Міністерство освіти і науки України
Центральноукраїнський національний технічний університет
Механіко-технологічний факультет
Кафедра кібербезпеки та програмного забезпечення

# Інформаційна безпека в комп'ютерних мережах

*Методичні вказівки до виконання лабораторних робіт для студентів денної форми навчання за спеціальністю 123 "Комп'ютерна інженерія", 122 «Комп'ютерні науки»*

ЗАТВЕРДЖЕНО
на засіданні кафедри кібербезпеки та програмного забезпечення, протокол № 1 від 05.07.2017.

Кропивницький
2017

Інформаційна безпека в комп’ютерних мережах: методичні вказівки до виконання лабораторних робіт для студентів за спеціальністю. 123 “Комп’ютерна інженерія”; 122 “Комп’ютерні науки”/ М-во освіти і науки України, Центральноукр. нац. техн. ун-т; [уклад. О. А. Смірнов, О. К. Конопліцька-Слободенюк, В.Д. Хох, С.А. Смірнов] – Кропивницький: ЦНТУ, 2017. – 69 с.

Укладачі: Смірнов О. А., докт. техн. наук, професор;
Конопліцька-Слободенюк О. К., викладач;
В.Д. Хох, аспірант;
С.А. Смірнов – ст. викладач.

Рецензенти: Сидоренко В. В., докт. техн. наук, професор;
Доренський О. П., канд. техн. наук.

**ЗМІСТ**

# ВСТУП

**Мета:** Основною метою є теоретична та практична підготовка студентів щодо вивчення теоретичних підвалин теорії інформаційної безпеки в комп'ютерних мережах та їх практичних застосувань.

**Завдання:**

– Вивчення теоретичних основ і положень теорії інформаційної безпеки в комп'ютерних мережах.

– Вивчення способів забезпечення інформаційної безпеки в комп'ютерних мережах.

– Отримання необхідних теоретичних знань побудови систем захисту інформації в комп'ютерних мережах.

– Отримання практичних навиків адміністрування систем захисту інформації в комп'ютерних мережах.

У результаті вивчення навчальної дисципліни студент повинен:

– знати: Загальні відомості про атаки на програмне забезпечення та дані у комп'ютерних системах та мережах; Міжмережні екрани (фаєрволи, брандмауери); Віртуальні приватні мережі (VPN); Технології тунелювання; Архітектуру безпеки для IP (IPSec); Протокол SSL/TLS; Безпеку бездротових з'єднань; Системи виявлення вторгнень (IDS-системи); Системи протидії вторгненням (IPS-системи); Реалізацію мережної безпеки у організаціях; Безпеку банківських електронних платіжних систем; Електронну комерцію: вимоги до безпеки;

– вміти: Програмно реалізовувати наступні проекти: мережного антивірусу; міжмережного екрану; сніффера; протоколу IPSec; протоколу TLS/SSL; системи виявлення вторгнень

Структурно логічна схема підготовки спеціаліста, магістра.

Враховуючи послідовність накопичення знань та інформації, дисципліна вивчається після викладання наступних дисциплін:

– Вища математика.

– Теорія ймовірності та математична статистика.

– Алгоритми та методи обчислень.

– Програмування.

– Організація баз даних.

– Інженерія програмного забезпечення.

– Системне програмне забезпечення.

Для опанування матеріалу дисципліни «Інформаційна безпека в комп'ютерних мережах» окрім лекційних та лабораторних занять, тобто аудиторного навантаження, значна увага приділяється самостійній роботі.

До основних видів самостійної роботи студента відносимо:

1. Вивчення лекційного матеріалу.

2. Робота з літературними джерелами.

3. Розв'язання практичних задач за індивідуальними варіантами.

4. Підготовка до модульних, підсумкового контролю, екзамену (денна та заочна).

5. Виконання курсової роботи для денної форми навчання.

Для підвищення рейтингу впродовж семестру студент може виконати згідно запропонованої викладачем теми самостійну роботу, обсяг якої складає не менше 10 сторінок.

