фальсифікацією повідомлення;
- аутентичність – гарантія того, що джерелом інформації є саме та особа, яка заявлена як її автор; порушення цієї категорії також називається фальсифікацією, але вже автора повідомлення;
- апелюємість – досить складна категорія, але часто вживана в електронній комерції – гарантія того, що при необхідності можна буде довести, що автором повідомлення є саме заявлена людина, і не може бути ніхто інший; відмінність цієї категорії від попередньої в тому, що при підміні автора, хтось інший намагається заявити, що він автор повідомлення, а при порушенні апелюємості – сам автор намагається "відхреститися" від своїх слів, підписаних їм одного разу.

Відносно інформаційних систем застосовуються інші категорії :

- надійність – гарантія того, що система поводиться в нормальному і позаштатному режимах так, як заплановано;
- точність – гарантія точного і повного виконання всіх команд;
- контроль доступу – гарантія того, що різні групи осіб мають різний доступ до інформаційних об'єктів, і ці обмеження доступу постійно виконуються;
- контрольованість – гарантія того, що у будь-який момент може бути проведена повноцінна перевірка будь-якого компоненту програмного комплексу;
- контроль ідентифікації – гарантія того, що клієнт, підключений в даний момент до системи, є саме тим, за кого себе видає;
- стійкість до навмисних збоїв – гарантія того, що при навмисному внесенні помилок в межах наперед обумовлених норм система поводитиметься так, як обумовлено наперед.

Кожен користувач, перш ніж здійснювати які-небудь дії в комп'ютерній системі, повинен ідентифікувати себе. У свою чергу, система повинна перевірити достовірність особи користувача, тобто що він є саме тим, за кого себе видає. Стандартним засобом перевірки достовірності (аутентифікація) – пароль, хоча у принципі можуть використовуватися також особисті картки, біометричні пристрої (сканування рогівки ока або відбитків пальців) або їх комбінація.

Перебір паролів по словнику був якийсь час однією з найпоширенішої техніки підбору паролів. Нині, як хоч найменший результат пропаганди інформаційної безпеки, він став складати свої позиції. Хоча розвиток швидкодії обчислювальної техніки і все більш складні алгоритми складання слів-паролів не дають "загинути" цьому методу. Технологія перебору паролів народилася в той час, коли найскладнішим паролем було скажемо слово "brilliant", а в русифікованих ЕОМ воно ж, але для "хитрості" набране в латинському режимі, але дивлячись на росій-

ські літери (ця тактика на жаль до цих пір надзвичайно поширена, хоч і збільшує інформаційну насиченість пароля всього на 1 біт). У той час простенька програма із словником в 5000 іменників давала позитивний результат в 60% випадків. Величезне число інцидентів із зламуваннями систем примусило користувачів додавати до слів 1-2 цифри з кінця, записувати першу і/або останню літеру у верхньому регістрі, але це збільшило час на перебір варіантів з врахуванням зростання швидкодії ЕОМ всього у декілька разів. Так в 1998 році було офіційно заявлено, що навіть складання двох цілком не зв'язаних осмислених слів підряд, не дає наскільки або реальної надійності паролю. До цього ж часу набули широкого поширення мови складання паролів, що записують в абстрактній формі основні принципи складання паролів середньостатистичними користувачами ЕОМ.

Існує ціла область знань, що розглядає питання аутентифікації, і зокрема, проблеми пов'язані з паролями. Наприклад, обговорюються необхідність і періодичність зміни паролів користувачами, вимоги до забезпечення секретності паролів, необхідність використання машинно – згенерованих (а не вибраних вручну) паролів, використання комбінованих паролів (звичайного введення символів плюс біометричні технології або введення смарт - карт), необхідність надання користувачу деякої реєстраційної інформації (дати і часу останнього входу в систему і т.п.) і ін. У цій лабораторній роботі не розглядатимуться методи отримання паролів на основі методів соціальної інженерії, які спрямовані на здобування паролів або інформації, що дозволяє отримати пароль методом відмінним від методу прямого перебору.

Однак, безумовно, однією з основних вимог є достатня криптостійкість паролів, що не дозволяє системам зламування паролів за відносно невеликий час розітнути пароль. Криптостійкість паролів визначається кількістю символів пароля (що звичайно вводяться з клавіатури), використовуваними наборами символів і кількістю наборів. Можна виділити наступні набори символів:

- заголовні (прописні) латинські літери;
- рядкові латинські літери;
- цифри;
- спеціальні символи, розділові (наприклад № % « !) знаки.

Чим більше символів містить пароль, і чим більша кількість використовуваних наборів символів, тим довше виконуватиметься злом пароля і тим вище криптостійкість пароля.

При створенні пароля необхідно враховувати, що системи злому паролів шляхом перебору часто використовують словники, що містять слова тієї або іншої мови. Тому використання паролів у вигляді слів природної мови знижує їх криптостійкість. Крім того, криптостійкій пароль, створений з урахуванням приведених вище рекомендацій, складно запам'ятати, і тому велика спокуса його фіксації на папері, що в свою чергу, різко підвищує ймовірність розкрадання пароля і реалізацію несанкціонованого доступу. Для створення одночасно і криптостійкого пароля, що запам'ятовується, рекомендується:

- набирати пароль на іншій розкладці клавіатури;
- використовувати замість літер інші схожі спеціальні символи, наприклад замість S - $, замість l – 1, замість K - |< і т.д.

Наступною модифікацією підбору паролів є перевірка паролів, що встановлюються в системах за умовчанням. В деяких випадках адміністратор програмного забезпечення, проінсталювавши або одержавши новий продукт від розробника, не спромагається перевірити, з чого складається система безпеки. Як наслідок, пароль, встановлений у фірмі розробнику за умовчанням, залишається основним паролем в системі. У Internet можна знайти величезні списки паролів за умовчанням практично до всіх версій програмного забезпечення, якщо вони встановлюються на ньому виробником.

Основні вимоги до інформаційної безпеки, засновані на аналізі даного методу, наступні:

1. Вхід всіх користувачів в систему повинен підтверджуватися введенням унікального для клієнта пароля.
2. Пароль повинен ретельно підбиратися так, щоб його інформаційна ємкість відповідала часу повного перебору пароля. Для цього необхідно детально інструктувати клієнтів про поняття "простій до підбору пароль", або передати операцію вибору пароля у ведення інженера по безпеці.
3. Паролі за умовчанням повинні бути змінені до офіційного запуску системи і навіть до прилюдних випробувань програмного комплексу. Особливе це відноситься до мережевого програмного забезпечення.
4. Всі помилкові спроби увійти до системи повинні враховуватися, заноту вуватися у файл журналу подій і аналізуватися через "розумний" проміжок часу. Якщо в системі передбачена можливість блокування клієнта або всієї системи після певної кількості невдалих спроб входу, цією можливістю необхідно скористатися. Така можливість є основним бар'єром до підбору паролів повним перебором. Розумно блокувати клієнта 3-їй підряд неправильної спроби набору пароля, і, відповідно, блокувати систему K=max(int(N*0.1*3)+1, 3) невдалих спроб входу за деякий період (годину, зміну, добу). У цій формулі N – середня кількість тих клієнтів, що підключаються за цей період до системи, 0.1 – 10%-ова межа "забудькуватості пароля", 3 – ті ж самі три спроби на пригадування пароля. Інформація про блокування клієнта або системи повинна автоматично надходити на пульт контролю за системою.
5. У момент відправки пакету підтвердження або відкидання пароля в системі повинна бути встановлена розумна затримка (2-5 секунд). Це не дозво-

лить зловмиснику, потрапивши на лінію з добрим зв'язком до об'єкту атаки перебирати по сотні тисяч паролів за секунду.

6. Всі дійсні в системі паролі бажано перевіряти сучасними програмами підбору паролів, або оцінювати особисто адміністратору системи.

7. Через певні проміжки часу необхідна примусова зміна пароля у клієнтів. Найбільш часто використовуваними інтервалами зміни пароля є рік, місяць і тиждень (залежно від рівня конфіденційності інформації і частоти входу в систему).

8. Всі невживані протягом довгого часу імена реєстрації повинні переводитися в закритий (недоступне для реєстрації) стан. Це відноситься до співробітників, що знаходяться у відпустці, хворіють, у відрядженні, а також до імен реєстрації, створених для тестів, випробувань системи і т.п.

9. Від співробітників і всіх операторів терміналу необхідно вимагати суворе нерозголошування паролів, відсутність яких-небудь взаємозв'язків пароля з широковідомими фактами і даними, і відсутність паперових записів пароля "із-за поганої пам'яті".

Наступною по частоті використання є методика отримання паролів з самої системи. Проте тут вже немає можливості дати які-небудь загальні рекомендації, оскільки всі методи атаки залежать тільки від програмної і апаратної реалізації конкретної системи. Основними двома можливостями з'ясування пароля є несанкціонований доступ до носія, що містить їх, або використання недокументованих можливостей і помилок в реалізації системи.

Отримання доступу до паролів завдяки недокументованим можливостям систем зустрічається в даний час украй рідко. Раніше ця методика використовувалася розробниками набагато частіше в основному в цілях відладки, або для екстреного відновлення працездатності системи. Але поступово з розвитком, як технологій зворотної компіляції, так і інформаційної зв'язаності світу вона поступово ста-

ла зникати. Будь-які недокументовані можливості рано чи пізно стають відомими, після чого новина про це із великою швидкістю облітає світ і розробникам доводиться розсилати всім користувачам скомпрометованої системи "програмні латочки" або нові версії програмного продукту. Єдиною мірою профілактики цього методу є постійний пошук на серверах, присвячених комп'ютерній безпеці, оголошень про всі неприємності з програмним забезпеченням. Для розробників же необхідно пам'ятати, що будь-яка подібна вбудована можливість може на порядок знизити загальну безпеку системи, як би добре вона не була завуальована в коді програмного продукту.

Наступною поширеною технологією отримання паролів є копіювання буфера клавіатури у момент набору пароля на терміналі. Цей метод використовується рідко, так для нього необхідний доступ до термінальної машини з можливістю запуску програм. Але якщо зловмисник все-таки отримує подібний доступ, дієвість даного методу дуже висока:

1. Робота програми-перехоплювача паролів (так званого "троянського коня") на робочій станції непомітна.

2. Подібна програма сама може відправляти результати роботи на заздалегідь задані сервера або анонімним користувачам, що різко спрощує саму процедуру отримання паролів хакером, і утрудняє пошук і доказ його провини. Наприклад, широкого поширення набула подібна троянська програма, що підписується до архівів, що власнорозпаковуються.

Двома основними методами боротьби з копіюванням паролів є:

1. Адекватний захист робочих станцій від запуску невідомих програм:

а) відключення змінних носіїв інформації (гнучких дисків);

б) спеціальні драйвера, блокуючи запуск здійснимих файлів без відома оператора, або адміністратора;

в) монітори, що повідомляють про будь-які зміни системних настройок і списку програм, що автоматично запускаються;

2. Дуже могутня, але незручна міра – система одноразових паролів (при кожній реєстрації в системі клієнтам з дуже високим рівнем відповідальності самою системою генерується новий пароль).

Сканування сучасними антивірусними програмами також може допомогти у виявленні "троянських" програм, але тільки тих з них, які набули широкого поширення по країні. А отже, програми, написані зловмисниками спеціально для атаки на Вашу систему, будуть пропущені антивірусними програмами без яких-небудь сигналів.

Наступний метод отримання паролів відноситься тільки до мережевого програмного забезпечення. Проблема полягає в тому, що в багатьох програмах не враховується можливість перехоплення будь-якої інформації, що йде по мережі – так званого мережевого трафіку. Спочатку, з впровадженням локальних комп'ютерних мереж так воно і було. Мережа розташовувалася в межах 2-3 кабінетів, або будівлі з обмеженим фізичним доступом до кабелів. Проте, стрімкий розвиток глобальних мереж спрямував на спільний ринок ті ж версії програмного забезпечення без якого-небудь зволікання для посилення безпеки. Більше половини протоколів мережі Інтернет передають паролі в нешифрованому вигляді – відкритим текстом. До ним відносяться протоколи передачі електронної пошти SMTP і POP3, протокол передачі файлів FTP, одна з схем авторизації на WWW-серверах.

Сучасне апаратне і програмне забезпечення дозволяє одержувати всю інформацію, що проходить по сегменту мережі, до якого підключений конкретний комп'ютер, і аналізувати її в реальному масштабі часу. Можливі декілька варіантів прослуховування трафіку: 1) це може зробити службовець компанії з свого робочого комп'ютера, 2) зловмисник, що підключився до сегменту за допомогою портативної ЕОМ або мобільного пристрою. Нарешті, трафік, що йде по мережі Ін-

тернет, технічно може прослуховуватися з боку безпосереднього провайдера, з боку будь-якої організації, що надає транспортні послуги для мережі Інтернет (листування усередині країни в середньому йде через 3-4 компанії, за межі країни – через 5-8).

Для комплексного захисту від подібної можливості крадіжки паролів необхідно виконувати наступні заходи:

1. Фізичний доступ до мережевих кабелів повинен відповідати рівню доступу до інформації.

2. При визначенні топології мережі слід при будь-якій нагоді уникати широкомовних топологій. Оптимальною одиницею сегментації є група операторів з рівними правами доступу, або якщо ця група складає більше 10 чоловік, то кімната або відділ усередині групи. У жодному випадку на одному кабелі не повинні знаходитися оператори з різними рівнями доступу, якщо тільки весь передаваний трафік не шифрується, а ідентифікація не проводиться по прихованій схемі без відкритої передачі пароля.

3. До всіх інформаційних потоків, що виходять за межі фірми, повинні застосовуватися ті ж правила, що описані вище для об'єднання різнорівневих терміналів.

Програма Advanced ZIP Password Recovery (рис. 1.1 – 1.2) є однією з програм розроблених компанією Elcom Ltd і призначених для відновлення пароля архівів, створених з використанням популярного архіватора ZIP. Програма має можливість гнучкої настройки з вказівкою довжини пароля, використовуваних наборів символів (включаючи набір користувача), застосування маски (коли відомі деякі з символів пароля і їх місце) і підключення словників.



Рисунок 1.1 – Інтерфейс програми Advanced ZIP Password Recovery

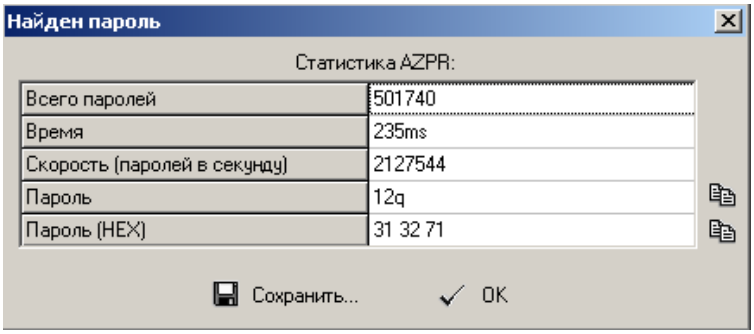


Рисунок 1.2 – Вікно з підібраним паролем і параметрами, що характеризують процес підбору

У лабораторній роботі пропонується використовувати програму Advanced ZIP Password Recovery для дослідження криптостійкости паролів з різною кількістю символів і різними наборами символів.

2 Порядок виконання роботи

1. За допомогою програми архівації даних створіть архівний файл і встановіть на нього пароль завдовжки 2 рядкових латинських символу.

2. Відкрийте створений файл в програмі Advanced ZIP Password Recovery і вкажіть використовуваний набір – рядкові латинські символи, мінімальну довжину пароля – 1, максимальну довжину пароля – 2 і виконаєте перебір. Зафіксуйте кількість перебраних паролів і час перебору в таблиці 1.1.

Таблиця 1.1 – Результати перебору паролів в програмі Advanced ZIP Password Recovery

№ п/п	Набір символів	Довжина пароля, символів	Кількість перебраних паролів	Час перебору, с	Швидкість перебору, паролей/с

3. Повторіть дослідження архівного файлу захищеного паролями довгої 3, 4, 5 і 6 рядкових латинських символу і занесіть результати в табл. 1.1.

4. За допомогою програми архівації даних створіть архівний файл і встановіть на нього пароль завдовжки 2 символи з наборів рядкових і заголовних латинських символів і занесіть результати в табл. 1.1.

5. Повторіть дослідження архівного файлу захищеного паролями довгої 3, 4, 5, 6 рядкових і заголовних латинських символів; результати занесіть в табл. 1.1.

6. За допомогою програми архівації даних створіть архівний файл і встановіть на нього пароль завдовжки 2 символи з наборів рядкових, заголовних латинських символів і цифр, виконайте дослідження криптостійкости і занесіть результати в табл. 1.1.

7. Повторіть дослідження архівного файлу захищеного паролями довгої 3, 4, 5 рядкових, заголовних латинських символів і цифр, результати занесіть в табл. 1.1.

8. Створіть за допомогою програми архівації даних архівний файл і встановите на нього пароль завдовжки 2 символи з наборів рядкових, заголовних латинських символів, цифр і спеціальних символів, виконайте дослідження криптостійкости і занесіть результати в табл. 1.1.

9. Повторіть дослідження архівного файлу захищеного паролями довгої 3, 4, 5 рядкових, заголовних латинських символів, цифр і спеціальних символів, результати занесіть в табл. 1.1.

10. Побудуйте два графіки: залежність кількості перебраних паролів від довжини пароля для різних використаних наборів символів і залежність часу підбору пароля від довжини пароля для різних використаних наборів символів.

3 Контрольні питання

1. Що називають аутентифікацією і ідентифікацією користувачів?
2. Які способи підвищення криптостійкости і надійності паролів надаються сучасними операційними системами?
3. Чим визначається криптостійкость пароля?
4. Як реалізується метод повного перебору значень пароля?
5. Які існує комбіновані методи аутентифікації.

ЛАБОРАТОРНА РОБОТА №2

“Захист баз даних на прикладі MS ACCESS”

Мета роботи: вивчення способів захисту інформації в базах даних (БД) на прикладі СУБД MS Access.

1 Короткі теоретичні відомості

Захист інформації в БД є актуальним завданням як при одноосібному використанні БД, так і при спільній роботі користувачів з нею. Захист повинен забезпечувати незмінність і цілісність БД, інформації, що міститься в ній, а також регламентувати права доступу до неї. При корпоративній роботі групи користувачів з однією базою даних необхідно вибрати адміністратора, обслуговуючого БД і доступу, що володіє відповідними правами. Права доступу користувачів встановлюються адміністратором, який може включати і виключати користувачів і розбивати їх на групи. Користувачі, що входять до складу певної групи, володіють всіма наданими їй правами. Якщо особисті права користувача вищі за права доступу групи, то особисті права за ним зберігаються.

Система безпеки БД повинна забезпечувати фізичну цілісність БД і захист від несанкціонованого вторгнення з метою читання вмісту і зміни даних.

Захист БД проводиться на двох рівнях:

- на рівні пароля;
- на рівні користувача (захист облікових записів користувачів і ідентифікованих об'єктів).

Для захисту БД Access використовує файл робочих груп system.mdw (робоча група - це група користувачів, які спільно використовують ресурси мережі), до якого БД на робочих станціях підключаються за умовчанням. Файл робочих груп містить облікові записи користувачів і груп, а також паролі користувачів. Обліко-

вим записам можуть бути надані права на доступ до БД і її об'єктів, при цьому самі дозволи на доступ зберігаються в БД.

Для забезпечення захисту БД Access необхідно створити робочу групу, використовуючи файл - адміністратор робочих груп wrkgadm.exe. При створенні унікальної робочої групи задається ім'я користувача, назва організації і код робочої групи.

Файл робочої групи MS Access містить наступні вбудовані облікові записи:

1. Admins - стандартний обліковий запис користувача. Дані записи є однаковими для всіх примірників MS Access;
2. Admin - обліковий запис групи адміністратора - є унікальним в кожному файлі робочої групи;
3. Users - містить облікові записи користувачів.

Для створення файлу робочих груп необхідно вийти з Access і в папці system або system32 у каталозі windows знайти файл робочої групи і створити нову робочу групу (може бути до 20 цифрових або буквених позначень).

Група Admins може містити довільне число користувачів, але власник об'єкту завжди один (власником об'єкту може бути обліковий запис, який створював об'єкт або якому були передані права на його використання).

Оскільки читання запису Admin можливо для всіх робочих груп і дані облікові записи є однаковими, то користувача Admin необхідно видалити з групи адміністраторів, для чого слід створити новий обліковий запис адміністратора і задати пароль на його облікові записи і на облікові записи власника.

Дозволи до доступу називаються явними, якщо вони належать або привласнені обліковому запису користувача. Дозволи будуть неявними, якщо вони привласнені обліковому запису групи, при цьому користувач, включений в групу одержує всі її дозволи.

Таблиця 2.1 – Типи дозволів на доступ до БД

Дозволи	Дозволені дії	Об'єкти БД
Відкриття і запуск	Відкриття БД, форми або звіту	БД, форми, звіти, макроси
Монопольний доступ	Монопольне відкриття БД	БД
Читання макету	Проглядання об'єктів в режимі конструктора	Таблиці, запити, форми, звіти, макроси і модулі
Зміна макетів	Перегляд і зміна макетів, видалення	Таблиці, запити, форми, звіти, макроси і модулі
Дозволи адміністратора	Установка пароля в БД, реплікація БД	Надання прав доступу іншим користувачам
Читання даних	Проглядання даних	Таблиці і запити
Оновлення даних	Перегляд і зміна даних без видалення і вставки	Таблиці і запити
Вставка даних	Перегляд і вставка даних без видалення і зміни	Таблиці, запити
Видалення даних	Перегляд і видалення даних без із зміни і вставки	Таблиці, макроси

Повноваження користувача визначаються по мінімальних дозволах доступу. Змінити дозволи для користувачів можуть члени групи Admins, власник об'єкту і користувач, що одержав на цей об'єкт дозволу адміністратора.

При підключенні до БД користувачі отримують права груп, яким вони належать.

2 Завдання до роботи

1. Створити нову базу даних з БД «Борей» і імпортувати в неї наступні об'єкти:

   - Таблиці: Заказано, Заказы, Клиенты, Товары;

   - Запити: Сведения о заказах;

-Форми: Заказы клиентов, Подчиненная форма заказов 1 и Подчиненная форма заказов 2.

2. Визначити два рівні доступу до БД:

- для читання;

- для зміни.

При виконанні захисту БД необхідно виключити доступ до інформації несанкціонованих користувачів (провести перевірку надійності захисту).