Провідна форма навчання – лекція. Лекція дозволяє дуже економно, з мінімальними затратами часу і викладача, і студентів, надати великий обсяг інформації по темі, що розглядається. За характером логіки пізнання впроваджуються аналітичний, індуктивний та дедуктивний методи.

Супровідні методи – лабораторні роботи.

Основна дидактична мета практичного заняття – закріплення й деталізація знань, а головне – формування навичок і вмінь. Для проведення практичного заняття викладач готує відповідні методичні матеріали: тести для виявлення рівня оволодіння необхідними теоретичними положеннями ; набір

практичних завдань різної складності для розв'язування їх на занятті та дидактичні засоби.

**Шкала оцінювання: національна та ECTS**

| Сума балів за всі види навчальної діяльності | Оцінка ECTS | Оцінка за національною шкалою |
|---|---|---|
| | | для екзамену |
| 90 – 100 | **A** | відмінно |
| 82-89 | **B** | добре |
| 74-81 | **C** | |
| 64-73 | **D** | задовільно |
| 60-63 | **E** | |
| 35-59 | **FX** | незадовільно з можливістю повторного складання |
| 0-34 | **F** | незадовільно з обов’язковим повторним вивченням дисципліни |

Вибравши предметну область, над якою ви будете працювати, ви повинні виконати завдання до лабораторних робіт, а також відповісти на питання в кінці кожної лабораторної роботи. Звіт повинен містити хід виконання завдань а також графічні матеріали, що підтверджують виконання цих завдань.

## Лабораторна робота №1 Реалізація мережного антивірусу

**Ціль роботи: розібратися з будовою сучасних антивірусів і реалізувати свій антивірус.**

### Теоретична частина

***Антивірусна програма (антивірус)*** – спеціалізована програма для виявлення комп'ютерних вірусів, а також небажаних (які вважаються шкідливими) програм взагалі й відновлення заражених (модифікованих) такими програмами файлів, а також для профілактики – запобігання зараження (модифікації) файлів або операційної системи шкідливим кодом.

На даний момент антивірусне програмне забезпечення розробляється, в основному, для ОС сімейства Windows від компанії Microsoft. Це викликано великою кількістю шкідливих програм саме під цю платформу (а це, у свою чергу, викликано великою популярністю цієї ОС, так само, як і більшою кількістю засобів розробки, у тому числі безкоштовних і навіть «інструкцій з написання вірусів»). У даний момент на ринок виходять продукти й для інших операційних систем, таких, приміром, як Linux і Mac OS X. Це викликано початком поширення комп'ютерних вірусів і під ці платформи, хоча UNIX-подібні системи традиційно користуються репутацією більше стійких до впливу шкідливих програм.

Крім ОС для настільних комп'ютерів і ноутбуків, також існують платформи й для мобільних пристроїв, такі, як Apple iOS, BlackBerry, Android, Windows Phone і ін. Користувачі пристроїв на даних ОС також піддані ризику зараження шкідливим програмним забезпеченням, тому деякі розроблювачі антивірусних програм випускають продукти й для таких пристроїв.

### Робота антивірусу

Говорячи про системи Майкрософт, варто знати, що звичайно антивірус діє за схемою:

– пошук у базі даних антивірусного ПЗ сигнатур вірусів;

– якщо знайдено інфікований код у пам'яті (оперативній й/або постійній), запускається процес «карантину», і процес блокується;

– зареєстрована програма звичайно видаляє вірус, незареєстрованна просить реєстрації й залишає систему уразливою.

**Бази антивірусів**

Для використання антивірусів необхідні постійні відновлення так званих баз антивірусів. Вони являють собою інформацію про віруси – як їх знайти й знешкодити. Оскільки віруси пишуть часто, то необхідний постійний моніторинг активності вірусів у мережі. Для цього існують спеціальні мережі, які збирають відповідну інформацію. Після збору цієї інформації виробляється аналіз шкідливості вірусу, аналізується його код, поводження, і після цього встановлюються способи боротьби з ним. Найчастіше віруси запускаються разом з операційною системою. У такому випадку можна просто видалити рядок запуску вірусу з реєстру, і на цьому в простому випадку процес може закінчитися. Більш складні віруси використовують можливість зараження файлів. Наприклад, відомі випадки, як якісь навіть антивірусні програми, будучи зараженими, самі ставали причиною зараження інших чистих програм і файлів. Тому більше сучасні антивіруси мають можливість захисту своїх файлів від зміни й перевіряють їх на цілісність по спеціальному алгоритмі. Таким чином, віруси ускладнилися, як і ускладнилися способи боротьби з ними. Зараз можна побачити віруси, які займають уже не десятки кілобайт, а сотні, а часом можуть бути й розміром у парі мегабайт. Звичайно такі віруси пишуть у мовах програмування більше високого рівня, тому їх легше зупинити. Але як і раніше існує погроза від вірусів, написаних на низькорівневих машинних кодах на зразок асемблера. Складні віруси заражають операційну систему, після чого вона стає уразливою й неробочою. На жаль, за прогнозами, у найближчому майбутньому робота антивірусних компаній сильно ускладниться у зв'язку з тим, що будуть сильніше поширюватися віруси із захистом від копіювання.

**Функції:**

– Базовий захист:

– Захист від вірусів, троянських програм і хробаків.

– Захист декількох пристроїв.

– Перевірка посилань.

– Регулярні перевірки.

– Захист від мережних атак.

– Контроль активності програм і захист від блокерів.

– Захист від шпигунських і рекламних програм.

– Перевірка файлів в автоматичному режимі й на вимогу.

– Перевірка поштових повідомлень (для будь-яких поштових клієнтів).

– Перевірка інтернет-трафіку (для будь-яких інтернет-браузерів).

– Захист інтернет-пейджерів (ICQ, MSN).

– Проактивний захист від нових шкідливих програм.

– Перевірка Java– і Visual Basic-скриптів.

– Захист від схованих битих посилань.

– Постійна перевірка файлів в автономному режимі.

– Постійний захист від фішингових сайтів.

– Захист від злому веб-камери.

– Синхронізація паролів на всіх пристроях.

– Резервне копіювання й шифрування даних.

– Батьківський контроль.

Запобігання погроз:

– Пошук уразливостей в ОС і встановленому ПЗ.

– Аналіз і усунення уразливостей у браузері.

– Блокування посилань на заражені сайти.

– Розпізнавання вірусів за способом їхнього упакування.

– Глобальний моніторинг погроз.

Відновлення системи й даних:

– Можливість установки програми на заражений комп'ютер.

– Функція самозахисту програми від вимикання або зупинки.

– Відновлення коректних налаштувань системи після видалення шкідливого ПЗ.

– Наявність інструментів для створення диска аварійного відновлення.

Захист конфіденційних даних:

– Блокування посилань на фішингові сайти.

– Захист від всіх видів кейлоггерів.

Зручність використання:

– Автоматичне налаштування програми в процесі установки.

– Готові рішення (для типових проблем).

– Наочне відображення результатів роботи програми.

– Інформативні діалогові вікна для прийняття користувачем обґрунтованих рішень.

– Можливість вибору між простим (автоматичним) і інтерактивним режимами роботи.

– Цілодобова технічна підтримка.

– Автоматичне відновлення баз.

**Завдання:**

Реалізувати мережний антивірус. Розроблене програмне забезпечення повинне включати базу даних сигнатур, пошук по сигнатурах і реалізовувати 5-6 функцій антивірусу з перерахованих у теоретичній частині.

## Лабораторна робота №2 Реалізація міжмережного екрану

**Ціль роботи: Розібратися з будовою міжмережних екранів й реалізувати свій міжмережний екран**.

### Теоретична частина

Міжмережний екран являє собою комплекс завдань щодо запобігання несанкціонованого доступу, пошкодження або викрадення даних, або іншого негативного впливу, який може вплинути на працездатність мережі.