3 Алгоритм захисту БД MS Access

1. Створити нову унікальну робочу групу.

2. Створити новий обліковий запис адміністратора. Підключиться до нової робочої групи; відкрити будь-яку БД; у меню – Сервис вибрати захист і користувачів групи; створити нового користувача, ввести ім'я і код облікового запису (це не пароль); у списку наявної групи вибрати: Admins – додати.

3. Видалити з групи адміністраторів користувача Admin.

4. Вийти з Access і увійти новим користувачем в Access; обов'язково ввести пароль на даний обліковий запис.

5. Створити наново БД, яку хочемо захистити.

6. Виконати імпорт об'єктів з початкової БД до БД, створеної на попередньому кроці.

7. Виконати розподіл прав на необхідні об'єкти.

4 Порядок виконання і результати роботи

1. Захист на рівні пароля

Відкрийте БД, в пункті меню ***Сервис*** виберіть ***Защита/Задать пароль базы данных*** (див. рис. 2.1)

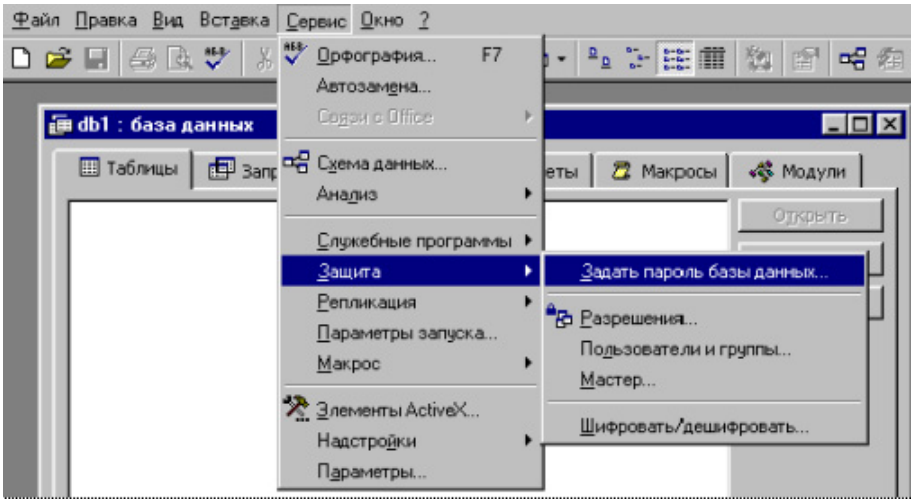


Рисунок 2.1 - Захист БД за допомогою пароля

З'явиться вікно, в якому вас попросять ввести пароль і повторити його (рис. 2.2).

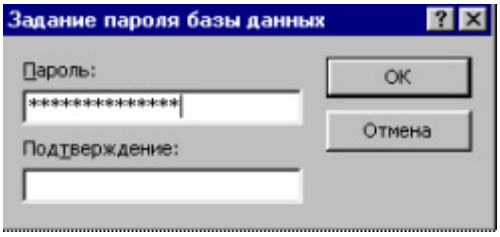


Рисунок 2.2 - Введення пароля

Рекомендації по вибору пароля:

- не бажано пароля використовувати як такі дані, як ваше ім'я, дата народження і т.д.;
- не варто вибирати короткий пароль, оскільки він може бути підібраний за допомогою спеціальних програм за достатньо короткий час;
- небажана комбінація літер і цифр, оскільки це утрудняє підбір пароля і робить даремною атаку по словнику.

2. Захист на рівні користувача

Для цього виду захисту необхідно спочатку створити нову робочу групу (якщо ви використовуватимете стару, то БД легко можна буде розкрити, оскільки в цьому випадку для алгоритму захисту братимуться дані, вказані при установці Windows або MS Access).

Для створення нової робочої групи запустите програму WRKGADM.EXE, що знаходиться в каталозі C:\Program Files\Microsoft Office\Office\1049, і натисніть кнопку ***Создать*** (рис. 2.3).

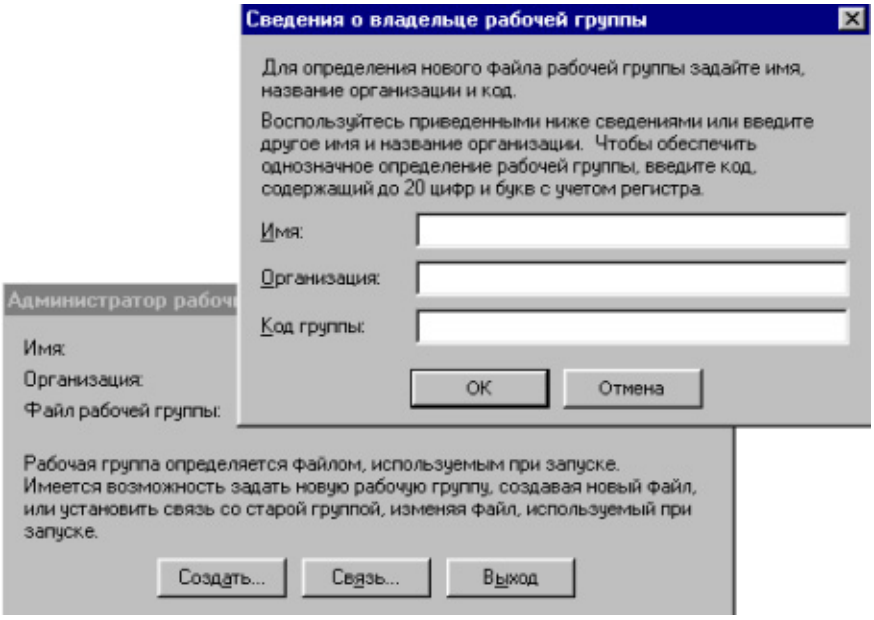


Рисунок 2.3 - Створення нової робочої групи

Далі введіть запрошувану інформацію і натисніть кнопку ***ОК.*** Задайте ім'я нової робочої групи, наприклад MY_GR.MDW (рис. 2.4).

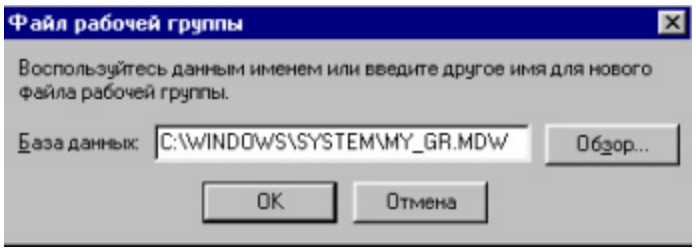


Рисунок 2.4 - Введення імені нової робочої групи

У разі правильного введення даних і їх підтвердження з'явиться повідомлення про завершення створення робочої групи. Тепер можна вийти з програми *Администратор рабочих групп.*

Запустіть БД, яку необхідно захистити. У пункті меню ***Сервис*** виберіть ***Защита/Пользователи и группы*** (рис. 2.5).

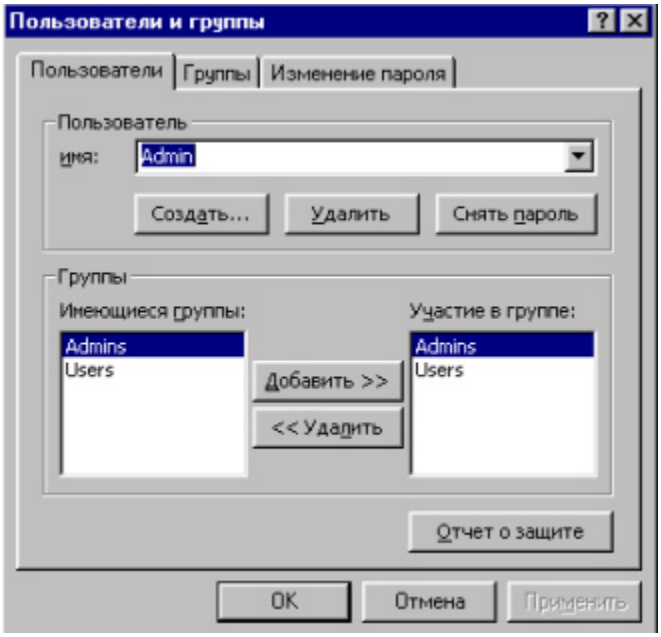


Рисунок 2.5 - Вікно властивостей користувачів і груп

Натисніть кнопку ***Создать…*** і введіть ім'я нового користувача, наприклад *user1,* вкажіть його код. За умовчанням запис увійде до групи *Users.* Повторіть ці дії для всіх користувачів, які працюватимуть з БД.

Перейдіть у вкладку ***Изменение пароля.*** Задайте пароль адміністратора, після чого при кожному запуску *Access* з'являтиметься вікно, що пропонує ввести ім'я користувача і пароль (рис. 2.6).

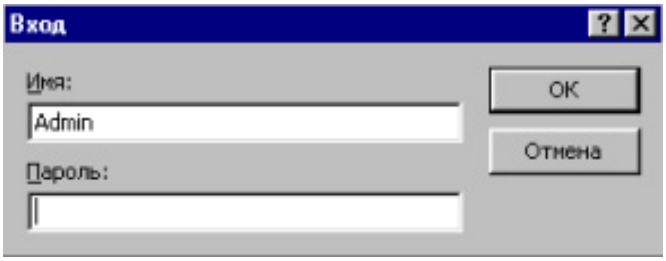


Рисунок 2.6 - Запит імені і пароля користувача

У пункті меню ***Сервис*** виберіть ***Защита/Разрешения*** (рис. 2.7). Виберете об'єкт, що захищається, наприклад Таблица1. Задайте дозволи для групи *Users,* а потім і для кожного з користувачів.

Тепер необхідно кожному користувачу самому ввести свій пароль. Для цього необхідно зайти в БД під своїм маєтком і виконати дії як при створенні пароля Адміністратора.

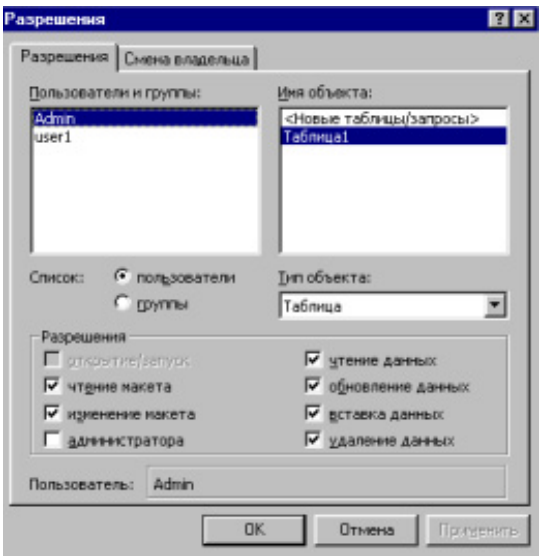


Рисунок 2.7 – Вікно визначення прав доступу для кожного користувача

5 Контрольні питання

1. Способи захисту інформації в БД Access.
2. Групи і користувачі БД Access . Файл робочої групи.

3. Об'єкти БД Access і права доступу до об'єктів. Поняття власника об'єкту.
4. Алгоритм захисту БД Access.
5. Користувачі і групи в Windows NT. Об'єкти Windows NT, що захищаються.
6. Принцип дії систем безпеки в Windows NT.

ЛАБОРАТОРНА РОБОТА №3

«Використання програмних засобів шифрування»

Мета роботи: освоїти принципи асиметричного шифрування і створення електронного цифрового підпису

1 Короткі теоретичні відомості

Криптосистеми з відкритим ключем (асиметричні криптосистеми) для шифрування і дешифровки даних використовують різні ключі. Один з ключів, званий відкритим, може бути переданий по відкритих каналах передачі даних або навіть може бути опублікований. Розшифровка даних проводитися за допомогою закритого ключа, який генерується і зберігається на стороні одержувача шифротексту у секреті. Ідея систем з відкритим ключем була запропонована Діффі і Хелманом.

Узагальнена схема асиметричної криптосистеми показана на рис. 3.1. При організації шифрованої передачі одержувач повинен згенерувати пару ключів (використовуючи деяку початкову умову – НУ, часто визначувану паролем доступу до ключів), відкритий ключ одержувача $К_{ОП}$ і закритий ключ одержувача $К_{ЗП}$, і переслати свій відкритий ключ відправнику. Відправник, використовуючи вибрану їм і одержувачем асиметричну криптосистему, виконають шифрування відкритого тексту М з використанням відкритого ключа $К_{ОП}$, після чого посилає шифротекст одержувачу. Одержувач виконує розшифровку, використовуючи свій закритий ключ $К_{ЗП}$.

Перехоплювач, володіючи відкритим ключем одержувача $К_{ОП}$ і шифротекстом $С=ЕК_{ОП}(М)$, при спробі розшифрувати шифрограму стикається з нерозв'язною обчислювальною проблемою, такий же як і при спробі обчислення закритого ключа по відомому відкритому. Таким чином, достоїнством асиметричних криптосистем є можливість пересилки відкритих ключів по незахищених каналах.

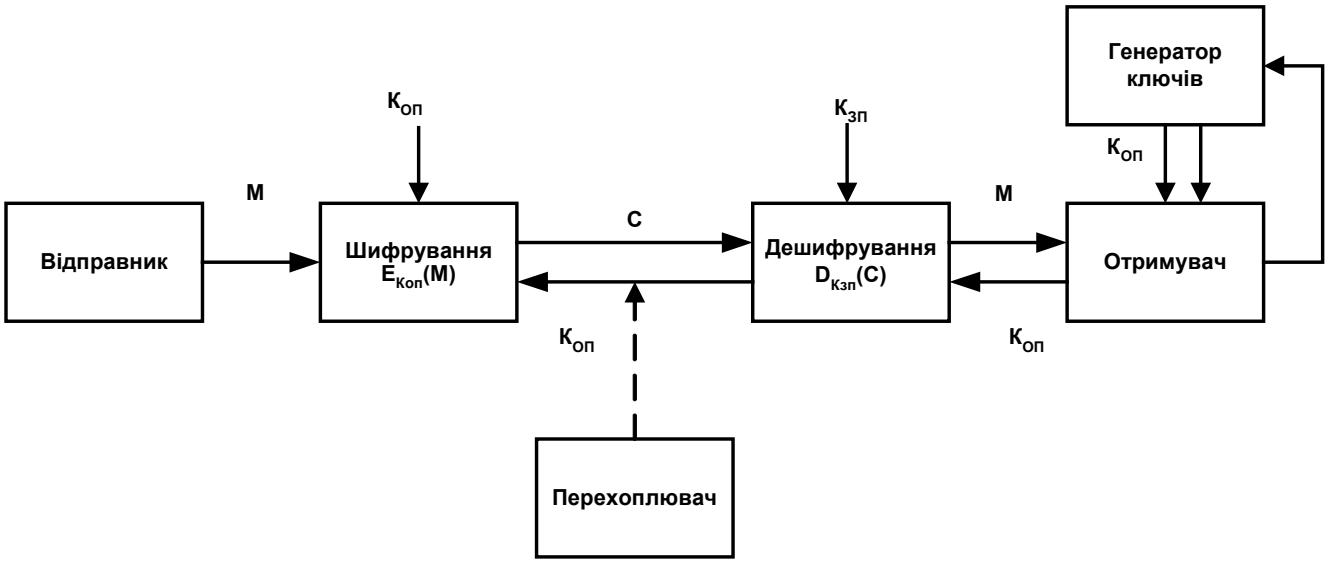
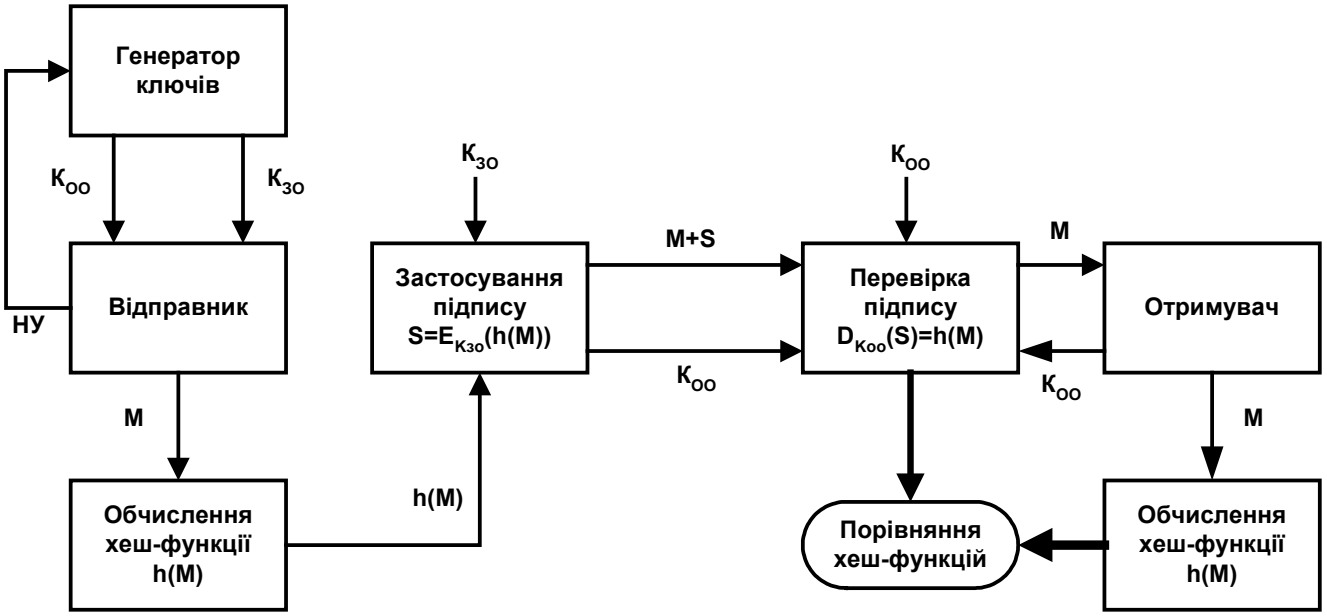


Рисунок 3.1 – Узагальнена схема асиметричної криптосистеми

Концепція асиметричних криптосистем заснована на застосуванні однонаправлених функцій. Однонаправленою називається функція y=f(x), обчислення якої по відомому x не складає складнощів, але обчислення x по відомому представляє серйозну обчислювальну проблему.

Проблема аутентифікації даних полягає в тому, що при прийомі конфіденційних (і інших) даних часто необхідно мати упевненість, що ці дані виходять від певної особи. Для паперових документів така упевненість задається, якщо вони завірені підписом і (або) печаткою. Проблема цілісності даних полягає в тому, що при отриманні даних необхідно мати упевненість, що вони не були навмисно спотворені під час передачі (випадкове спотворення ж цифрового підпису при передачі електронних даних по мережі практично виключається сучасними мережевими протоколами). Електронний цифровий підпис (ЕЦП) – це відносно невелика додаткова інформація, передавана разом з відкритим або зашифрованим документом, що засвідчує автора документа (тобто вирішальна проблему аутентифікації) і підтверджуюча відсутність спотворень початкового документа (тобто проблему цілісності). Система електронного підпису використовує асиметричні криптосис-

теми і включає дві процедури: процедуру накладення ЕЦП і процедуру перевірки ЕЦП. У процедурі накладення ЕЦП використовується секретний закритий ключ відправника $К_{ЗО,}$ а в процедурі перевірки – відкритий ключ відправника $К_{ОО,}$ використовуваний одержувачем. Відкритий ключ може бути пересланий по незахищеному каналу.



НУ – початкова умова

h(M) – хэш-функція від відкритого тексту M

S – електронний цифровий підпис

Рисунок 3.2 – Схема накладення і перевірки електронного цифрового підпису

При формуванні ЕЦП відправник, перш за все, обчислює так звану хэш-функцію підписуваного відкритого тексту M, що є відносно короткий блок даних, характеризуючий весь відкритий текст M в цілому. Якщо хоч би один символ відкритого тексту буде змінений або переставлений, то хэш-функція з дуже високою ймовірністю матиме інше значення.

Після обчислення хэш-функції h(M) відправник шифрує її своїм закритим ключем $К_{ЗО}$, додає до відкритого тексту М і пересилає їх одержувачу. Крім того, відкритий текст може бути додатково зашифрований за допомогою відкритого ключа одержувача КОП так, як це описано при поясненні асиметричних крипто-систем.

Одержавши підписаний ЕЦП текст, одержувач за допомогою відкритого ключа відправника відновлює хэш-функцію h(M), після чого він також виконує обчислення цієї хэш-функції від одержаного тексту М. У випадку якщо обидва значення хэш-функції співпадають, робиться висновок про те, що повідомлення дійсно послане відправником – володарем закритого ключа, пов'язаного з відкритим ключем, яким одержувач виконував розшифровку і що не порушена цілісність повідомлення. У випадку якщо значення хэш-функцій не співпадають, робиться висновок про порушення цілісності повідомлення і/або підміну відправника.

Принциповим моментом в системі ЕЦП є неможливість підробки ЕЦП без знання секретного ключа відправника.

Звичайно, окрім даних хэш-функції, кожна ЕЦП також включає:

- дату накладення ЕЦП;
- термін закінчення дії ключів ($К_{ОО}$ і $К_{ЗО}$ генеруються одночасно і мають однаковий термін дії) ЕЦП;
- інформацію про особу, що підписала файл (П.І.Б., посада, назва організації);
- ідентифікатор того, що підписав (ім'я відкритого ключа).

Однією з найбільш поширених програм шифрування даних є програма PGP (Pretty Good Privacy – досить добра секретність) автора Філа Зіммерманнна. Програма PGP використовує гібридну криптографічну систему. При цьому кодування і декодування виконується у декілька етапів:

1. Шифрування

- стиснення відкритих даних (plaintext), призначених для пересилки (що підвищує швидкість передачі і знижує ймовірність використання зламаних фрагментів тексту для декодування всього пакету), зашифровані дані неможливо піддати додатковому стисненню;
- створення ключа сесії (session key) – секретного одноразового ключа (secret key),который генерується програмою як похідна випадкових переміщень миші і даних, набраних на клавіатурі;
- шифрування даних за допомогою секретного ключа сесії (session key);
- шифрування самого ключа сесії (session key) за допомогою відкритого ключа (public key);
- передача зашифрованого тексту (ciphertext) і зашифрованого ключа сесії (session key) одержувачу.

2. Розшифровка

- одержувач використовує свій власний закритий ключ (private key) для декодування використаного відправником ключа сесії (session key);
- зашифрований текст (ciphertext) розкривається ключем сесії (session key);
- розпаковування даних, стислих при відправленні (plaintext).

Для того, щоб мати можливість обмінюватися секретними повідомленнями з тими, що оточують (що вже інсталювали на свої комп'ютери програму PGP) необхідно згенерувати пару ключів: відкритий і закритий. Ця пара буде використана надалі також і для створення цифрового підпису. Відкритим ключем можна ділитися з ким завгодно, в таємниці закритого ключа необхідно зберігати від всіх. Запустіть програму (*Пуск – Программы - PGP – PGPkeys),* вкажіть власне ім'я *(Full name)* і адресу електронної пошти *(Email address),* з якими будуть асоційовані створювані ключі.

Як тип ключа *(Key Pair Type)* виберіть *Diffie-Hellman/DSS* (ключ *RSA* більш застарілий і повільніший, проте, якщо серед респондентів є користувачі раніших версій, ніж PGP 5.0, доведеться використовувати ключ *RSA).*

Виберіть довжину відкритого ключа *(Key Pair Size):* за умовчанням, при використанні методу *Diffie-Hellman/DSS,* пропонується вибрати 2048 - розрядний ключ; погоджуйтеся з цим значенням. Також можна встановити крайній термін, до якого дані ключі можуть бути використані для кодування і підпису (здібність до розшифровки і перевірки достовірності зберігатиметься).

Введіть ключову фразу *(passphrase)* завдовжки не менше 8 символів, можна використовувати будь-які регістри, спеціальні символи, пропуски (допускається запис фрази на будь-якій мові) і ще раз підтвердите її введення в нижньому вікні (рис.3.3).



Рисунок 3.3 - Вікно введення ключової фрази

Програма оцінить криптостойкость фрази *(Passphrase Quality).* Після чергового натиснення на кнопку *Далі (Next)* доведеться трохи почекати, процедура генерації ключів може зайняти декілька хвилин; цей час рекомендується присвятити безладному

натисненню на клавіші і кнопки, тим самим ви полегшите програмі дуже трудомісткий процес генерації випадкових даних.

Коли ключі згенерували (рис 3.4), користувачу програми пропонується вийти в Інтернет і відправити відкритий ключ на сервер, де вже складені ключі інших працівників компанії (root server).

Програма PGP пропонує зробити резервну копію відкритого і закритого ключів (файли pubring.pkr і secring.scr, відповідно) на змінному носії. Сукупний обсяг "брелків" складає приблизно 5 Кбайт в пропорції 3:1 на користь відкритого ключа.

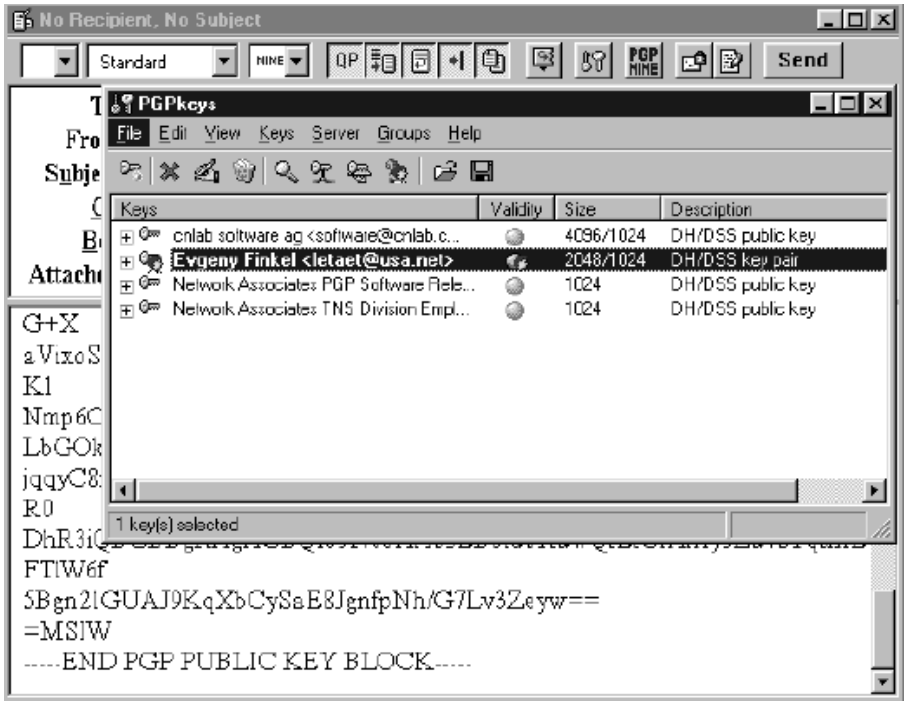


Рисунок 3.4 – Згенеровані ключі і відправка відкритого ключа

Відправте свій відкритий ключ всім потенційним одержувачам вашої респонденції. Для цього необхідно позначити рядок з вашим ключем у вікні PGPkeys і перетягнути (притиснувши ліву клавішу миші) рядок з ключем у вікно текстового повідомлення вашої поштової програми, потім розіслати цей лист по адресатах. Існує також можливість копіювання символів відкритого ключа в текстовий файл і пересилки його адресатам. Відкритий ключ починається з фрази -BEGIN PGP PUBLIC KEY

BLOCK-, після якої вказується версія програми, і завершується фразою -END PGP PUBLIC KEY BLOCK-.

Простим способом використання програми є вибір команд з контекстного меню після виділення файлу, який необхідно зашифрувати і/або підписати (рис. 3.5 – 3.7).

Існує також можливість розшифрування повідомлення шляхом копіювання його, починаючи з символів -BEGIN PGP MESSAGE- і закінчуючи символами -END PGP MESSAGE- в Буфер обміну і розшифровкою його.

Для додавання відкритого ключа іншого користувача в програму, одержаного, наприклад, по електронній пошті виділите текстовий блок від -BEGIN PGP PUBLIC KEY BLOCK- до -END PGP PUBLIC KEY BLOCK-, занесіть його в буфер обміну і вставте його у вікні програми (або клацніть на піктограмі PGP  в правій частині панелі завдань, виберіть команду *Add Key from Clipboard* і натисніть на *Import).* З цієї миті відкритий ключ респондента знаходиться у Вас і одержавши від нього закодоване повідомлення або файл, Ви можете розшифрувати його через команди контекстного меню *(Decrypt & Verify).*

Накладення і перевірка електронного підпису відбувається аналогічно шифруванню, тільки в меню необхідно вибрати пункти *Sign Clipboard,* щоб підписати повідомлення в буфері (або *Encrypt & Sign Clipboard,* щоб зашифрувати, а потім підписати), і *Decrypt & Verity Clipboard* - для розшифровки і перевірки достовірності підпису. Сам по собі підпис повідомлення не шифрує, а тільки підтверджує його достовірність. Підписана фраза "Мишка очень любит мед" виглядатиме:

—BEGIN PGP SIGNED MESSAGE-
Hash: SHA1
Мишка очень любит мед
—BEGIN PGP SIGNATURE - Version:
PGPfreeware 6.0.2i
J1Ne9hjjfejirlfFYG56F678njklThiKIHF=GUJh
-END PGP SIGNATURE-



Рисунок 3.5 - Контекстне меню шифрування і/або підпису файлу

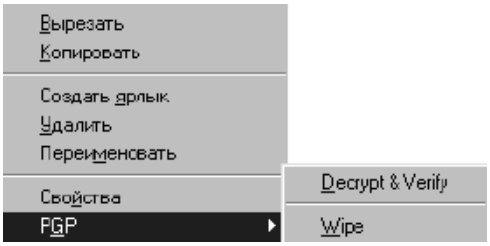


Рисунок 3.6 - Контекстне меню розшифрування і перевірки підпису файлу

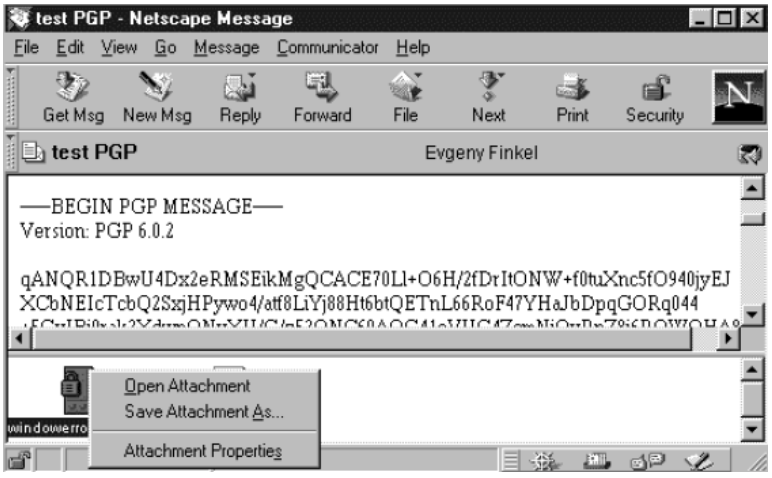


Рисунок 3.7 - Зашифроване повідомлення

При перевірці достовірності послання, що прийшло, у вікні текстового повідомлення буде виведений текст, а у вікні PGPLog - результат перевірки достовірності.

2 Порядок виконання роботи

1. Створіть пару ключів в програмі PGP.

2. Відправте відкритий ключ своєму респонденту або по електронної пошти або шляхом обміну через загальні ресурси локальної мережі. Занесіть в звіт по роботі Ваш відкритий ключ.
3. Додайте закритий ключ Вашого респондента в свою програму.
4. Зашифруйте файл (що складається з 1-2 коротких пропозицій) відкритим ключем Вашого респондента і перешліть йому зашифрований файл.
5. Отримайте аналогічний зашифрований файл від Вашого респондента і розшифруйте його. Занесіть в звіт отриманий криптотекст і розшифрований текст.
6. Накладіть свій електронний підпис на файл і відправте його Вашому респонденту. Занесіть в звіт свій електронний підпис.
7. Отримайте від Вашого респондента файл, підписаний ним, і перевірте підпис. Приведіть в звіті електронний підпис Вашого респондента.
8. Зашифруйте файл відкритим ключем Вашого респондента, накладіть на нього свій електронний підпис і перешліть Вашому респонденту зашифрований файл.
9. Отримайте підписаний зашифрований файл від Вашого респондента, розшифруйте його і перевірте підпис.

3 Контрольні питання

1. Опишіть алгоритм роботи асиметричних криптосистем
2. Які функції називаються однонаправленими. Приведіть відомі Вам однонаправлені функції і асиметричні алгоритми шифрування, їх що використовують.
3. Опишіть алгоритм роботи електронного цифрового підпису.
4. Що називають хэш-функцією, приведіть відомі Вам алгоритми обчислення хэш-функцій?
5. Опишіть етапи шифрування - дешифровки даних програмою PGP.

ЛАБОРАТОРНА РОБОТА №4

«Робота з антивірусними програмами різних типів»

Мета роботи: вивчити способи і засоби захисту від комп'ютерних вірусів

1 Короткі теоретичні відомості

Вважається, що термін "комп'ютерний вірус" вперше застосував співробітник Лехаїського університету (США) Ф. Коен на конференції з безпеки інформації в 1984 р. Комп'ютерні віруси також володіють здібністю до самовідтворення з наступним впровадженням копій вірусу у файли, системні області комп'ютера або навіть на інший комп'ютер по мережі. При цьому дублікати також зберігають здібність до подальшого розповсюдження. Як правило, віруси володіють якою-небудь деструктивною дією, хоча це і не є обов'язковою умовою. Здібність до самовідтворення треба розуміти дуже широко. Різні примірники одного вірусу не тільки не зобов'язані повністю збігатися, але можуть, навіть не мати жодного загального байта (йдеться про так звані складнополіморфні віруси).

1.1 Класифікація вірусів

1.1.1 Класифікація по меті дії

- Завантажувальні (бутові) віруси - віруси, що заражають завантажувальні сектори дискет і жорстких дисків. Існують бутові віруси, завантажувальні сектори дискет, інші заражають - ще і завантажувальні сектори вінчестерів, деякі - заражають головний завантажувальний сектор вінчестера (MBR). Часто віруси "всеїдні" і заражають і те, і інше.
- Файлові віруси - заражають виконувані файли. Крім того, до файлових відносяться так звані macro-віруси - файлові віруси, що заражають файли деяких систем документообігу, наприклад, MS Office, AmiPro ін. Такі системи мають вбудовані макро - мови (VBA, VB). Ці мови володіють достатніми можливостями, щоб

проводити практично всі операції, необхідні вірусу. Для розмноження в середовищі MS Office такі віруси використовують файл шаблону за умовчанням normal.dot.

- Віруси, що вражають код BIOS комп'ютера.
- Скрипт - віруси створюються на мові сценаріїв VBS, розробленій Microsoft. Така мова виконується інтерпретатором Windows, крім того, веб - браузером MS Internet Explorer. Подібні віруси можуть видаляти файли на жорсткому диску, змінювати настройки системи, запускати троянських коней і ін. Якщо у веб - браузері неправильно настроєний розділ «Безопасность», то загроза інфікування такими вірусами існує при відвідуваннях веб - сторінки, що містять шкідливий VBS - скрипт.
- Мережеві черв'яки - це шкідницькі комп'ютерні програми, які можуть створювати свої копії на одному і тому ж комп'ютері чи ж копіювати себе на інші комп'ютери мережі, використовуючи найчастіше систему електронної пошти. Розсилка черв'яків за адресами адресної книги поштових клієнтів призводить до вибухоподібного зростання повідомлень, що пересилаються, що може привести до перевантаження каналів і відмови в обслуговуванні.
- Троянські коні - це програми, які маскуються під які-небудь корисні застосування (наприклад, утиліти або навіть антивірусні програми), але при цьому завдають різні руйнівні дії. Трояни не вбудовуються в інші файли і не володіють здібністю до власнодублювання. В порівнянні з іншими типами вірусів «троянські коні» мало поширені, оскільки після запуску вони або знищують себе разом з рештою даних на диску, або знищуються самим постраждалим користувачем.
- Backdoor-програми - утиліти прихованого адміністрування аналогічні троянським коням, але на відміну від останніх, самі не виконують активних дій, а тільки відчиняють доступ на Ваш комп'ютер зловмиснику.

1.1.2 Класифікація по властивостях вірусів

- Віруси, що маскуються (Stealth). Видів маскування безліч, але всі вони засновані на перехопленні вірусами переривань BIOS і операційної системи. Перехопивши переривання, віруси контролюють доступ до заражених об'єктів. Наприклад, при прогляданні зараженого об'єкту, вони можуть "підсунути" замість нього здоровий. Крім того, віруси спотворюють інформацію BIOS (наприклад, повертають невірне значення довжини файлу, приховуючи свою присутність в ньому). Для більшості антивірусних програм віруси, що використовують стелс - технологію, є серйозною проблемою. Винятком є програми-ревізори дисків (наприклад, AdInf).
- Поліморфні віруси. Аналіз вірусів полягає у виділенні в них сигнатур (послідовності байт, специфічної (у ідеалі - унікальної) для конкретної програми) і подальшому їх пошуку в потенційних об'єктах вірусної атаки. Поліморфні віруси шифрують свій код. Мета такого шифрування достатньо очевидна: маючи заражений і оригінальний файли, і, отже, маючи можливість виділити вірус "в чистому вигляді", не представляється можливим проаналізувати його код за допомогою звичайного дизасемблювання. Розшифровка проводиться самим вірусом вже безпосередньо під час виконання. При цьому можливі самі різні варіанти: вірус може розшифрувати себе всього відразу, а може виконувати таку розшифровку по частинах, може знов шифрувати вже відпрацьовані ділянки і т.д. (причому іноді не тим способом, яким вони були зашифровані раніше). Проте зашифровані віруси обов'язково містять деякий незашифрований фрагмент - розшифровщик (або його частка). По ньому можна побудувати сигнатуру такого вірусу і далі вже боротися з ним звичайними способами. Ситуація змінилася, коли були розроблені алгоритми, що дозволяють не тільки шифрувати код вірусу, але і змінювати розшифровщики - кожна нова копія вірусу містить новий розшифровщик, який може в кожному біті відрізнятися

від розшифровщика, що створив похідні копії. Навіть з простих комбінаторних міркувань ясно, що розшифровщиків, довжина яких звичайно обмежена, не може бути нескінченно багато. Але якщо їх всього лише два-три трильйона, то ясно, що задача перебору всіх можливих не стоїть. Віруси, що використовують описану технологію, одержали назву поліморфних.

- Резидентні віруси відрізняються від нерезидентних тим, що після запуску інфікованої програми вони залишаються в оперативній пам'яті комп'ютера.

1.2 Типи антивірусних засобів:

- Сканери - поліфаги. Пошук вірусів в простому випадку зводиться до пошуку їх сигнатур. Після виявлення вірусу в тілі програми поліфаг знешкоджує його. Для цього розробники антивірусних засобів ретельно вивчають роботу кожного конкретного вірусу: що він псує, як він псує, де він ховає те, що зіпсує і т.д. В більшості випадків поліфаг здатний благополучно видалити вірус і відновити працездатність зіпсованих програм. Але необхідно добре розуміти, що це можливо не завжди. Сучасні антивірусні програми містять засоби евристичного аналізу, що дозволяють розпізнавати деструктивний код в нових, ще не відомих поліфагу вірусах.
- Сторожа - невеликі резидентні програми, перевіряючі завантажувані в пам'ять файли на предмет знаходження в них сигнатур вірусів. Достоїнство - швидке виявлення вірусів. Недолік - сильне завантаження систем і зниження їх продуктивності.
- Ревізори - програми, у функції яких входить антивірусна перевірка шляхом контролю властивостей файлів (розміру, дати і часу створення / зміни, контрольної суми і ін.). При цьому виконується порівняння властивостей файлів з їх первинними значеннями (дуже важливо, щоб вони не належали вже

зараженим файлам). Внаслідок перевірки користувачу виводиться список файлів із зміненими властивостями, і користувач вирішує, які з файлів змінив він сам, а які можливо заражені. Часто ревізори працюють в тандемі зі сканерами - поліфагами і автоматично передають їм на перевірку файли із зміненими властивостями.

- Антивірусні вакцини, які використовувалися для обробки файлів і завантажувальних секторів. Вакцини бувають пасивними (приклад такої вакцини описаний нижче) і активними. Активна вакцина, "заражаючи" файл, подібно до вірусу, оберігає його від будь-якої зміни і у ряді випадків здатна не тільки виявити сам факт зараження, але і вилікувати файл. Пасивні вакцини використовувалися (тепер це вже велика рідкість) для запобігання зараження файлів деякими вірусами, що використовують прості ознаки їх зараженості - "дивні" час або дата створення, певні символьні рядки і ін. Нині вакцинація широко не застосовується.

Крім того, існують спеціальні додаткові пристрої (звичайно електронна плата із спеціалізованим процесором), що забезпечують досить надійний захист. Перевагою плати є незалежність від операційної системи і BIOS (які можуть бути заражені), а також практично відсутність використання ресурсів комп'ютера. Прикладом апаратних засобів антивірусного захисту може служити плата Sheriff.

Існує таке поняття як вірусна містифікація (Virus hoax) - помилкове попередження про нібито грізному вірусі, який насправді не існує. Наприклад, поступило по e-mail попередження про загрозу небезпечного вірусу, який записує своє тіло в один з файлів в каталозі Windows. В результаті багато користувачів помилково видаляли цей необхідний для функціонування операційної системи файл.

1.3 Робота зі сканером - поліфагом DrWeb

Пакет антивірусних програм DrWeb містить сканер - поліфаг DrWeb, резидентний сторож Spider Guard, поштовий антивірусний фільтр Spider Mail, що працює з програмами електронної пошти і планувальник, що дозволяє настроювати автоматичну перевірку файлів і завантаження нових вірусних баз за розкладом.

Запуск програми здійснюється стандартним чином через головне меню або з контекстного меню (у останньому випадку будуть перевірені виділені об'єкти). Після запуску виконується сканування програм, що знаходяться в пам'яті і їх файлів, розташованих на жорсткому диску комп'ютера.

Після закінчення перевірки користувач може вказати у верхньому вікні диск (диски) або папку (папки)), які необхідно перевірити, і натиснути на кнопку запуску перевірки (рис. 4.1).

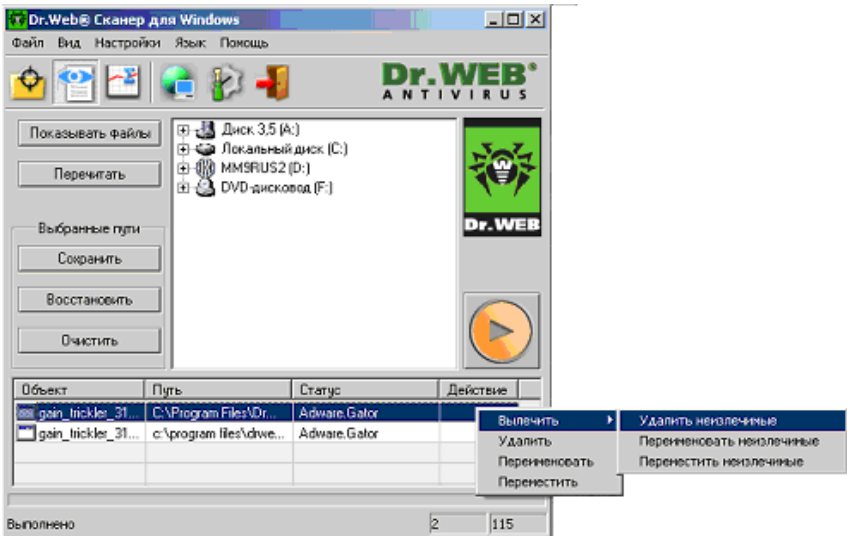


Рисунок 4.1 - Інтерфейс сканера DrWeb

У випадку якщо при скануванні виявлено вірус, дані зараженого файлу з'являться в нижньому вікні. При клацанні на цьому файлі правою кнопкою миші, з'явиться контекстне меню, з якого можна вибрати команди, приведені на рис. 3.1. Окрім команд лікування і видалення заражених файлів доступні команди пере-

йменування файлів (звичайно міняється їх розширення, що не дозволяє запускати або використовувати цей файл програмами) і переміщення заражених файлів в спеціальну папку - карантин.