Міжмережний екран, його також називають фаєрвол (Від англ. Firewall) або брандмауер на шлюзі дозволяє забезпечити безпечний доступ користувачів до мережі Інтернет, при цьому захищаючи віддалене підключення до внутрішніх ресурсів. Міжмережний екран переглядає через себе весь трафік, що проходить між сегментами мережі, і для кожного пакета реалізує рішення – пропускати або не пропускати. Гнучка система правил брандмауера дозволяє забороняти або дозволяти з'єднання за численними параметрами: адрес, мереж, протоколам і портів.

***Методи контролю трафіку між локальної та зовнішньою мережею***

– Фільтрація пакетів. Залежно від того чи задовольняє вхідний пакет зазначеним у фільтрах умовами він пропускається в мережу або відкидається.

– Stateful inspection. У цьому випадку здійснюється інспектування вхідного трафіку – один з найбільш передових способів реалізації Firewall. Під інспекцією мається на увазі аналіз не всього пакету, а лише його спеціальної ключової частини і в порівнянні із заздалегідь відомими значеннями з бази даних дозволених ресурсів. Такий метод забезпечує найбільшу продуктивність роботи Firewall і найменші затримки.

– Proxy-сервер. У даному випадку між локальної та зовнішньої мережами встановлюється додатковий пристрій proxy-сервер, який служить «воротами», через який повинен проходити весь вхідний і вихідний трафік.

Міжмережний екран дозволяє настроювати фільтри, які відповідають за пропуск трафіку по:

– IP-адресі. Задавши якусь адресу або певний діапазон можна заборонити отримувати з них пакети, або навпаки дозволити доступ тільки з даних IP адрес.

– Порту. Фаєрвол може налаштувати точки доступу додатків до послуг мережі. Приміром, ftp використовує порт 21, а додатки для перегляду web-сторінок порт 80.

– Протоколу. Брандмауер може бути налаштований на пропуск даних лише якогось одного протоколу, або заборонити доступ з його використанням. Найчастіше тип протоколу може говорити про виконуваних завданнях, використовуваного ним програми та про набір параметрів захисту. У зв'язку з цим, доступ може бути налаштований тільки для роботи якого-небудь одного специфічного додатки і запобігти потенційно небезпечний доступ з використанням всіх інших протоколів.

– Доменному імені. У даному випадку фільтр забороняє або дозволяє з'єднання конкретних ресурсів. Це дозволяє заборонити доступ з небажаних сервісів і додатків мережі, або навпаки дозволити доступ тільки до них.

Для установки можуть застосовуватися й інші параметри для фільтрів, характерні для даної конкретної мережі, залежно від виконуваних у ній завдань.

Найчастіше міжмережний екран використовується в комплексі з іншими засобами захисту, наприклад, антивірусне програмне забезпечення.

**Принцип дії брандмауера**

Брандмауер може бути виконаний:

– Апаратно. У такому випадку в ролі апаратного фаєрвола виступає маршрутизатор, який розташовується між комп'ютером і мережею Інтернет. До фаєрвол може бути підключено кілька ПК і при цьому всі вони будуть захищені фаєрволом, який виступає частиною маршрутизатора.

– Програмно. Найбільш поширений тип брандмауера, який представляють собою спеціалізоване програмне забезпечення, яке користувач встановлює на свій ПК.

Навіть якщо підключений маршрутизатор з вбудованим міжмережним екраном, додатково може бути встановлений програмний фаєрвол на кожен комп'ютер окремо. У такому випадку зловмисникові буде складніше проникнути в систему.

**Завдання:**

Реалізувати міжмережний екран. Розроблене програмне забезпечення повинно включати функції перераховані у теоретичній частині.

Міжмережний екран повинен настроювати фільтри, які відповідають за пропуск трафіку по:

– IP-адресі.

– Порту.

– Протоколу.

– Доменному імені.

## Лабораторна робота №3 Реалізація сніффера

**Ціль роботи: Розібратися з будовою сніффера й реалізувати свій сніффер.**

### Теоретична частина

***Аналізатор трафіку, або сніффер*** (від англ. to sniff – нюхати) – мережний аналізатор трафіку, програма або програмно-апаратний пристрій, призначений для перехоплення й наступного аналізу, або тільки аналізу мережного трафіку, призначеного для інших вузлів.