Програма характеризується гнучкими засобами настройки, що дозволяють встановлювати різні параметри перевірки і настроювати автоматичне оновлення баз. Ви можете запускати програму Dr.Web сканер для Windows в режимі командного рядка. Такий спосіб дозволяє задати настройки поточного сеансу сканування і перелік сканованих об'єктів як параметри виклику. Саме в такому режимі можливий автоматичний виклик сканера за розкладом. Синтаксис команди запуску наступний:

[путь_к_программе] drweb32w [объекты] [ключи]

Замість програми Dr.Web сканер для Windows може використовуватися Dr.Web консольний сканер для Windows. В цьому випадку замість *drweb32w* необхідно набрати ім'я команди *drwebwcl.*

Список об'єктів сканування може бути порожній або містити декілька елементів, розділених пропусками. Найбільш поширені варіанти завдання об'єктів сканування описані нижченаведеними прикладами:

- - сканувати всі жорсткі диски;
- C: - сканувати диск C:;
- D:\games - сканувати файли в каталозі;
- D:\games*.exe - сканувати у вказаному каталозі всі файли з розширенням exe.

Параметри - ключі командного рядка задають настройки програми. При їх відсутності сканування виконується з раніше збереженими настройками (або настройками за умовчанням, якщо ви не міняли їх). Кожен параметр цього типа починається з символу /, ключі розділяються пропусками. Нижче приведено декілька найбільш часто використовуваних ключів:

/ci - лікувати інфіковані об'єкти (без запиту).

/icm - переміщати невиліковні файли (у каталог за умовчанням)

/icr - перейменовувати (за умовчанням)

/qu - закрити вікно сканера після закінчення сеансу

/go - не видавати жодних запитів.

Останні два параметри особливо корисні при автоматичному запуску сканера (наприклад, за розкладом).

1.4 Робота з ревізором AdInf

При першому запуску програми AdInf пропонується виконати сканування дисків з метою побудови таблиць з параметрами контрольованих файлів. Файли вибираються в настройках програми по типах, крім того, там же вказується алгоритм перевірки і контрольовані зміни (рис.4.2). Програма контролює файли по змінній довжині і контрольній сумі.

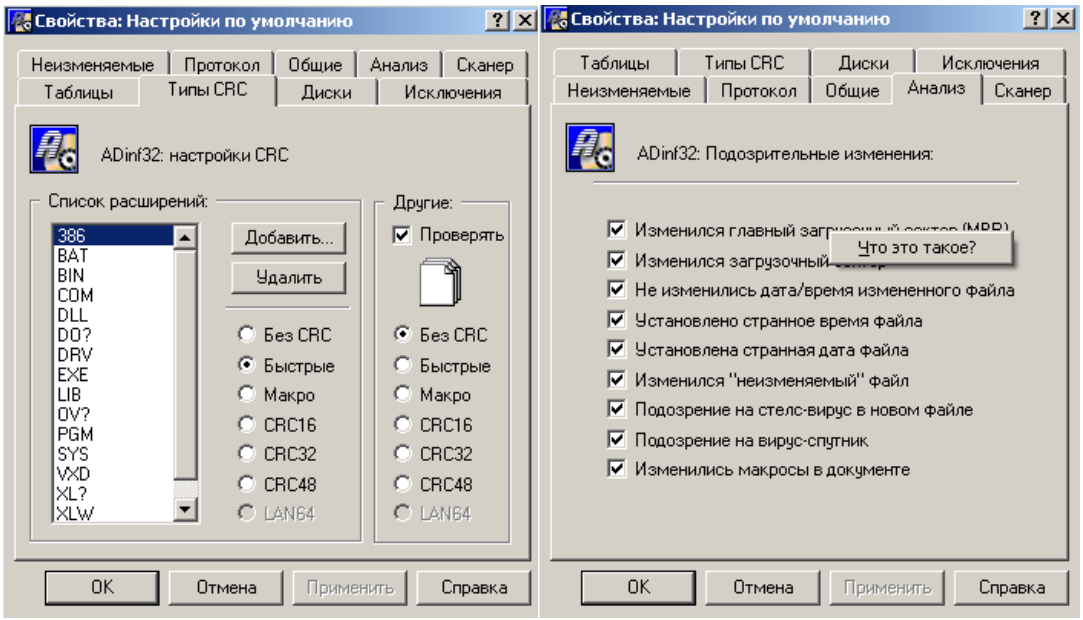


Рисунок 4.2 - Настройки ревізора AdInf

Програмою підтримуються наступні типи контрольних сум:

- Швидкі - використовують знання структури COM і EXE файлів MS-DOS і використовуються для їх перевірки (перевіряється лише та частина, яка обов'язково повинна змінитися при зараженні);
- Швидкі (Win32) - використовують знання структури COM і EXE файлів MS-DOS і MS Windows, а також VxD файлів (перевіряється лише та частина, яка обов'язково повинна змінитися при зараженні);
- Макро - використовують знання структури файлів MS Word і MS Excel - контролюють зараження макро-вірусами. Якщо для фалів типів DOC, DOT, XLS, XLT використовується цей алгоритм, то програма фіксуватиме тільки додавання/зміну макросів в документі, позбавляючи від повідомлень про зміни файлу;
- CRC16 - розраховується CRC16 контрольна сума для всього файлу;
- CRC32 - розраховується CRC32 контрольна сума для всього файлу;
- CRC48 - реалізується одночасний розрахунок контрольних сум CRC16 і CRC32 для всього файлу;
- LAN64 - реалізують алгоритм, розроблений російською фірмою ПАН Крипто з 64-бітовою хэш-функцією, що має гарантовано високу надійність виявлення як випадкових, так і навмисних змін файлів (цей тип доступний лише у версії програми AdInf Pro).

Побудова таблиць також може бути ініційована вручну для окремих дисків (рис.4.3).

Перевірка здійснюється порівнянням властивостей файлів із збереженими в таблиці і користувачу вказуються зміни і доповнення (рис. 4.4).

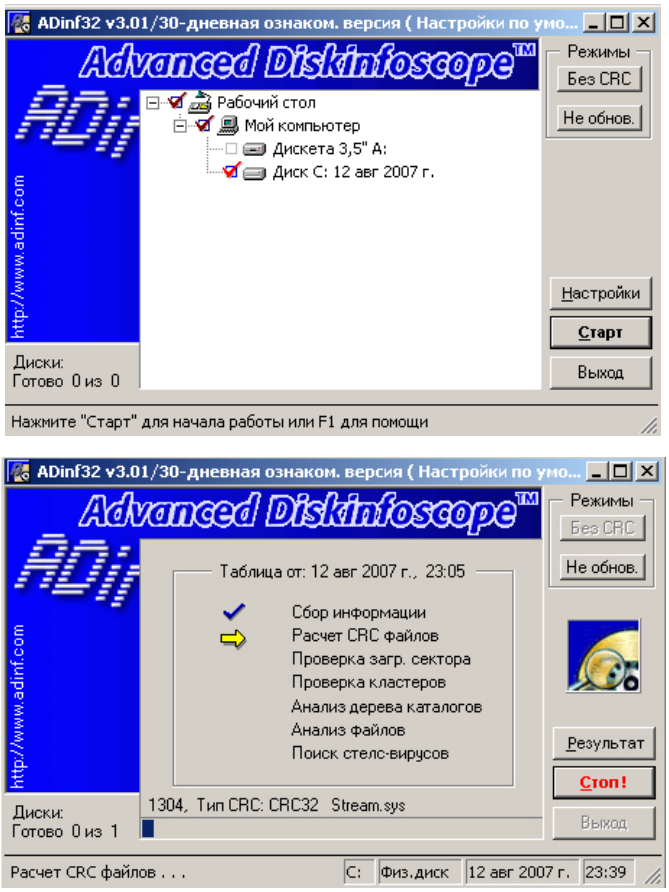


Рисунок 4.3 - Побудова таблиць для дисків

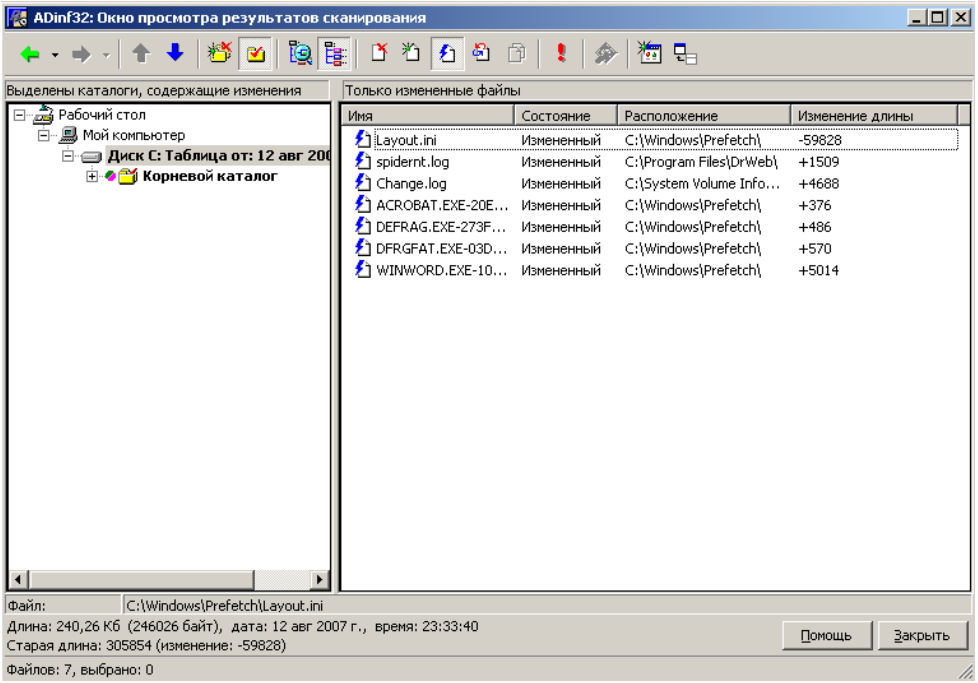


Рисунок 4.4 - Вікно із результатами сканування і даними про зміну файлів

Крім того, можливо виконувати контроль нижньої оперативної пам'яті (рис.4.5), доступній операційній системі, зміну головного завантажувального сектора (рис.4.5) і зміну завантажувального сектора логічного диска (рис.4.6).

Існує можливість настройки програми-ревізора на автоматичну взаємодію із сканером з метою перевірки підозрілих файлів.

2 Порядок виконання роботи

1. Ознайомтеся з можливостями настройки програми сканера DrWeb.

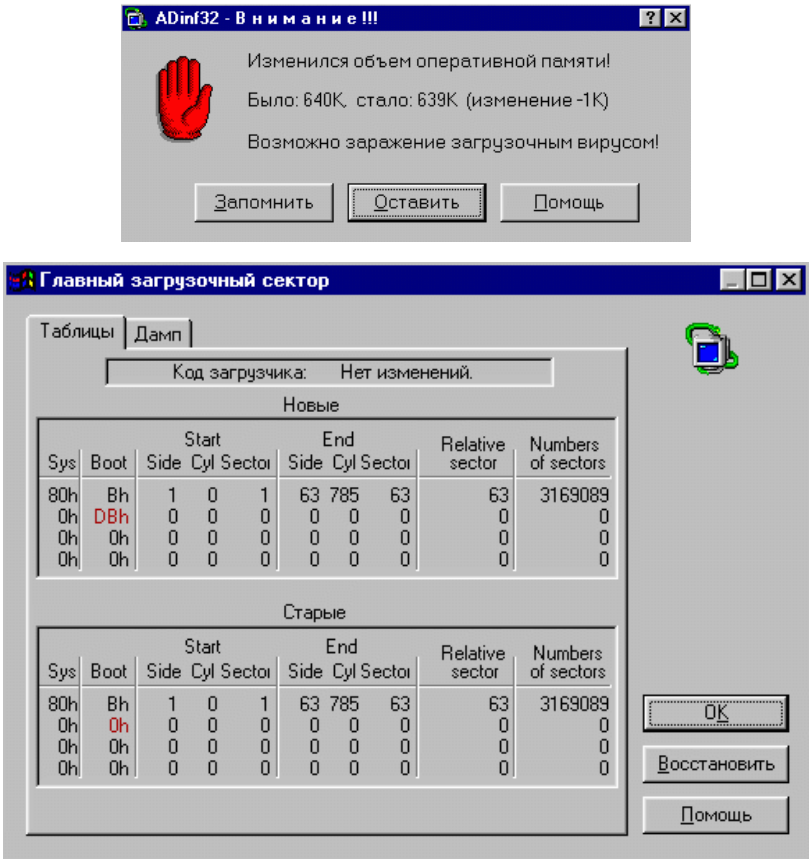


Рисунок 4.5 - Контроль нижньої оперативної пам'яті і головного завантажувального сектора програмою AdInf

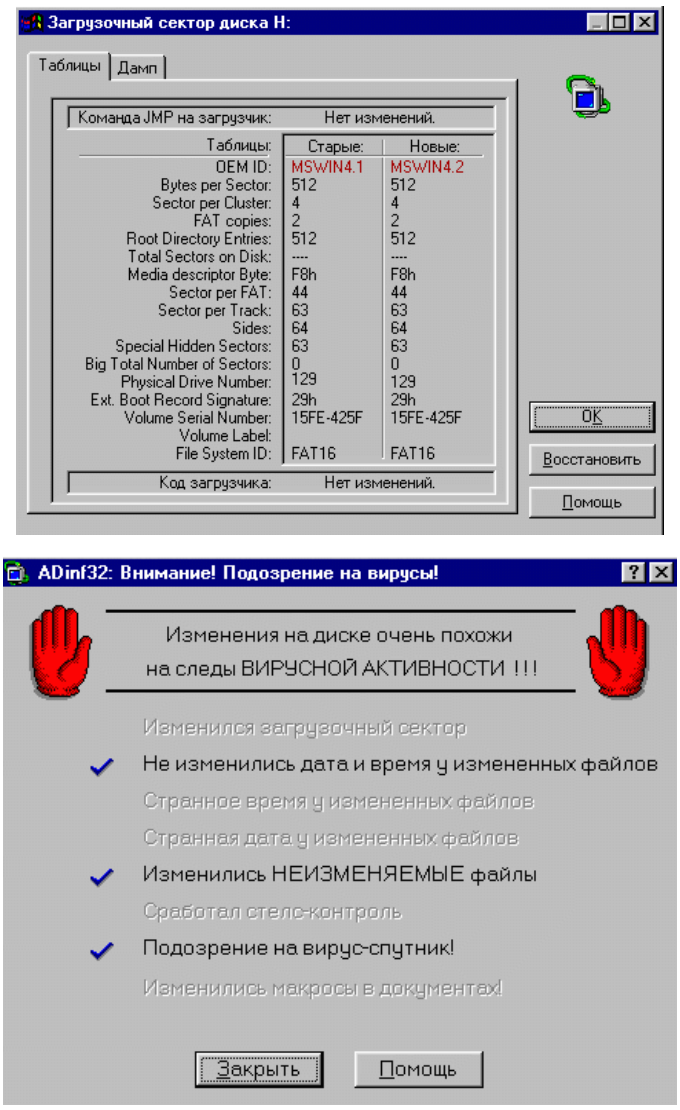


Рисунок 4.6 - Контроль завантажувального сектора логічного диска і повідомлення про підозру на вірус

2. Напишіть скрипт перевірки з евристичним аналізом файлів, архівів і контейнерів (по внутрішньому формату виконуваних модулів), що знаходяться в папці Мои документы і її підпапках з переміщенням невиліковних файлів в карантин і без інтерактивної взаємодії з користувачем, а також з перевіркою ОЗП, завантажувальних областей дисків і MBR.
3. Виконайте настройку завдання, що здійснює щоденний запуск скрипта в 12^{30}.
4. Приведіть лістинг скрипта і знімок екрану завдання.

5. Побудуйте таблиці дисків в програмі Adinf.
6. Створіть документ MS Word на диску і виконайте перевірку диска.
7. Створіть макрос в документі і виконайте перевірку ще раз.
8. Приведіть знімок вікна програми з результатами перевірки.

3 Контрольні питання

1. Дайте визначення і назвіть основні ознаки комп'ютерних вірусів.
2. Виконайте класифікацію вірусів по меті дії і їх властивостям.
3. Назвіть відомі Вам типи антивірусних засобів і принципи їх роботи.
4. Назвіть типи контрольних сум, підтримувані ревізором AdInf.
5. Назвіть основні ключі, використовувані з консольною версією сканера DrWeb.

ЛАБОРАТОРНА РОБОТА №5

«Робота з програмою резервного копіювання даних»

Мета роботи: вивчити методи резервного копіювання і архівації даних і навчитися працювати з програмою резервного копіювання.

1 Короткі теоретичні відомості

Резервне копіювання - це процес виготовлення копій файлів, що знаходяться на жорсткому диску комп'ютера, з метою забезпечення можливості відновлення початкових даних у випадку, якщо оригінал буде стертий, перезаписаний, пошкоджений, або знищений. Архівація - це процес відносно більш довготривалого (або потрібного згідно із законом) зберігання важливої інформації, звичайно у вигляді додаткової копії, часто в стислому вигляді. Різниця між резервним копіюванням і архівацією даних досить принципова - резервне копіювання застосовують для зберігання інформації на зовнішніх носіях із тим, щоб її можна було відновити (англ., еквівалент - restore або recovery) при аваріях або збоях, а також в деяких випадках для перенесення системних даних з машини на машину. При резервному копіюванні метою є збереження саме поточного стану системи. Попередні стани при цьому не завжди потрібно зберігати, хоча у ряді випадків це все-таки робиться. Архівація (archive) призначена для забезпечення довгострокового збереження напрацьованої інформації, причому так, щоб дані можна було відновити, навіть якщо вони створені декілька місяців або років тому. Наприклад, архівації, як правило, піддаються дані бухгалтерських і складських програм, різні фінансові документи, вже закінчені проекти. Завдання резервного копіювання і архівації в цілому аналогічні і на практиці часто ці поняття вважають синонімами. У сучасній технології роботи з копіями даних з'явилася нова технологія - клонування - створення файлу-образу жорсткого диска (або його розділу). Клонування є окремим випадком резервного копіювання.

Резервне копіювання і архівація здійснюються відповідно до трьох основних програмних методів запису на зовнішні носії: повним, інкрементальним і диференціальним.

При повному (звичайному) методі кожного разу проводиться копіювання всього обсягу вказаної інформації, наприклад, переноситься цілком файлова система, файли бази даних або окремий каталог на диску. Кожний з файлів позначається як архівний. Даний метод займає багато часу при записі і веде до великої витрати інформаційних носіїв. Однак в цьому випадку відновлення інформації здійснюється швидше, ніж при будь-якому іншому методі (для цього буде потрібно лише один образ (один набір носіїв)). Повне копіювання є найбільш простим рішенням при резервному копіюванні системної інформації і служить відправною крапкою для інших методик.

Інкрементальний (додатковий) метод є поетапним способом запису інформації. При такому методі перший запис на зовнішній носій є повною копією. При кожній подальшому запису на нього поміщаються лише модифіковані в порівнянні з попереднім копіюванням файли (тобто ті, у яких змінилися властивості або права доступу). Після закінчення заданого адміністратором часу цикл повторюється, а значить, знову робиться повна копія, а потім додаються інкрементальні. При цьому файли спеціальним атрибутом позначаються як архівні. З погляду швидкості і економії носіїв, даний метод є найшвидшим і веде до мінімальної витрати його простору. Проте відновлення інформації займає багато часу, оскільки спочатку інформацію потрібно відновити з повної копії, а потім по порядку зі всіх наступних.

При диференціальному (різницевому) методі перший запис на цифровий носій також є повною копією. На наступних етапах копіюються лише змінені з часу проведення першого (повного) копіювання файли. При цьому файли не позначаються як архівні, що вказує, що файли не заархівували. Після закінчення циклу вся процедура знову починається з повної копії. Диференціальний метод резервування потребує

більше часу, ніж інкрементальний, та зате при відновленні інформації вимагає тільки дві копії: повну і останню диференціальну.

Програма архівації даних операційної системи Windows окрім цих трьох методів пропонує також:

- Метод копій - використовується, коли потрібно виконати архівацію окремих файлів в проміжку між сеансами повної (звичайної) і інкрементальної (додаткової) архівації, оскільки при цьому не скасовуються результати попередніх завдань архівації. При цьому файли не позначаються як архівні.
- Щоденний метод - використовується для архівації файлів, які були змінені в день виконання архівації по інших методиках.

Головною проблемою інкрементального і диференціального копіювання є проблема вибору надійного критерію для встановлення факту модифікації файлу. Звичайно у якості цього застосовується атрибут Archive (для систем DOS/Windows), час створення/зміни файлів, його розмір або контрольна сума вмісту файлу (рис. 5.1). Але всі вони мають ті або інші недоліки, пов'язані з особливостями обробки атрибутів і прав доступу окремими прикладними програмами. Тому найнадійнішим (і при цьому найекономічнішим по витраті носіїв) є метод повного резервного копіювання.

Рекомендовані схеми ротації носіїв надають можливість використання декількох комплектів носіїв і забезпечують "глибину" версій файлів, дозволяючи відновлювати файл виходячи з його стану в певний момент часу. Однією з таких популярних схем є Grandfather-Father-Son (Дід – батько - син) - GFS. "Син" - це інкрементальне або диференціальне щоденне резервування, "Батько" - повне щотижневе резервування, а "Дід" - повне щомісячне резервування. Для цієї базової схеми ротації необхідно 12 комплектів носіїв (за умови 5 робочих днів в тиждень): чотири щоденних (понеділок-четвер); п'ять щотижневих, що виконуються в кожну п'ятницю 1-5; і три щомісячних, місяці 1-3). Носії використовуються в день, тиждень

або місяць, відповідні їх етикетці. Така схема забезпечує глибину історії до 4-х місяців.