Сніффер може аналізувати тільки те, що проходить через його мережну карту. Усередині одного сегмента мережі Ethernet всі пакети розсилаються всім машинам, через цього можливо перехоплювати чужу інформацію. Використання комутаторів (switch, switch-hub) і їхня грамотна конфігурація вже є захистом від прослуховування. Між сегментами інформація передається через комутатори. Комутація пакетів – форма передачі, при якій дані, розбиті на окремі пакети, можуть пересилатися з вихідного пункту в пункт призначення різними маршрутами. Тому якщо хтось в іншому сегменті посилає всередині нього які-небудь пакети, то у ваш сегмент комутатор ці дані не відправить.

***Перехоплення трафіку може здійснюватися:***

– звичайним «прослуховуванням» мережного інтерфейсу (метод ефективний при використанні в сегменті концентраторів (хабів) замість комутаторів (світчів), у противному випадку метод малоефективний, оскільки на сніффер попадають лише окремі фрейми);

– підключенням сніффера в розрив каналу;

– відгалуженням (програмним або апаратним) трафіку й напрямком його копії на сніффер (Network tap);

– через аналіз побічних електромагнітних випромінювань і відновлення в такий спосіб що прослуховується трафіку;

– через атаку на канальному (2) (MAC-spoofing) або мережному (3) рівні (IP-spoofing), що приводить до перенапрямку трафіку жертви або всього трафіку сегмента на сніффер з наступним поверненням трафіку в належну адресу.

Сніффери застосовуються як у конструктивних, так і в деструктивних цілях. Аналіз трафіку, який пройшов через сніффер, дозволяє:

– Виявити паразитний, вірусний і закільцьований трафік, наявність якого збільшує завантаження мережного встаткування й каналів зв'язку (сніффери тут малоефективні; як правило, для цих цілей використовують збір різноманітної статистики серверами й активним мережним устаткуванням і її наступний аналіз).

– Виявити в мережі шкідливе й несанкціоноване ПЗ, наприклад, мережні сканери, флудери, троянські програми, клієнти пірінгових мереж і інші (це звичайно роблять за допомогою спеціалізованих сніфферів – моніторів мережної активності).

– Перехопити будь-який незашифрований (а часом і зашифрований) користувальницький трафік з метою одержання паролів і іншої інформації.

– Локалізувати несправність мережі або помилку конфігурації мережних агентів (для цієї цілі сніффери часто застосовуються системними адміністраторами)

Оскільки в «класичному» сніффері аналіз трафіку відбувається вручну, із застосуванням лише найпростіших засобів автоматизації (аналіз протоколів, відновлення TCP-потоку), то він підходить для аналізу лише невеликих його обсягів.

Знизити погрозу сніффингу пакетів можна за допомогою таких засобів, як:

– Автентифікація.

– Криптографія.

– Антисніффери.

Інфраструктура, що комутується.

**Приклад простого сніффера під Windows**

**(дані зі сторінки http://habrahabr.ru/post/164901/)**

**Ціль:** написати програму, що буде захоплювати мережний трафік (Ethernet, WiFi), що передається за протоколом IP.

**Засоби:** Visual Studio 2005 або вище.

**Теорія**

У цей момент переважна більшість сучасних інформаційних мереж базуються на фундаменті стека протоколів TCP/IP. Стек протоколів TCP/IP (англ. Transmission Control Protocol/Internet Protocol) – збірна назва для мережних протоколів різних рівнів, використовуваних у мережах. IP – маршрутизуємий мережний протокол, використовуваний для негарантованої доставки даних, поділюваних на так звані пакети (більше вірний термін – дейтаграма) від одного вузла мережі до іншого.

Особливий інтерес для нас представляють IP-пакети, призначені для передачі інформації. Це досить високий рівень мережний OSI-моделі даних, коли можна абстрагуватися від пристрою й середовища передачі даних, оперуючи лише логічним поданням.