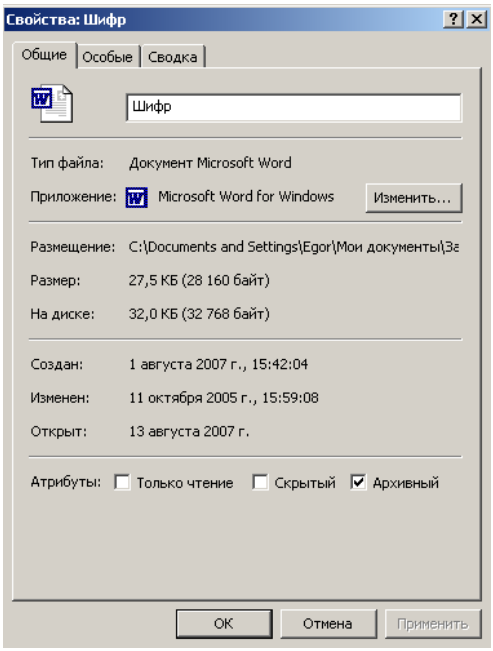


Рисунок 5.1 - Властивості файлу зі встановленим програмою архівації атрибутом "архивный"

Як програмне забезпечення для резервного копіювання і архівації в Windows використовується програма Backup для Windows (рис. 5.2 – 5.3), розроблена VERITAS Software Corp. Програма надає наступні можливості:

- Архівація вибраних файлів і папок на жорсткому диску (включаючи архівацію за розкладом).
- Відновлення файлів, що архівуються, і папок на локальний жорсткий диск або будь-який інший доступний диск.
- Використання засобу аварійного відновлення системи для збереження і відновлення всіх системних файлів і параметрів конфігурації, необхідних для відновлення системи після збою.
- Створення копії даних з будь-якого зовнішнього сховища або даних, що зберігаються на приєднаних дисках.

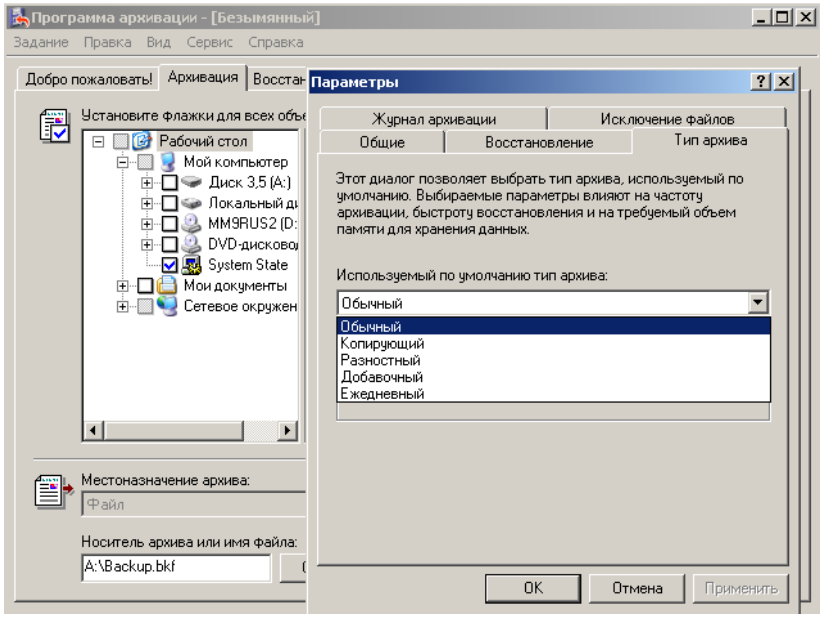


Рисунок 5.2 - Настройка методу архівації в програмі архівації даних Windows

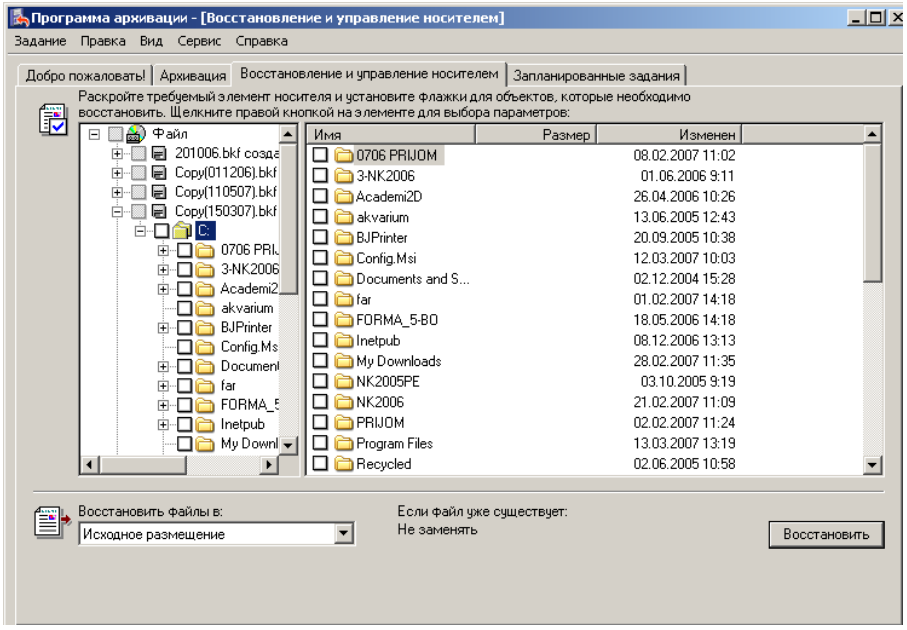


Рисунок 5.3 - Відновлення даних з архіву

- Створення копії даних стану системи локального комп'ютера, що включають реєстр, служби компонентів, базу даних Active Directory і базу даних служб сертифікації.

- Створення копії системного розділу, завантажувального розділу і файлів, необхідних для завантаження системи у разі збою комп'ютера або мережі.
- Планування періодичного виконання архівації для отримання поточних версій архівів.

Програма архівації дозволяє архівувати і відновлювати дані на томах з файловими системами FAT і NTFS. Програма надає користувачу зручний інтерфейс, що дозволяє виконувати архівацію, відновлення і управління носіями.

2 Порядок виконання роботи

2.1 Звичайний (повний) метод архівації

1. Створіть папку з ім'ям 1 і в ній текстовий файл backup 1.txt.
2. З'ясуйте, чи встановлений атрибут "архивный" у властивостях файлу і занесіть цю інформацію в табл. 5.1.
3. Виконайте звичайну (повне) архівацію папки 1 у файл архіву з ім'ям standart.bkf (розширення додається автоматично) у знов створеній папці з ім'ям 2.
4. З'ясуйте, чи встановлений атрибут "архивный" у властивостях початкового файлу, а також кількість файлів в архіві і занесіть цю інформацію в табл. 5.1.
5. Видаліть початковий файл в папці 1 і виконаєте відновлення файлу з архіву в початкове місце розташування - папку з ім'ям 1.
6. З'ясуйте, чи встановлений атрибут "архивный" у властивостях відновленого файлу і занесіть цю інформацію в табл. 5.1.

2.2 Додатковий (інкрементальний) метод архівації

1. Створіть в папці з ім'ям 1 ще один текстовий файл backup2.txt.
2. З'ясуйте, чи встановлений атрибут "архивный" у властивостях файлу і занесіть цю інформацію в табл. 5.1.

3. Виконайте додаткову (инкрементальное) архівацію папки 1 у файл архіву з ім'ям increment.bkf (розширення додається автоматично) в папці з ім'ям 2.
4. З'ясуйте, чи встановлений атрибут "архивный" у властивостях початкових файлів backupl.txt і backup2.txt, а також кількість файлів в архіві і занесіть цю інформацію в табл. 5.1.
5. Виконайте зміну у файлі backupl.txt і повторите додаткову (інкрементальну) архівацію в той же файл архіву.
6. З'ясуйте, чи встановлений атрибут "архивный" у властивостях файлів backupl.txt і backup2.txt, а також кількість файлів в архіві і занесіть цю інформацію в табл. 5.1.
7. Нічого не змінюйте у файлах і ще раз виконайте додаткову (інкрементальну) архівацію в той же файл архіву.
8. З'ясуйте, чи встановлений атрибут "архивный" у властивостях файлів backupl.txt і backup2.txt, а також кількість файлів в архіві і занесіть цю інформацію в табл. 5.1.

2.3 Різницевий (диференціальний) метод архівації

1. Встановіть атрибути "архивный" для файлів backupl.txt і backup2.txt у папці 1 (це можна зробити зміною і збереженням файлів або установкою атрибуту уручну у властивостях файлу).
2. Виконайте звичайну (повне) архівацію папки 1 у файл архіву з ім'ям standart.bkf (розширення додається автоматично) у папці з ім'ям 2.
3. Виконайте зміну у файлі backupl.txt і виконаєте різницеву (диференціальне) архівацію у файл архіву different.bkf у папці 2.
4. З'ясуйте, чи встановлений атрибут "архивный" у властивостях файлів backupl.txt і backup2.txt, а також кількість файлів в архіві і занесіть цю інформацію в табл. 5.1.

5. Нічого не змінюйте у файлах і ще раз виконайте різницеву (диференціальне) архівацію в той же файл архіву.
6. З'ясуйте, чи встановлений атрибут "архивный" у властивостях файлів backup1.txt і backup2.txt, а також кількість файлів в архіві і занесіть цю інформацію в табл. 5.1.
7. Зробіть висновок про алгоритм різних методів архівації і використання атрибуту архівний.

Таблиця 5.1 – Результати виконання роботи

№ п/п	Виконана операція	Атрибут "архивный" backup1.txt	Атрибут "архивный" backup2.txt	Файли в архіві
Звичайний (повний) метод архівації				
1	Створення файлу backup1.txt			
2	Звичайна (повне) архівація			
3	Відновлення з повного архіву			
Додатковий (інкрементальний) метод архівації				
4	Створення файлу backup2.txt			
5	Додаткова (інкрементальна) архівація			
6	Зміна у файлі backup1.txt			
7	Додаткова (інкрементальна) архівація			
8	Зміна у файлі backup2.txt			
9	Додаткова (інкрементальна) архівація			
10	Повторна додаткова (інкрементальна) архівація			
Різницевий (диференціальний) метод архівації				
11	Установка атрибутів "архівний" для файлів backup1.txt і backup2.txt			
12	Звичайна (повне) архівація			
13	Зміна у файлі backup1.txt			
14	Різницева (диференціальне) архівація			
15	Зміна у файлі backup2.txt			
16	Різницева (диференціальне) архівація			
17	Повторна різницева (диференціальне) архівація			

3 Контрольні питання

1. Дайте визначення резервному копіюванню і архівації і назвіть відмінності між ними.
2. Назвіть принципи роботи основних методів резервного копіювання.
3. Опишіть відомі Вам схеми ротації носіїв для резервного копіювання.
4. Назвіть відомі Вам пристрої резервного копіювання і їх характеристики.
5. Опишіть можливості стандартної програми архівації Windows і дайте характеристику використовуваним в ній методам архівації.

ЛАБОРАТОРНА РОБОТА №6

«Конфігурація міжмережевих екранів»

Мета роботи: вивчити можливості і правила конфігурації міжмережевих екранів для забезпечення безпеки комп'ютерних мереж.

1 Короткі теоретичні відомості

Міжмережевий екран (firewall, брендмауер) - система міжмережевого захисту, що дозволяє розділити загальну мережу на дві або більш частин і реалізувати набір правил, визначальних умови проходження пакетів з даними через кордон з однієї частини загальної мережі в іншу. Як правило, ця межа проводиться між корпоративною (локальної) мережею і глобальною мережею Internet, хоча її можна організувати і усередині корпоративної мережі підприємства. Міжмережевий екран пропускає через себе весь трафік, ухвалюючи для кожного пакету рішення - пропускати його або відкинути - на основі визначених для нього набору правил фільтрації (рис. 6.1).

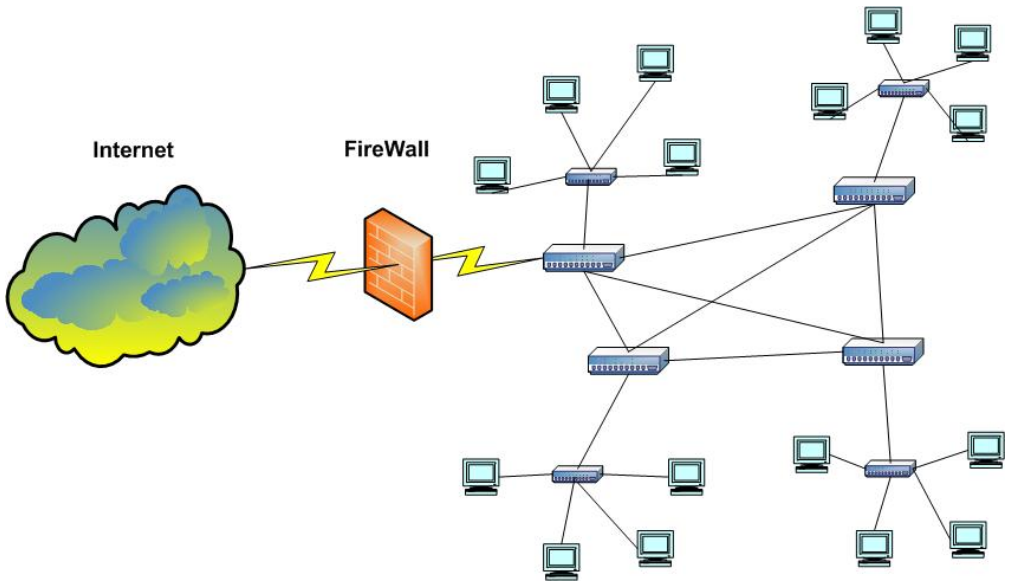


Рисунок 6.1 - Схема встановлення міжмережевого екрану

Існують чотири основні засоби, за допомогою яких міжмережеві екрани здійснюють контроль доступу і забезпечують реалізацію політик захисту:

- управління сервісами - визначення типів служб Internet, до яких можна дістати доступ з внутрішньої мережі назовні і із зовнішнього оточення усередину; міжмережевий екран може фільтрувати потік даних на основі IP-адрес і номерів портів TCP;
- управління напрямом руху - визначення напряму, в якому можуть ініціюватися і проходити через міжмережевий екран запити до тих або інших служб;
- управління користувачами - надання доступу до служб залежно від прав доступу користувачів, що звертаються до цих служб; ця функція звичайно застосовується до локальних користувачів; однак вона може застосовуватися і до потоку даних, що поступає від зовнішніх користувачів, що вимагає реалізації в якійсь формі технології аутентифікації, наприклад, забезпечуваної протоколом IPSec;
- управління поведінкою – контроль за використанням окремих служб; так, міжмережевий екран може фільтрувати електронну пошту, відсіваючи спам чи ж вирішувати доступ ззовні певної частини інформації, що знаходиться на локальному Web-сервері.

Можна виділити наступні типи міжмережевих екранів:

- фільтруючі маршрутизатори (packet-filtering router);
- шлюзи мережевого рівня;
- шлюзи прикладного рівня.

Звичайно міжмережеві екрани включають всі або більшість з цих компонент.

Фільтруючий маршрутизатор є маршрутизатором або працюючою на сервері програмою, що фільтрує пакети, які входять в корпоративну мережу і виходять з

неї. Фільтрація пакетів звичайно здійснюється на основі інформації, що міститься в IP і TCP заголовках пакетів, найчастіше для цього використовуються:

- IP адреса відправника пакету;
- IP адреса одержувача пакету;
- порт системи відправника;
- порт системи одержувача.

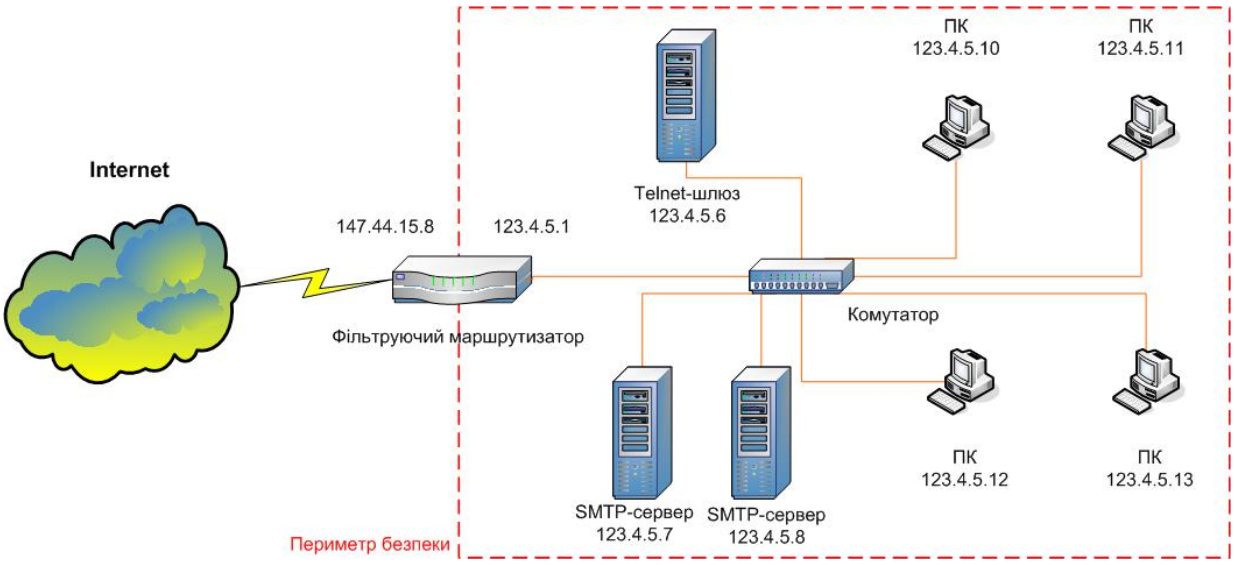
Фільтр пакетів звичайно представляється у вигляді списку правил, що використовують значення полів заголовків IP і TCP. Якщо виявляється відповідність одному з правил, то на підставі цього правила ухвалюється рішення про можливість передачі пакету далі. При невідповідності всім правилам виконується операція, передбачена для використання за умовчанням. Для неї існують наступні варіанти локальних політик захисту:

- Default = discard. Все, що не дозволено, заборонено.
- Default = forward. Все, що незаборонено, дозволено.

Як приклад роботи фільтруючого маршрутизатора розглянемо реалізацію політики безпеки, що допускає певні з'єднання з локальною корпоративною мережею 123.4/16 (рис. 6.2). При цьому з'єднання по протоколу telnet дозволені тільки з хостом 123.4.5.6, що виконує роль прикладного telnet-шлюзу, а з'єднання по протоколу SMTP - тільки з двома хостами 123.4.5.7 і 123.4.5.8, що виконують роль серверів електронної пошти. Обмін по протоколу NNTP дозволений тільки від зовнішнього сервера новин з адресою 129.6.48.254 і лише з NNTP сервером локальної мережі 123.4.5.9. Використання мережевої служби синхронізації часу NTP дозволене для всіх комп'ютерів локальної мережі.

Позитивні якості фільтруючих маршрутизаторів:

- порівняно невисока вартість;
- гнучкість у визначенні правил фільтрації;
- невелика затримка при проходженні пакетів.



Протокол	Адреса від-правника	Адреса одер-жувача	Порт відправ-ника	Порт одер-жувача	Дія
TCP		123.4.5.6	>1023	23	allow
TCP		123.4.5.7	>1023	25	allow
TCP		123.4.5.8	>1023	25	allow
TCP	1239.6.48.254	123.4.5.9	>1023	119	allow
UDP		123.4/16	>1023	123	allow
*	*	*	*		deny

Рисунок 6.2 - Приклад набору правил фільтрації

Недоліки фільтруючих маршрутизаторів:

- внутрішня мережа маршрутизується з Internet;
- правила фільтрації важкі в описі і відсутні засоби їх тестування;
- відсутня аутентифікація на призначеному для користувача рівні.

Шлюзи мережевого рівня інколи називають системами трансляції мережевих адрес (Network Address Translation - NAT), оскільки вони виключають пряму взаємодію між авторизованим клієнтом і зовнішнім хост - комп'ютером. Шлюз мережевого рівня приймає запит довіреного клієнта на конкретні послуги і після перевірки

чи задовольняє клієнт базовим критеріям фільтрації (наприклад, чи може DNS - сервер визначити IP-адреса клієнта і асоційоване з ним ім'я) встановлює з'єднання із зовнішнім хостом. Шлюз мережевого рівня виконує процедуру трансляції адрес, при якій відбувається перетворення внутрішніх IP адрес локальної мережі на одну "надійну" IP адресу інтерфейсу міжмережевого екрану, через який передаються всі витікаючі з локальної мережі пакети.

Кожен раз, коли витікаючий пакет передається через NAT - маршрутизатор локальна IP адреса відправника замінюється на "чесну" IP адресу (рис. 6.3). Окрім цього, в полі порту відправника заголовка TCP заноситься 16-бітовий індекс таблиці трансляції адрес (з максимум 65536 рядками). Кожен рядок таблиці містить цей індекс, первинну IP адресу відправника і первинний порт відправника. Після описаної заміни прораховуються контрольні суми заголовків IP і TCP і заносяться у відповідні поля пакету. Коли NAT- маршрутизатор отримує у відповідь пакети для даного процесу, значення порту одержувача в заголовку TCP використовується для пошуку індексу в таблиці трансляції адрес. З визначеного по індексу рядка витягується локальна IP адреса відправника і первинний TCP порт відправника, які заносяться у відповідні поля пакету. Після цього знову перераховуються контрольні суми в полях IP і TCP заголовків, заносяться у відповідні поля пакету, після чого пакет прямує на робочу станцію локальної мережі.

Шлюз мережевого рівня відстежує процедуру квитування зв'язку по протоколу TCP через порядкові номери сегментів даних. Коли сеанс завершується, шлюз видаляє відповідний елемент з таблиці і розриває з'єднання. Недоліком шлюзів мережевого рівня є те, що після встановлення зв'язку вони не можуть перевіряти вміст пакетів на прикладному рівні. Також для них відсутня призначена для користувача аутентифікація.

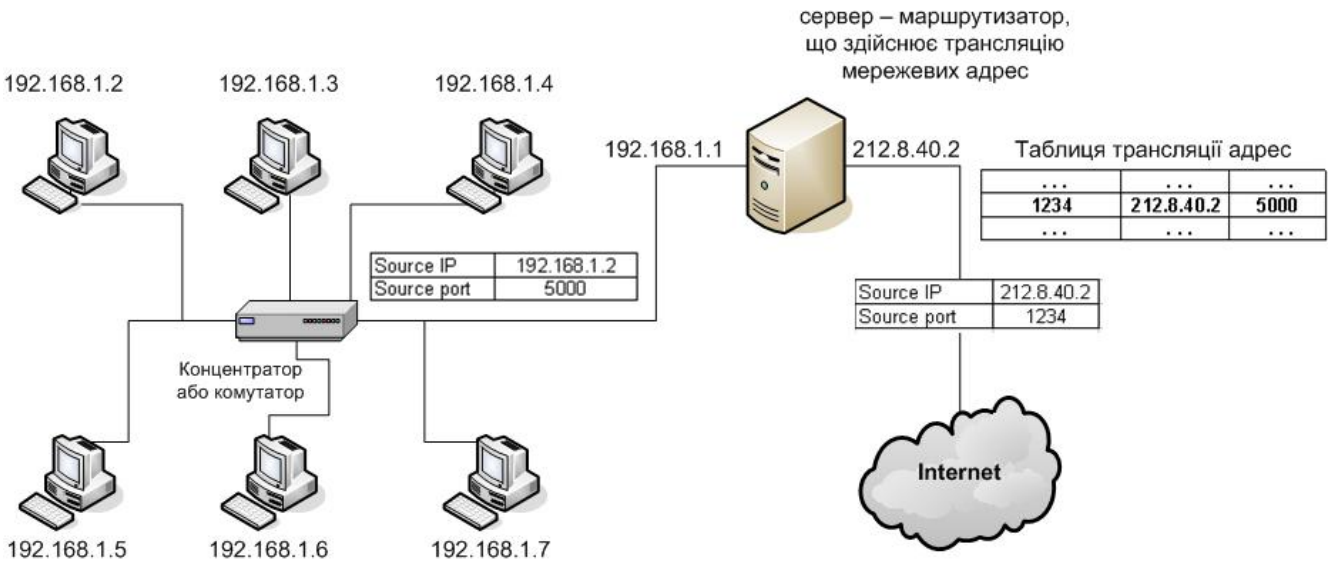


Рисунок 6.3 - Трансляція мережевих адрес шлюзом мережевого рівня

Шлюзи прикладного рівня також виключають пряму взаємодію між авторизованим клієнтом і зовнішнім хостом, проте при цьому вони фільтрують всі вхідні і витікаючі пакети на прикладному рівні. Це досягається установкою і настройкою програмного забезпечення повноважних серверів – проксі - серверів (proxy servers), що перенаправляють через шлюз інформацію прикладного рівня, що генерується хостами (рис. 6.4). Користувач зв'язується з цим шлюзом за допомогою такого побудованого на основі TCP/IP застосування, як HTTP-браузер, Telnet-термінал або FTP- клієнт. Ці застосування передають шлюзу ім'я віддаленого вузла, до якого необхідно дістати доступ. Шлюз зв'язується з відповідним застосуванням віддаленого вузла і ретранслює сегменти TCP, що містять дані застосування локального користувача. Шлюз можна настроїти так, щоб він підтримував тільки певні можливості застосування, які адміністратор мережі вважає допустимими з погляду безпеки.

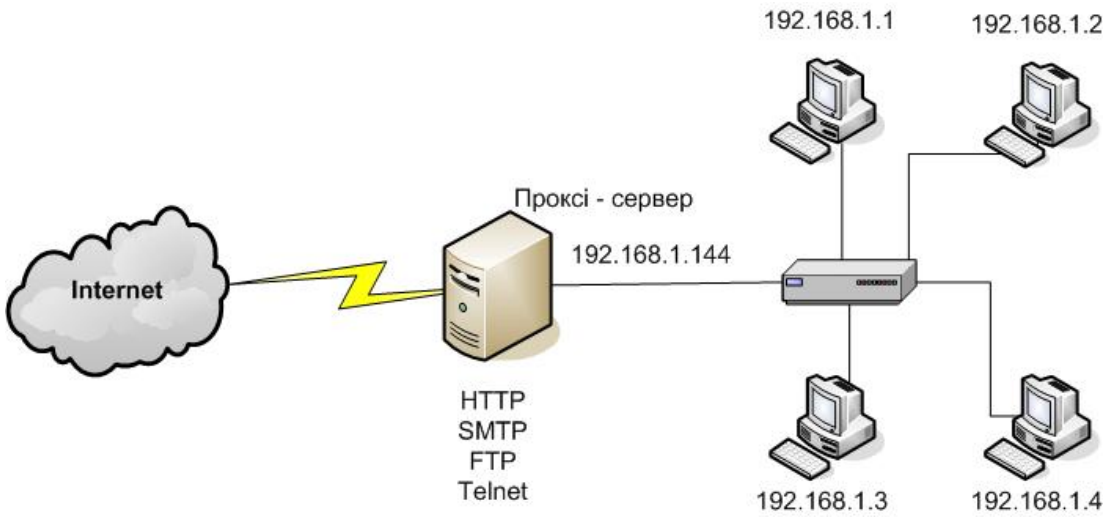


Рисунок 6.4 - Шлюз прикладного рівня – проксі – сервер

У загальному випадку шлюзи прикладного рівня забезпечують надійніший захист, ніж фільтри пакетів. Замість того щоб перевіряти численні можливі комбінації дозволів і заборон на рівні TCP і IP, шлюзу прикладного рівня доводиться розглядати лише обмежене число застосувань. Крім того, при цьому легко протоколювати і контролювати весь вхідний потік даних прикладного рівня. Проте, для досягнення вищого рівня безпеки і гнучкості шлюзи прикладного рівня і фільтруючі маршрутизатори можуть об'єднуватися в одному міжмережевому екрані. Переваги використання проксі-серверів:

- пропускають пакети лише тих служб, які їм доручено обслуговувати (частіше за все HTTP, HTTPS, FTP, Gopher);
- приховують структуру мережі, що захищається;
- дозволяють виконувати кешування запрошуваної Інтернет інформації і будувати ієрархію кешів;
- дозволяють виконувати аутентифікацію користувачів;
- дозволяють вести статистику відвідувань за витікаючими IP адресами.

Основним недоліком шлюзів даного типа є додаткове навантаження на процесор, що створюється кожним з'єднанням. Насправді між двома кінцевими корис-

тувачами встановлюється два зв'язані з'єднання, загальною точкою яких є шлюз, і шлюзу доводиться перевіряти весь потік даних в обох напрямах, щоб переправити його далі.

Міжмережеві екрани можуть бути інтегровані в маршрутизатори або виконані як окремі пристрої.

Основні схеми захисту на базі міжмережевих екранів:

- міжмережевий екран - фільтруючий маршрутизатор;
- міжмережевий екран на основі двопортового шлюзу;
- міжмережевий екран на основі екранованого шлюзу;
- міжмережевий екран - екранована підмережа.

Міжмережевий екран - фільтруючий маршрутизатор - найбільш простий в реалізації - виконує фільтрацію вхідних і витікаючих пакетів на основі аналізу їх адрес і портів (рис.6.5).

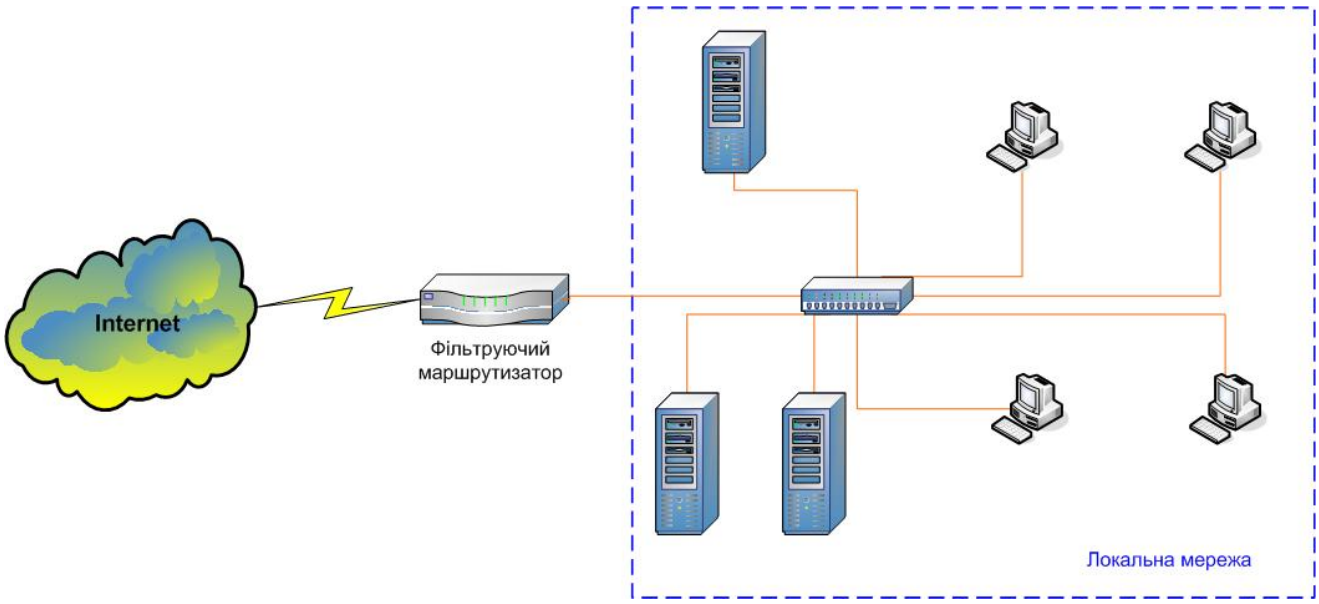


Рисунок 6.5 - Міжмережевий екран на основі фільтруючого маршрутизатора

Міжмережевий екран на основі двопортового шлюзу прикладного рівня часто є комп'ютером з двома мережевими інтерфейсами, при передачі даних між якими здійснюється фільтрація (рис. 6.6). Для забезпечення додаткового захисту між шлюзом прикладного рівня і Internet звичайно розміщують фільтруючий маршрутизатор. Між прикладним шлюзом і фільтруючим маршрутизатором утворюється внутрішня екранована підмережа - демілітаризована зона, в якій можна розташовувати доступні ззовні інформаційні сервери.

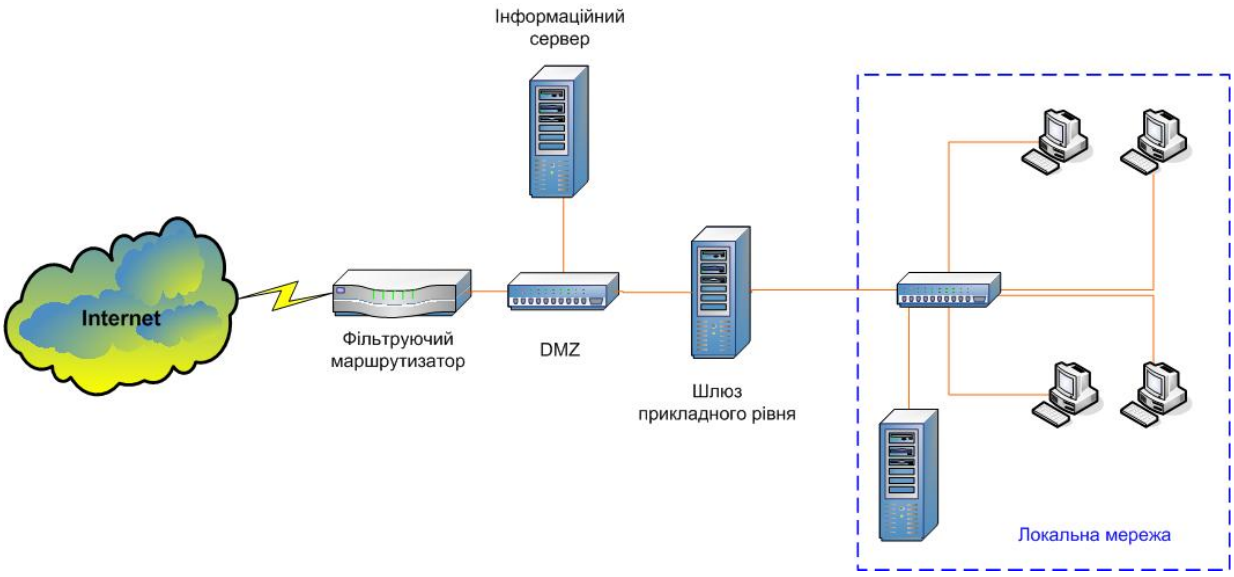


Рисунок 6.6 - Міжмережевий екран з шлюзом прикладного рівня і фільтруючим маршрутизатором

Міжмережевий екран на основі екранованого шлюзу прикладного рівня, реалізованого на комп'ютері лише з одним мережевим інтерфейсом, є гнучкіший варіант реалізації міжмережевого екрану, оскільки є можливість реалізувати з'єднання з Internet для деяких служб через проксі - сервер, а для деяких через фільтруючий маршрутизатор (рис.6.7).

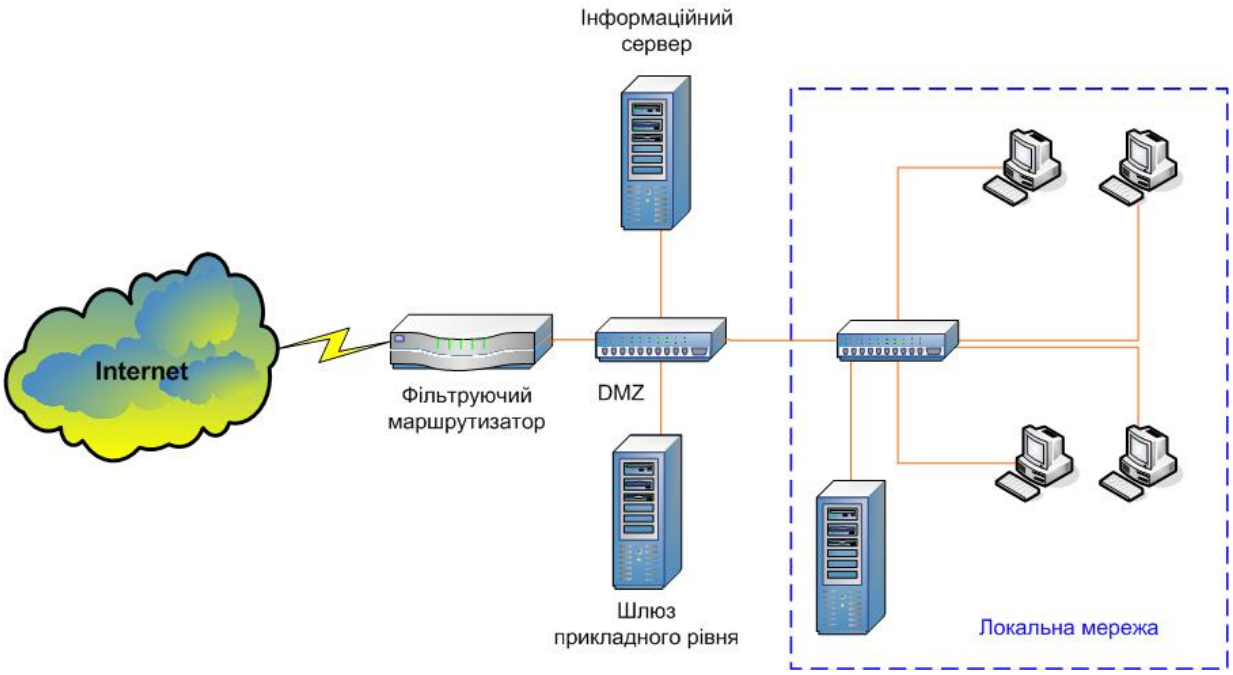


Рисунок 6.7 - Міжмережевий екран з екранованим шлюзом прикладного рівня і фільтруючим маршрутизатором

Міжмережевий екран, що складається з екранованою зовнішнім і внутрішнім фільтруючими маршрутизаторами підмережею показаний на рис. 6.8.

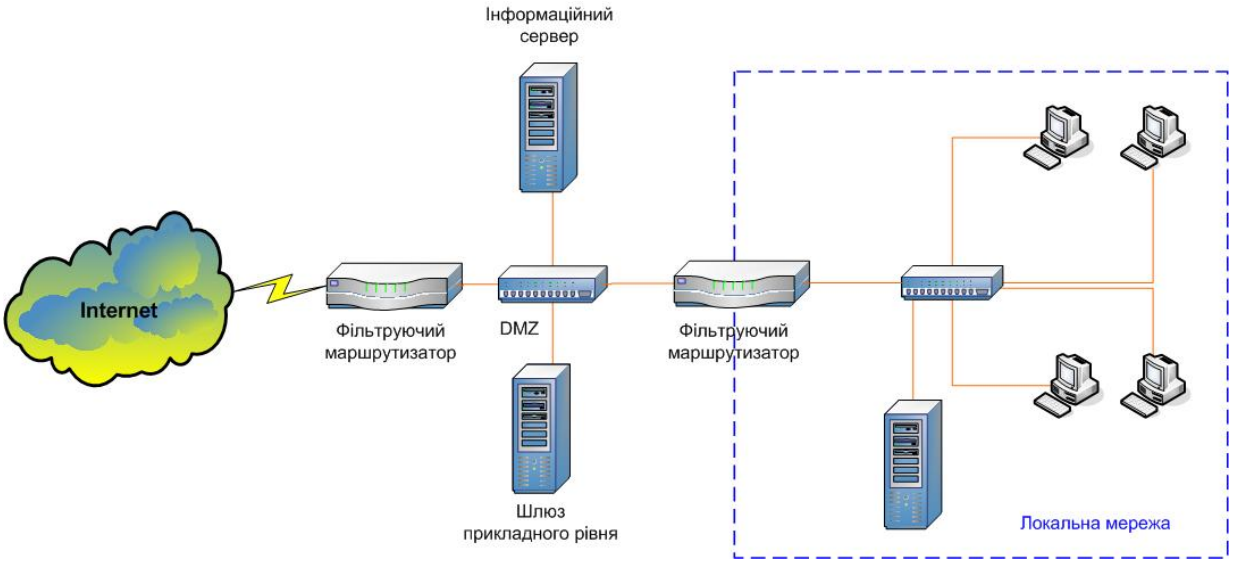


Рисунок 6.8 - Міжмережевий екран з екранованою фільтруючими маршрутизторами підмережею

У цій підмережі розташовується шлюз прикладного рівня і сервери. Зовнішній маршрутизатор дозволяє трафік зовнішніх користувачів лише до проксі - серверу і серверам в демілітаризованій зоні. Внутрішній маршрутизатор захищає внутрішню мережу як від Internet, так і від екранованої підмережі (у разі її компрометації).

2 Порядок виконання роботи

Запишіть команди, що встановлюють правила фільтруючого маршрутизатора, що захищає наведену на рис. 6.9 мережу (123.4.5.0/28), так, щоб:

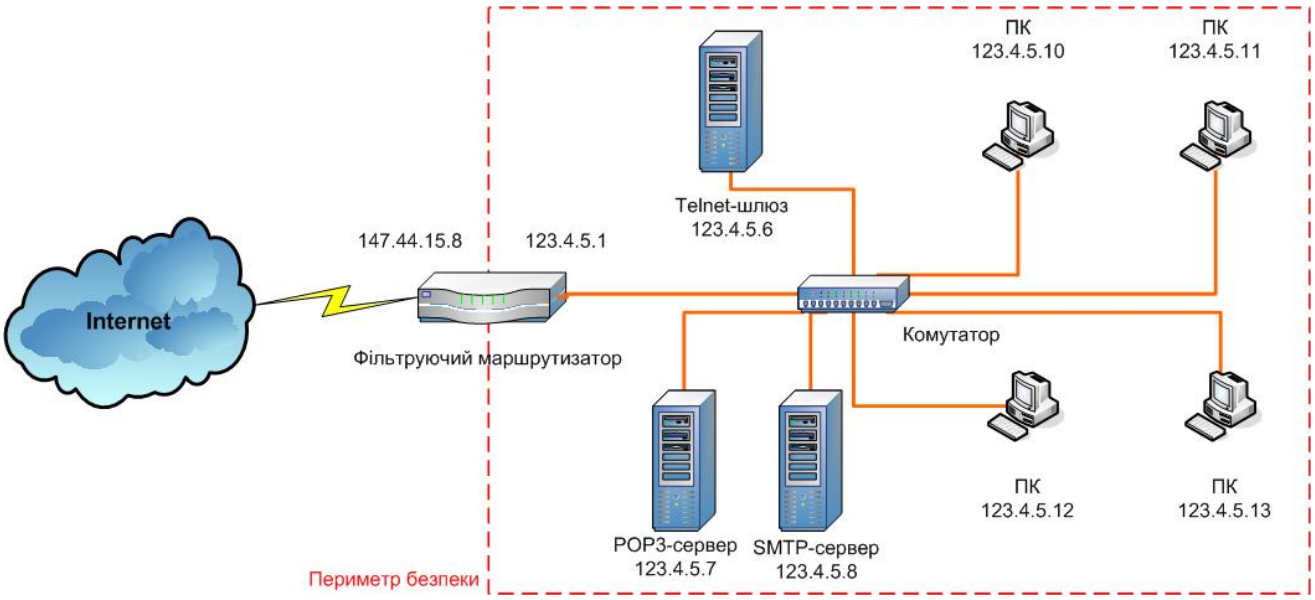


Рисунок 6.9 – Структура мережі для виконання завдання

- зовнішні користувачі Internet могли працювати лише з Telnet -шлюзом і SMTP - сервером;
- внутрішні користувачі могли відправляти і отримувати електронну пошту через SMTP і POP3 сервер, відповідно;
- внутрішні користувачі мали доступ до Internet для роботи з HTTP і FTP серверами і могли користуватися символьними іменами вузлів Internet.

Команди записати у вигляді:

ipfw команда (add) № правила allow/deny протокол

from адреса відправника to адреса одержувача номер протоколу.

3 Контрольні питання

1. Що називають міжмережевими екранами? Які типи міжмережевих екранів Ви знаєте?
2. Опишіть принцип роботи фільтруючих маршрутизаторів, їх переваги і недоліки.
3. Опишіть принцип роботи систем трансляції адрес, їх переваги і недоліки.
4. Опишіть принцип роботи проксі - серверів, їх переваги і недоліки.
5. Опишіть основні схеми захисту мереж за допомогою міжмережевих екранів.

ЛАБОРАТОРНА РОБОТА №7

“Конфігурація списків доступу маршрутизаторів”

Мета роботи: вивчити можливості і правила конфігурації міжмережевих екранів для забезпечення безпеки комп'ютерних мереж.

1 Короткі теоретичні відомості

Списки доступу (access-lists) є послідовністю команд, що дозволяють або забороняють передачу пакетів через маршрутизатор. Команди стандартних IP списків доступу ухвалюють рішення про дозвіл або заборону пакетів винятково на підставі значення адреси відправника пакету. Списки доступу можуть використовуватися для управління розповсюдженням і прийомом пакетів з маршрутною інформацією, формування трафіку, визначення трафіку, який дозволить забезпечити необхідний захист. Причини застосування політики захисту на базі списків доступу можуть бути різні, наприклад, - необхідність запобігання зовнішнім атакам на комп'ютери локальної мережі, необхідність ізоляції трафіку між підрозділами компанії, необхідність розподілу навантаження на мережу і ін. При використанні списків доступу для організації міжмережевого екрану маршрутизатори зможуть обмежувати або запобігати доступу до комп'ютерів внутрішньої мережі із зовнішньої мережі, наприклад, Internet. Звичайно списку доступу такого типа розміщуються в маршрутизаторі, об'єднуючому такі мережі. При використанні списків доступу для організації ізоляції трафіку між підрозділами, список доступу звичайно розміщується на маршрутизаторі усередині локальної мережі.

Основний формат стандартного IP списку доступу наступний:

```
access-list[#] [permit|deny] [source-address|keyword any] [source
mask]
```

Список доступу звичайно містить достатньо багато рядків вказаного вище формату з метою регулювання трафіку, витікаючого з багатьох адрес. Кожен рядок списку доступу повинен містити однаковий ідентифікатор списку доступу. Списки доступу обробляються зверху "вниз", що означає, що спочатку інформація з пакету оцінюється першим рядком списку, потім другий і т.д. Маршрутизатор припиняє роботу із списком доступу після першої відповідності інформації з пакету параметрам команди, тому найзагальніші інструкції повинні бути розміщені на початку списку. Різні списки доступу можуть бути визначені відповідно до різних протоколів, підтримуваних маршрутизатором. Маршрутизатор "знає" тип списку доступу по його номеру. Діапазон номерів для стандартних IP списків доступу - від 1 до 99.

Команди списку містять ключові слова permit (дозволити) або deny (відхилити), які інструктують маршрутизатор дозволити передавати пакет або відхилити його залежно від значень наступних в команді параметрів, перш за все, від значення адреси відправника пакету. Існує можливість заміни значень адреси відправника ключовим словом any (будь-які) що примусить маршрутизатор аналізувати кожен пакет.

Стандартний список доступу IP враховує маску підмережі адреси відправника, яка буде застосована до початкової адреси IP. Проте алгоритм використання цієї маски дещо відрізняється від стандартного алгоритму маски IP підмережі. При використанні маски підмережі адреси відправника із списками доступу IP біти, встановлені в 0 означають точну відповідність, а біти, встановлені в 1, означають будь-яку комбінацію відповідних бітів в адресі. Наприклад, для вказівки в команді всіх комп'ютерів мережі класу C 192.1.1.0, комбінація адреси відправника і його маски буде наступною: 192.1.1.0 і 0.0.0.255. Це означає, що в першому, другому, і третьому октеті адреси всі біти адреси відправника аналізованого пакету повинні точно відповідати вказаній адресі (192.1.1), а біти в четвертому октеті можуть мати будь-яку комбінацію. Використовуючи таку комбінацію адреси і маски можна в

одній команді як параметр вказати цілу мережу. Таким чином, ключове слово any (будь-які) відповідає комбінації адреси і маски 0.0.0.0 255.255.255.255, де маска 255.255.255.255 дозволяє встановлювати будь-яку комбінацію бітів в будь-якому з чотирьох октетів. Використання параметра маски адреси відправника є опціональним. Якщо в команді не указується маска, маршрутизатор за умовчанням використовуватиме маску 0.0.0.0, що означає точну відповідність адресі.

Існує одна особливість використання списків доступу. Після створення списку доступу, маршрутизатор Cisco відхилятиме будь-які пакети з адресами відправника, не вказаними ні в одній команді списку.

Виконання роботи здійснюється в програмі симуляції роботи комп'ютерних мереж Boson NetSim™, що дозволяє виконувати моделювання комп'ютерних мереж на персональному комп'ютері з використанням широкого діапазону керованих мережевих пристроїв, включаючи маршрутизатори Cisco®, декілька типів комутаторів Catalyst™ і мережеві робочі станції. Boson NetSim™ грунтується на технології Boson's Virtual Packet, яка створює при моделюванні віртуальні мережеві пакети, які маршрутизуються і комутуються в модельованій мережі. Це дозволяє будувати віртуальні таблиці маршрутизації для різних мережевих протоколів і тим самим повністю емулювати мережеве середовище. Boson NetSim™ підтримує наступні протоколи і мережеві засоби: RIP, OSPF, IP, IGRP, EIGRP, IPX, ISDN, Hosts, PPP, CHAP, ARP, Ping, Reload, Ping IPX. Крім того можливе завантаження/збереження конфігурації мережевих пристроїв, установка консольних паролів, управління паролями, контекстна довідка і ін.

Списки доступу створюються в глобальному режимі конфігурації маршрутизатора. Всі стандартні списки доступу IP повинні бути пронумеровані в діапазоні 1-99, ми використовуватимемо #1. Припустимо, що необхідно дозволити трафік від адреси 1.1.1.1 і заборонити решту трафіку. Процедура конфігурації включає наступні команди:

```
eRouter#conf t - перехід в глобальний режим конфігурації
Enter configuration commands, one per line. End with CNTL/Z.
eeRouter(config) #access-list 1 permit 1.1.1.1 - завдання списку доступу з однією командою
eeRouter(config) #^Z  - вихід з глобального режиму конфігурації
eRouter#
```

Оскільки маска адреси відправника не вказана, маршрутизатор використовує значення за умовчанням 0.0.0.0 (що означає точну відповідність адресі). Також маршрутизатор автоматично відхиляє всі пакети з адресами відправника іншими, чим 1.1.1.1.

Перш, ніж список доступу розпочне працювати, його необхідно прив'язати до певного інтерфейсу. Команда, що виконує цю функцію, має вигляд:

```
ip access-group [access-list-number] [in | out]
```

Списки доступу можуть бути прив'язані до інтерфейсу маршрутизатора, як для витікаючих, так і для вхідних пакетів. У разі прив'язки для вхідних пакетів, маршрутизатор аналізує адресу відправника кожного вхідного в інтерфейс пакету на відповідність командам списку. Якщо команда явно або неявно "дозволяє" пакет, він маршрутизується і передається у напрямку до мережі одержувача, якщо ж команди списку доступу явно або неявно "забороняють" пакет, пакет відкидається. У разі прив'язки для витікаючих пакетів, маршрутизатор аналізує адресу відправника кожного витікаючого з інтерфейсу пакету на відповідність командам списку і виконує аналогічні дії.

Для прив'язки створеного списку доступу до інтерфейсу маршрутизатора Ethernet 0 для вхідних пакетів необхідно виконати наступні команди:

```
eRouter#conf t - перехід в глобальний режим конфігурації
Enter configuration commands, one per line. End with CNTL/Z.
eeRouter(config) #int Ethernet 0  - перехід в режим конфігурації інтерфейсу
                                    Ethernet 0
eeRouter(config-if) #ip access-group 1 in - прив'язка списку доступу з
                                            номером 1 до поточного
                                            інтерфейсу для вхідних
                                            пакетів
eeRouter(config-if) #^Z - вихід з глобального режиму конфігурації
eRouter#
```

Аналогічно, для прив'язки створеного списку доступу до інтерфейсу маршрутизатора Ethernet 0 для витікаючих пакетів необхідно виконати наступні команди:

```
eRouter#conf t - перехід в глобальний режим конфігурації
Enter configuration commands, one per line. End with CNTL/Z.
eeRouter(config) #int Ethernet 0 - перехід в режим конфігурації інтерфейсу
                                   Ethernet 0
eeRouter(config-if) #ip access-group 1 out - прив'язка списку доступу з
                                             номером 1 до поточного
                                             інтерфейсу для витікаю-
                                             чих пакетів
eeRouter(config-if) #^Z - вихід з глобального режиму конфігурації
eRouter#
```

Створимо список доступу #2, з наступними критеріями: вирішуються всі пакети з мережі 10.1.1.0 255.255.255.128, але забороняються всі пакети з мережі 10.1.1.128 255.255.255.128, також забороняються всі пакети з мережі 15.1.1.0, окрім пакетів комп'ютера 15.1.1.5, решта трафіку дозволена. Процедура конфігурації включає наступні команди:

`eRouter#conf t` - перехід в глобальний режим конфігурації
`Enter configuration commands, one per line. End with CNTL/Z.`

Далі задаються команди списку доступу з номером 2

`eeRouter(config) #access-list 2 deny 10.1.1.128 0.0.0.127` - забороняються пакети з мережі 10.1.1.128 255.255.255.128
`eeRouter(config) #access-list 2 permit 15.1.1.5` - дозволяються пакети хоста 15.1.1.5
`eeRouter(config) #access-list 2 deny 15.1.1.0 0.0.0.255` - забороняються пакети з мережі 15.1.1.0
`eeRouter (config) #access-list 2 permit any` - дозволяється вся решта трафіку (у тому числі і з мережі 10.1.1.0 255.255.255.128)
`eeRouter (config-if) #^Z` - вихід з глобального режиму конфігурації
`eRouter#`

У першій команді списку використовується маска підмережі адреси відправника 0.0.0.127. Вона означає, що молодші 7 бітів адреси (використовувані для адресації хостів даної підмережі) можуть бути будь-якими, а 8 біт повинен точно відповідати відповідному біту адреси (в даному випадку він рівний 1). Таким чином, параметри команди 10.1.1.128 0.0.0.127 відповідають адресам підмережі 10.1.1.128 255.255.255.128 з діапазоном адрес 10.1.1.128-10.1.1255.

Дуже важливо звернути увагу на те, що список доступу не відповідають переліку критеріїв, які необхідно було виконати. Це наслідок того, що списки доступу обробляються зверху вниз, і при першій же відповідності адреси відправника пакету параметру команди подальша обробка списку припиняється. Тому спочатку записується команда, що дозволяє пакети комп'ютера 15.1.1.5, а потім команда, що забороняє пакети з мережі 15.1.1.0. Якщо ці команди поміняти місцями, то пакети комп'ютера 15.1.1.5 також заборонялися б, оскільки цей комп'ютер належить мережі 15.1.1.0 255.255.255.0, і команда заборони знаходилася б перед командою дозволу.

Таким чином, послідовність конфігурації списків доступу наступна:

- визначення правил передачі трафіку через маршрутизатор.
- створення списку доступу з номером в діапазоні від 1 до 99.
- прив'язка списку доступу до певного інтерфейсу або для тих, що входять, або для витікаючих пакетів.

Виходячи із загальних міркувань, стандартні IP списки доступу повинні конфігуруватися на маршрутизаторах, розташованих якомога ближче до одержувачів пакетів і далі від їх відправників, однак це не завжди можна виконати. Унаслідок того, що стандартні IP списки доступу оперують лише адресою відправника, деталізація трафіку не завжди можлива. Необхідно дотримуватися обережності для запобігання конфігурації небажаної політики. У випадку якщо стандартний список доступу розміщується на маршрутизаторі в мережі відправника пакетів, можливо, що бажаний доступ до інших пристроїв буде неможливий. Наприклад, якщо створений список доступу 2 прив'язаний для вхідних пакетів до інтерфейсу Ethernet, безпосередньо пов'язаного з мережею 15.1.1.0, маршрутизовуваться будуть пакети єдиної робочої станції 15.1.1.5. При цьому маршрутизація пакетів інших станцій в інші локальні мережі неможлива. Тому в цьому випадку має сенс прив'язки списку до інтерфейсу маршрутизатора, який пов'язує локальну мережу підприємства з територіальною мережею і прив'язка його для витікаючих пакетів.

Припустимо, що робоча станція С на рис. 7.1 має адресу 15.1.1.5, а робоча станція D - 10.1.1.133.

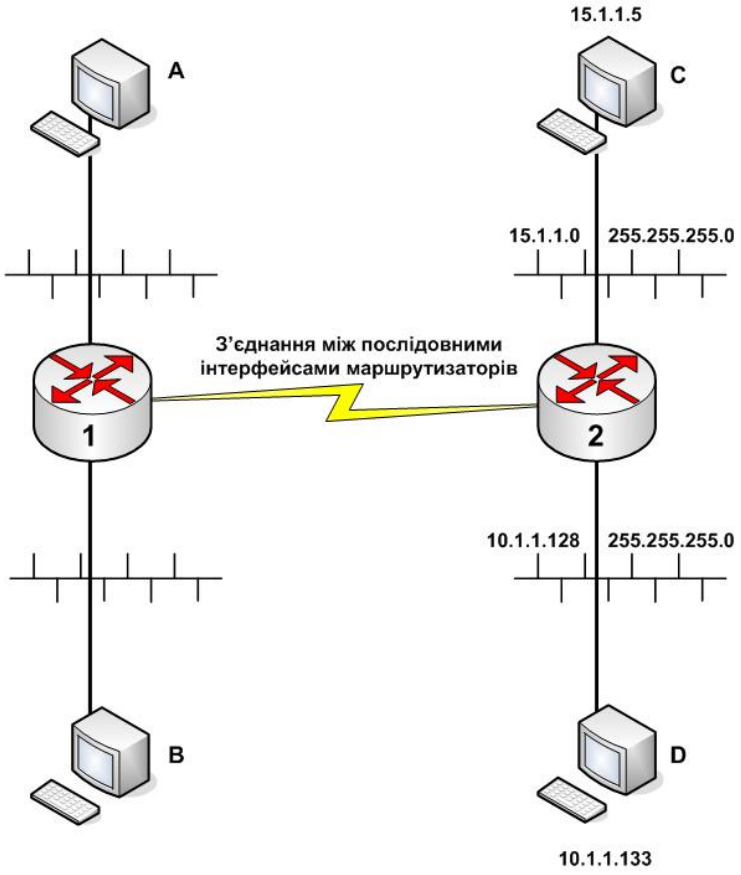


Рисунок 7.1 - Приклад визначення місця розміщення списку доступу

Допустимий необхідно застосувати політику для робочої станції А, що дозволяє доступ до неї тільки з комп'ютера С мережі 15.1.1.0 255.255.255.0 і що забороняє доступ з решти комп'ютерів мережі 15.1.1.0 255.255.255.0 і всіх комп'ютерів мережі 10.1.1.128 255.255.255.0. Таку політику може реалізувати створений список доступу 2. Проте для реалізації такої політики критичним є розміщення списків доступу. Якщо список доступу 2 прив'язати для витікаючих пакетів до послідовного інтерфейсу маршрутизатора 2, ми виконаємо задачу, але ми також заборонятимемо

трафік з мережі 10.1.1.128 255.255.255.0 в мережу, до якої належить робоча станція B, що є небажаним. Ті ж проблеми зберігаються, якщо список доступу застосовується до вхідних пакетів в послідовний інтерфейс маршрутизатора 1. Якщо ж ми розмістимо цей список доступу для витікаючих пакетів з інтерфейсу Ethernet маршрутизатора 1, приєднаного до мережі, до якої належить робоча станція A, необхідна політика буде застосована без збитку для проходження решти трафіку.

2 Порядок виконання роботи

1. Завантажте конфігурацію лабораторної роботи, виконавши натиснення кнопки Lesson Navigator в програмі Boson NetSim. У дереві об'єктів виберіть Main Menu - Boson Lessons and Labs - CCNA/ICND Stand-alone Labs - Lesson 23 Standard Access List - The Lab. Виконайте натиснення кнопки Load This Lab для завантаження мережевої топології. Прогляньте топологію, виконавши натиснення кнопки NetMap.
2. Підключіться до маршрутизатора eRouter1 (виконайте команду en) і встановите IP адресу 24.17.2.1 255.255.255.240 інтерфейсу ethernet0. Для цього увійдіть до глобального режиму конфігурації, потім в режим конфігурації інтерфейсу і виконаєте дві команди:

`eRouterl (config-if) #ip address 24.17.2.1 255.255.255.240` - призначення IP адреси і маски підмережі інтерфейсу

eRouterl `(config-if) #no shutdown` - переведення інтерфейсу в активний стан

3. Аналогічно встановите IP адресу 24.17.2.17 255.255.255.240 послідовного інтерфейсу (serial0).
4. Підключіться до маршрутизатора eRouter2 і встановите IP адресу 24.17.2.2 255.255.255.240 інтерфейсу ethernet0.

5. Виконайте ping інтерфейсу ethernet0 eRouter1 щоб переконатися в наявності зв'язку (ping 24.17.2.1).
6. Підключіться до маршрутизатора eRouter4 і встановите IP адресу 24.17.2.18 255.255.255.240 інтерфейсу serial0 і виконаєте ping інтерфейсу serial0 eRouter1 щоб переконатися в наявності зв'язку (ping 24.17.2.17).
7. Після призначення адрес інтерфейсів необхідно визначити протокол маршрутизації трафіку eRouter2 і eRouter4. Використовуйте RIP у якості протоколу маршрутизації. Дозвольте використання RIP на eRouterl і додайте мережу для ethernet0 і serial0. Для цього перейдіть в режим глобальної конфігурації і виконайте команди:

`eRouterl(config) #router rip` - вхід в режим конфігурації маршрутизації

`eRouterl(config-router) #network 24.0.0.0` - вказівка номерів мереж, приєднаних до маршрутизатора

8. Дозвольте використання RIP на eRouter2 і додайте мережу для ethernet0.
9. Дозвольте використання RIP на eRouter4 і додайте мережу для serial0.
10. Перевірте наявність зв'язку, виконавши ping інтерфейс eRouter2 ethernet0 (24.17.2.2) з eRouter4.
11. Стандартний список доступу повинен забороняти проходження пакетів ping eRouter4 до eRouter2. Виконайте конфігурацію цього списку доступу на eRouter2. Підключіться до eRouter2 і перейдіть в глобальний режим конфігурації.
12. Створіть список доступу з номером 1, який заборонятиме пакети, що надходять з єдиної адреси 24.17.2.18 і вирішувати всю решту трафіку.
13. Виконайте прив'язку створеного списку доступу до інтерфейсу ethernet0 для вхідних пакетів.

14. Перевірте, чи зможете Ви виконати ping eRouter2 з eRouter4. Підключіться до eRouter4 і перевірте чи зможете Ви виконати ping інтерфейсу ethernet0 (24.17.2.2) eRouter2. У випадку, якщо пакети не проходять, список доступу конфігурований правильно.
15. Ви можете використовувати команду show ip interface для проглядання списків доступу на визначеному (або всіх) інтерфейсах.
16. Команда show access-lists відобразить списки доступу для даного маршрутизатора, також вона виводить інформацію про команди, які були використані і кількості пропущених і відкинутих пакетів.
17. Виконайте команду show access-lists для eRouterl eRouter2 і eRouter4 і приведіть лістинг виведеної інформації.

3 Контрольні питання

1. Що є списки доступу, використовувані маршрутизаторами Cisco? Для чого вони використовуються?
2. Приведіть формат стандартного IP списку доступу і опишіть використання його параметрів.
3. Назвіть особливості використання маски підмережі відправника для вказівки діапазонів адрес комп'ютерів в командах списків доступу.
4. Опишіть алгоритм перевірки пакетів командами списків доступу.
5. Назвіть послідовність і команди конфігурації списків доступу маршрутизаторів Cisco.

ЛАБОРАТОРНА РОБОТА №8

«Дослідження методів видалення надмірної інформації»

Мета роботи: визначити можливість використання стандартних програмних засобів архівації даних для видалення надмірної інформації в шифрованих повідомленнях

1 Короткі теоретичні відомості

Більшість сучасних форматів запису даних містять їх у вигляді, зручному для швидкого маніпулювання, для зручного прочитання користувачами. При цьому дані займають обсяг більший, ніж це дійсно потрібно для їх зберігання. Алгоритми, які усувають надмірність запису даних, називаються алгоритмами стиснення даних, або алгоритмами архівації. Нині існує величезна безліч програм для стиснення даних, заснованих на декількох основних способах.

У сучасному криптоаналізі, доведено, що ймовірність злому криптосхемы за наявності кореляції між блоками вхідної інформації значно вище, ніж за відсутності такої. Алгоритми стиснення даних за визначенням і мають своїм основним завданням усунення надмірності, тобто кореляцій між даними у вхідному тексті.

Всі алгоритми стиснення даних якісно діляться на:

- алгоритми стиснення без втрат, при використанні яких дані на приймальні відновлюються без щонайменших змін;
- алгоритми стиснення із втратами, які видаляють з потоку даних інформацію, що трохи впливає на суть даних, або взагалі не сприйману людиною (такі алгоритми зараз розроблені лише для аудіо- і відео- зображень).

У криптосистемах використовується тільки перша група алгоритмів.

Існує два основні методи архівації без втрат:

- алгоритм Хаффмана (англ. Huffman), орієнтований на стиснення послідовностей байт, не зв'язаних між собою;

- алгоритм Лемпеля-Зіва (англ. Lempel, Ziv), орієнтований на стиснення будь-яких видів текстів, тобто що використовує факт неодноразового повторення "слів" – послідовностей байт.

Практично всі популярні програми архівації без втрат (ARJ, RAR, ZIP і т.п.) використовують об'єднання цих двох методів – алгоритм LZH.

2 Порядок виконання роботи

2.1. Виконайте архівацію кожного файлу з папки «Задание» за допомогою програми WinZip і занесіть параметри початкових файлів і отриманих архівів в табл. 8.1.

Таблиця 8.1 – Результати архівації файлів різного формату

№ п/п	Ім'я і тип файлу	Початковий розмір, кБ	Розмір архіву, кБ		Ступінь стиснення %	
			WinZIP	WinRAR	WinZIP	WinRAR
1						
2						
.						
.						
.						
10						

2.2. Для кожного створеного архіву визначите ступінь стиснення початкового файлу.

2.3. Визначте способи підвищення ступеня стиснення архівів і перевірте їх на кожному файлі з папки «Задание».

2.4. Повторіть п. 2.1 – 2.4 за допомогою програми WinRAR.

2.5. Порівняйте дані, одержані за допомогою програм WinZip і WinRAR і зробити висновки про ефективність їх використання для видалення надмірної інформації.

3 Контрольні питання

1. Загальні принципи архівації даних
2. Узагальнена схема симетричної криптосистеми
3. Способи підвищення ступеня стиснення
4. Методика розрахунку ступеня стиснення архівів
5. Залежність ступеня стиснення від типу файлів, що архівуються

ЛАБОРАТОРНА РОБОТА №9

«Вивчення принципів архівації даних методами Хаффмана і Лемпеля-Зіва»

Мета роботи: виконати дослідження алгоритмів Хаффмана і Лемпеля-Зіва

1 Короткі теоретичні відомості

1.1. Алгоритм Хаффмана

Алгоритм заснований на тому факті, що деякі символи із стандартного 256-символьного набору в довільному тексті можуть зустрічатися частіше за середній період повтору, а інші, відповідно, – рідше. Отже, якщо для запису поширених символів використовувати короткі послідовності біт, завдовжки менше 8, а для запису рідкісних символів – довгі, то сумарний обсяг файлу зменшиться.

Хаффман запропонував дуже простий алгоритм визначення того, який символ необхідно кодувати яким кодом для отримання файлу з довжиною, дуже близькою до його ентропії (тобто інформаційної насиченості). Нехай є список всіх символів, що зустрічаються в початковому тексті, причому відома кількість появ кожного символу в ньому. Випишемо їх вертикально в ряд у вигляді графа (рис. 9.1). Виберемо два символи з найменшою кількістю повторень в тексті (якщо три або більше число символів мають однакові значення, вибираємо будь-які два з них). Проведемо від них лінії вліво до нової вершини графа і запишемо в неї значення, рівне сумі частот повторення кожного з об'єднуваних символів (рис. 9.1). Тепер не братимемо до уваги при пошуку найменших частот повторення два об'єднані вузли (для цього зітремо числа в цих двох вершинах), але розглядатимемо нову вершину з частотою появи, рівній сумі частот появи двох вершин, що з'єдналися. Повторюватимемо операцію об'єднання вершин до тих пір, поки не дійдемо однієї вершини з числом. Для перевірки: очевидно, що в ній буде записана довжина кодованого файлу. Тепер розставимо на двох ребрах графа, витікаючих з кож-

ної вершини, біти 0 і 1 довільно – наприклад, на кожному верхньому ребрі 0, а на кожному нижньому – 1. Тепер для визначення коду кожної конкретної літери необхідно просто пройти від вершини дерева до неї, виписуючи нулі і одиниці по маршруту проходження. Для рис. 9.1 символ "А" отримує код "000", символ "Б" – код "01", символ "К" – код "001", а символ "О" – код "1".

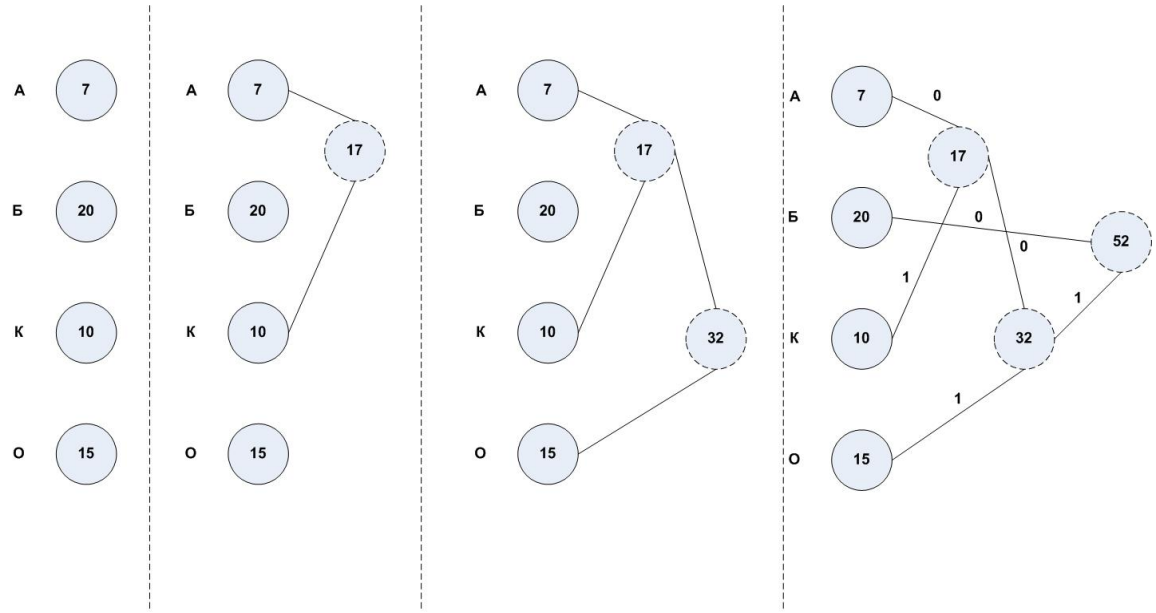


Рисунок 9.1 – Приклад отримання дерева Хаффмана

У теорії кодування інформації показується, що код Хаффмана є префіксним, тобто код жодного символу не є початком коду якого-небудь іншого символу. З цього виходить, що код Хаффмана однозначно відновлений одержувачем, навіть якщо не повідомляється довжина коду кожного переданого символу. Одержувачу пересилають лише дерево Хаффмана в компактному вигляді, а потім вхідна послідовність кодів символів декодується їм самостійно без якої-небудь додаткової інформації. Наприклад, при прийомі "0100010100001" їм спочатку відокремлюється перший символ "Б": "01-00010100001", потім знову починаючи з вершини дерева – "А" "01-000-10100001", потім аналогічно декодується весь запис "01-000-1-01-000-01" "БАОБАБ".

1.2. Алгоритм Лемпеля-Зіва

Класичний алгоритм Лемпеля-Зіва – LZ77, названий так по року своєї публікації, гранично простий. Він формулюється таким чином: "якщо в минулому раніше вихідному потоці вже зустрічалася подібна послідовність байт, причому запис про її довжину і зміщення від поточної позиції коротше ніж сама ця послідовність, то у вихідний файл записується посилання (зміщення, довжина), а не сама послідовність". Так фраза "КОЛОКОЛ_ОКОЛО_КОЛОКОЛЬНИ" закодується як "КОЛО(-4,3)_(-5,4)О_(-14,7)ЬНИ".

Поширений метод стиснення RLE (англ. Run Length Encoding), який полягає в записі замість послідовності однакових символів одного символу і їх кількості, є підкласом алгоритму LZ77. Розглянемо, наприклад, послідовність "AAAAAAA". За допомогою алгоритму RLE вона буде закодована як "(A,7)", у той же час її можна достатньо добре стиснути і за допомогою алгоритму LZ77: "A(-1,6)". Ступінь стиснення саме такої послідовності їм гірше (приблизно на 30-40%), але сам по собі алгоритм LZ77 більш універсальний, і може набагато краще обробляти послідовності взагалі нестискувані методом RLE.

2 Порядок виконання роботи

2.1. Одержати індивідуальний варіант завдання у керівника лабораторного практикуму.

2.2. З одержаного завдання видалити надмірну інформацію методом Хаффмана

2.3. Видалити надмірну інформацію методом Лемпеля – Зіва.

2.4. Порівняти отримані результати і зробити висновки про застосовність досліджуваних методів стиснення у криптосистемах.

3 Контрольні питання

1. Кодування інформації методом Хаффмана
2. Кодування інформації методом Лемпеля – Зіва
3. Кодування інформації методом RLE
4. Порівняльна характеристика методів стиснення даних
5. Способи підвищення ефективності реалізації алгоритму LZ77

ЛАБОРАТОРНА РОБОТА № 10

«Вивчення потокових шифрів»

Мета роботи: виконати дослідження апаратних і програмних способи реалізації скремблерів

1 Короткі теоретичні відомості

Залежно від розміру блоку інформації криптоалгоритми діляться на:

1. Потокові шифри. Одиницею кодування є один біт. Результат кодування не залежить від попередніх інформаційних біт. Схема застосовується в системах передачі потоків інформації, тобто в тих випадках, коли передача інформації починається і закінчується в довільні моменти часу і може випадково уриватися. Найбільш поширеними представниками потокових шифрів є скремблери.
2. Блочні шифри. Одиницею кодування є блок з декількох байтів (нині 4-32). Результат кодування залежить від всіх початкових байтів цього блоку. Схема застосовується при пакетній передачі інформації і кодуванні файлів.

Останнім часом сфера застосування скремблюючих алгоритмів значно скоротилася. Це пояснюється в першу чергу зниженням обсягів побітної послідовної передачі інформації, для захисту якої були розроблені такі алгоритми. Практично повсюдно в сучасних системах застосовуються мережі з комутацією пакетів, для підтримки конфіденційності якої використовуються блочні шифри. А їх криптостойкость перевершує, і деколи досить значно, криптостійкість скремблерів.

Суть скремблювання полягає в побітної зміні потоку даних, що проходить через систему. Практично єдиною операцією, використовуваною в скремблерах є XOR – що "побітне виключає АБО". Паралельно проходженню інформаційного потоку в скремблері за певним правилом генерується потік біт – що кодує потік.

Як пряме, так і зворотне шифрування здійснюється накладенням по XOR кодуючої послідовності на початкову.

Генерація кодуючої послідовності біт проводиться циклічно з невеликого початкового обсягу інформації – ключа по наступному алгоритму. З поточного набору біт вибираються значення декількох розрядів і складаються по XOR між собою. Всі розряди зсовуються на 1 біт, а тільки що набуте значення ("0" або "1") поміщається в наймолодший розряд, що звільнився. Значення, що знаходилося в самому старшому розряді до зсуву, додається в кодуючу послідовність, стаючи черговим її бітом (див. рис. 10.1).

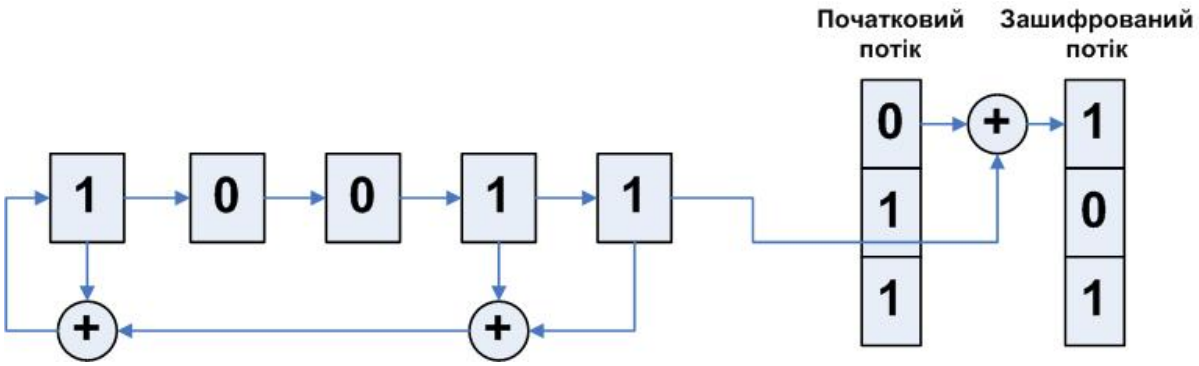


Рисунок 10.1 – Структура п'ятирозрядного скремблера

З теорії передачі даних криптографія запозичувала для запису подібних схем двійкову систему запису. По ній зображений на рис. 10.1 скремблер записується комбінацією "100112" – одиниці відповідають розрядам, з яких знімаються біти для формування зворотного зв'язку.

Розглянемо приклад кодування інформаційної послідовності 0101112 скремблером 1012 з початковим ключем 1102.

Стан ключа	Кодуючий біт		Інформаційний біт		Результат
110					
111	0	XOR	0	=	0
011	1	XOR	1	=	0
101	1	XOR	0	=	1
і т.д.					

Таким чином, устрій скремблера гранично просто. Його реалізація можлива як на електронній, так і на електричній базі, що і забезпечило його широке застосування в польових умовах. Той факт, що кожен біт вихідної послідовності залежить лише від одного вхідного біта, ще більш зміцнило становище скремблерів в захисті потокової передачі даних. Це пов'язано з тими, що неминуче виникають в каналі передачі перешкодами, які можуть спотворити в цьому випадку лише ті біти, на які вони припадають, а не пов'язану з ними групу байт, як це має місце в блочних шифрах.

Декодування скрембльованих послідовностей відбувається по тій же самій схемі, що і кодування. Саме для цього в алгоритмах застосовується результуюче кодування по тому, що "виключає АБО" – схема, однозначно відновлена при раскодуванні без яких-небудь додаткових обчислювальних витрат.

Головна проблема шифрів на основі скремблерів - синхронізація передавального (що кодує) і приймаючого (що декодує) пристроїв. При пропуску або помилковому вставлянні хоч би одного біта вся передавана інформація незворотно втрачається. Тому, в системах шифрування на основі скремблерів дуже велика увага приділяється методам синхронізації. На практиці для цих цілей звичайно застосовується комбінація двох методів:

а) додавання в потік інформації синхронізуючих бітів, напезаздалегідь відомих приймальній стороні, що дозволяє їй при незнаходженні такого біта активно розпочати пошук синхронізації з відправником;

б) використання високоточних генераторів часових імпульсів, що дозволяє в моменти втрати синхронізації проводити декодування бітів інформації, що приймаються, "по пам'яті" без синхронізації.

Число біт, охоплених зворотним зв'язком, тобто розрядність пристрою пам'яті, що містить кодуючу послідовність біт, називається розрядністю скремблера. Зображений на рис. 10.1 скремблер має розрядність 5. Відносно параметрів криптостойкости дана величина повністю ідентична довжині ключа блочних шифрів. Чим більша розрядність скремблера, тим вище криптостойкость системи, заснованої на його використанні.

При достатньо довгій роботі скремблера неминуче виникає його зациклення. Після виконання певного числа тактів створиться комбінація біт, яка вже одного разу виявлялася, і з цієї миті кодуюча послідовність почне циклічно повторюватися з фіксованим періодом. Ця проблема неусувна за своєю природою, оскільки в N розрядах скремблера не може перебувати більш 2^N комбінацій біт, і, отже, максимум, через, 2^N-1 цикл повтор комбінації обов'язково відбудеться. Комбінація "всі нулі" відразу ж виключається з ланцюжка графа станів скремблера – вона призводить скремблер до такого ж стану "всі нулі". Це вказує ще і на те, що ключ "всі нулі" непридатний для скремблера. Кожен біт, що генерується при зсуві, залежить лише від декількох біт тієї комбінації, що зберігається в даний момент скремблером. Тому після повторення деякої ситуації, що одного разу вже зустрічалася в скремблері, всі наступні за нею будуть в точності повторювати ланцюжок, що вже пройшов раніше в скремблері.

Можливі різні типи графів стану скремблера. На рис. 10.2 приведені зразкові варіанти для 3-розрядного скремблера. На рис. 10.2.а крім завжди присутнього ци-

клу "000">>"000" показані ще два цикли – з 3-мя станами і 4-мя. На рис. 10.2.б показаний ланцюжок, який сходиться до циклу з 3-х станів і вже ніколи звідти не виходить. На рис. 10.2.у всі можливі стани крім нульового, об'єднані в один замкнутий цикл. Очевидно, що саме в цьому випадку, коли всі 2^N-1 стан системи утворюють цикл, період повторення вихідних комбінацій максимальний, а кореляція між довжиною циклу і початковим станом скремблера (ключем), яка привела б до появи слабіших ключів, відсутня.

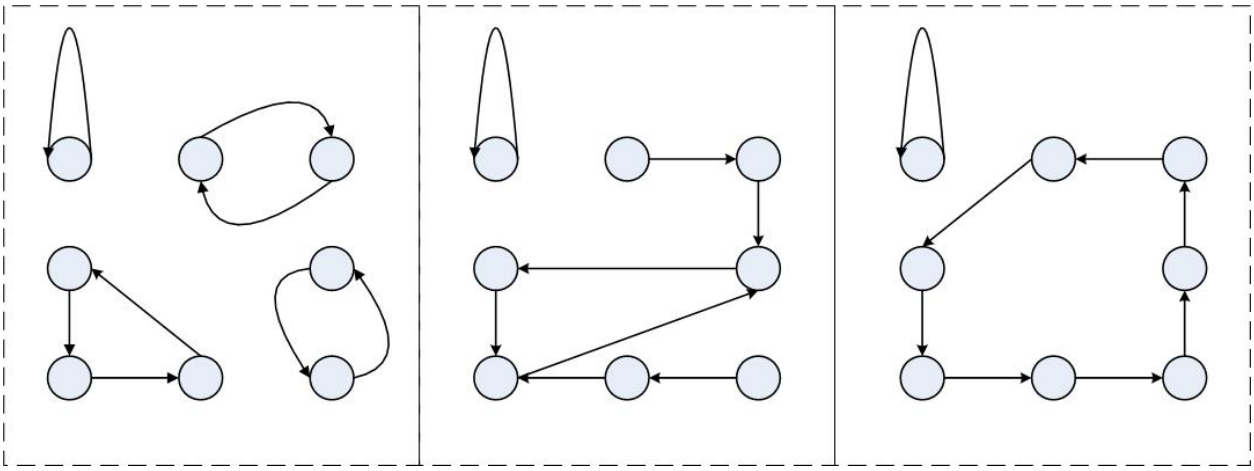


Рисунок 10.2 – Графи, що характеризують стан скремблера

Доведено, що для скремблера будь-якої розрядності N завжди існує такий вибір охоплюваних зворотним зв'язком розрядів, що послідовність біт, що генерується ними, матиме період, рівний 2^N-1 бітів. Так, наприклад, в 8-бітовому скремблері, при обхваті 0-го, 1-го, 6-го і 7-го розрядів дійсно за час генерації 255 біт послідовно минають всі числа від 1 до 255, не повторюючись жодного разу.

Схеми з вибраними по цьому закону зворотними зв'язками називаються генераторами послідовностей найбільшої довжини (ПНД), і саме вони використовуються в скремблюючей апаратурі. З безлічі генераторів ПНД заданої розрядності в часи, коли вони реалізовувалися на електричній або мінімальній електронній

базі вибиралися ті, у яких число розрядів, що беруть участь в створенні чергового біта, було мінімальним. Звичайно генератора ПНД вдавалося досягти за 3 або 4 зв'язки. Сама ж розрядність скремблерів перевищувала 30 біт, що давало можливість передавати до 240 біт = 100 Мбайт інформації без побоювання повторення кодуючої послідовності.

ПНД нерозривно пов'язані з математичною теорією поліномів, що неприводяться. Достатньо щоб поліном ступеня N не був представимо по модулю 2 у вигляді добутку ніяких інших поліномів, для того, щоб скремблер, побудований на його основі, створював ПНД. Наприклад, єдиним поліномом ступеня 3, що неприводиться, є x^3+x+1, у двійковому вигляді він записується як 1011_2 (одиниці відповідають присутнім розрядам). Скремблери на основі поліномів, що неприводяться, утворюються відкиданням самого старшого розряду (він завжди присутній, а, отже, несе інформацію тільки про ступінь полінома), так на основі вказаного полінома, ми можемо створити скремблер 011_2 з періодом зациклення $7(=2^3-1)$. На практиці застосовуються поліноми значно вищих порядків. А таблиці поліномів будь-яких порядків, що неприводяться, можна завжди знайти в спеціалізованих математичних довідниках.

Істотним недоліком скремблерів є їх нестійкість до фальсифікації.

2 Порядок виконання роботи

2.1. Перекласти свій номер по журналу обліку відвідуваної академічної групи в двійковий код.

2.2. При необхідності доповнити одержаний код до 5 розрядів шляхом додавання в старші розряди логічних 0.

2.3. Одержаний код використовувати як початкове значення ключа скремблера.

2.4. У програмі Electronics Workbench зібрати і провести дослідження п'ятирозрядного скремблера, використовуючи початкове значення ключа, одержане в п.п. 2.3.

2.5. Розробити схему скремблера, що виключає його зациклення при комбінації ключа 0 0 0 0 0.

2.6. Випробувати розроблену схему для дескремблювання зашифрованих даних.

2.7. Скласти алгоритм і розробити програмний код, що реалізовує функції скремблера і дескремблера.

2.8. Виконати тестування розробленого програмного забезпечення на початкових даних, одержаних в п.п. 2.3.

2.9. Провести порівняльний аналіз апаратного і програмного способів реалізації скремблера.

3 Контрольні питання

1. Принцип функціонування скремблера
2. Елементарні операції, використовувані для реалізації криптоалгоритмов
3. Способи підвищення криптостойкости скремблювання
4. Чому скремблери можливо використовувати для дескремблювання зашифрованих даних.

ПЕРЕЛІК РЕКОМЕНДОВАНИХ ДЖЕРЕЛ

1. Дж. Л. Месси. Введение в современную криптологию. // ТИИЭР, т.76, №5, Май 88 – М, Мир, 1988, С.24-42.
2. У. Диффи. Первые десять лет криптографии с открытым ключом. // ТИИЭР, т.76, №5, Май 88 – М, Мир, 1988, С.54-74.
3. А. В. Спесивцев и др. Защита информации в персональных компьютерах. – М., Радио и связь. 1992, С.140-149.
4. В. Жельников. Криптография от папируса до компьютера. – М., ABF, 1996.
5. Вирт Н. Алгоритмы + структуры данных = программы. М.: Мир, 1985.
6. Галлагер Р. Теория информации и надежная связь. М.: Советское радио, 1974.
7. Дэвис Д., Барбер Д., Прайс У., Соломонидес С. Вычислительные сети и сетевые протоколы. М.:Мир, 1982.
8. Кнут Д. Искусство программирования для ЭВМ. Т. 3. Сортировка и поиск. М.: Мир, 1978.
9. Баричев С. Криптография без секретов. М.: "ДИАЛОГ-МИФИ", - 1995.
10. Вакка Дж. Секреты безопасности в Internet. – К.: Диалектика, 1997.
11. http://www.citforum.ru/security/
12. Кулаков Ю.А., Луцкий Г.М. Компьютерные сети.-К. Юниор, 1998.-384с.
13. Таненбаум Э. Компьютерные сети. - СПб.: Питер, 2002. - 848 с.: ил.
14. Williams Ross N. Элементарное руководство по CRC - алгоритмам обнаружения ошибок ftp://www.internode.net.au/clients/rocksoft/papers/crc_v3. txt
15. Романец Ю.В., Тимофеев П.А., Шаньгин В.Ф. Защита информации в компьютерных системах и сетях/Под ред. В.Ф.Шаньгина.-М.: Радио и связь, 1999.-328с.

16. Мельников В.В. Защита информации в компьютерных системах. -М.: Финансы и статистика; Электроинформ, 1997.-368с.
17. Ведев Д Л. Защита данных в компьютерных сетях, http://www.osp.ru.
18. Решения компании Cisco Systems по безопасности компьютерных систем - Cisco SAFE http://www.cisco.com/warp/public/779/largeent/issues/security/safe.htm 1
19. PGP - лучший криптографический пакет - http://www.ukr.net/book/g6pgp.htm
20. Сергеев Л. Компьютерные вирусы: вчера, сегодня, завтра. "Мой компьютерный журнал" http ://www. compulog.ru/windows/m cj/public/virus/a3- l.html
21. Рули Д. и др. Сети Windows NT 4.0.-K.: Издательская группа BHV, 1998, - 800с.