Зовсім логічною є та обставина, що рано або пізно повинні були з'явиться інструменти для перехоплення, контролю, обліку й аналізу мережного трафіку. Такі засоби звичайно називаються аналізаторами трафіку, пакетними аналізаторами або сніфферами (від англ. to sniff – нюхати). Це – мережний аналізатор трафіку, програма або програмно-апаратний пристрій, призначений для перехоплення й наступного аналізу, або тільки аналізу мережного трафіку, призначеного для інших вузлів.

**Практика**

На даний момент створено досить багато програмного забезпечення для прослуховування трафіку. Найбільш відомий з них: Wireshark. Нас цікавить завдання перехоплення трафіку методом звичайного «прослуховування» мережного інтерфейсу. Важливо розуміти, що ми не збираємося займатися

зломом і перехоплювати чужий трафік. Потрібно всього лише переглядати й аналізувати трафік, що проходить через наш хост.

*Для чого це може знадобитися:*

– Дивитися поточний потік трафіку через мережне з'єднання (вхідний/вихідний/усього).

– Перенаправляти трафік для наступного аналізу на інший хост.

– Теоретично, можна спробувати застосувати його для злому WiFi-мережі.

На відміну від Wireshark, що базується на бібліотеці libpcap/WinPcap, наш аналізатор не буде використовувати цей драйвер. У нас взагалі не буде драйвера, і свій NDIS ми писати не збираємося. Про це можна прочитати в цьому топіці. Він буде просто пасивним спостерігачем, що використовує тільки бібліотеку WinSock. Використання драйвера в цьому випадку непотрібно.

Ключовим кроком у перетворенні простого мережного додатка в мережний аналізатор є перемикання мережного інтерфейсу в режим прослуховування (promiscuous mode), що й дозволить йому одержувати пакети, адресовані іншим інтерфейсам у мережі. Цей режим змушує мережну плату приймати всі кадри, поза залежністю від того, кому вони адресовані в мережі.

Починаючи з Windows 2000 (NT 5.0) створити програму для прослуховування сегмента мережі стало дуже просто, тому що її мережний драйвер дозволяє перевести сокет у режим прийому всіх пакетів.

***Включення нерозбірливого режиму***

```
long flag = 1;
SOCKET socket;
#define SIO_RCVALL 0x98000001

ioctlsocket(socket, SIO_RCVALL, &RS_Flag);
```

Наша програма оперує IP-пакетами, і використовує бібліотеку Windows Sockets версії 2.2 і «сирі» сокети (raw sockets). Для того щоб одержати прямий доступ до IP-пакета, сокет потрібно створювати в такий спосіб:

Створення сирого сокета:

```
s = socket(AF_INET, SOCK_RAW, IPPROTO_IP);
```

Тут замість константи SOCK_STREAM (протокол TCP) або SOCK_DGRAM (протокол UDP), ми використовуємо значення SOCK_RAW. Загалом кажучи, робота з raw sockets цікава не тільки з погляду захвата трафіку. Фактично, ми одержуємо повний контроль за формуванням пакета. Вірніше, формуємо його вручну, що дозволяє, наприклад, послати специфічний ICMP-пакет...

Відомо, що IP-пакет складається із заголовка, службової інформації й, властиво, даних. Раджу заглянути сюди, щоб освіжить знання. Опишемо у вигляді структури IP-заголовок (спасибі відмінній статті на RSDN [3]):

***Опис структури IP-пакета***

```
typedef struct _IPHeader
{
  unsigned char  ver_len;          // версія й довжина заголовку
  unsigned char  tos;                    // тип сервісу
  unsigned short length;                 // довжина всього пакета
  unsigned short id;                     // Id
  unsigned short flgs_offset;            // прапори й зсув
  unsigned char  ttl;                    // час життя
  unsigned char  protocol;               // протокол
  unsigned short xsum;                   // контрольна сума
  unsigned long  src;                    // IP-адреса відправника
  unsigned long  dest;                   // IP-адреса призначення
```

```
  unsigned short *params;              // параметри (до 320 біт)
  unsigned char  *data;                // дані (до 65535 октетів)
}IPHeader;
```

Головна функція алгоритму прослуховування буде виглядати в такий спосіб:

Функція захвату одного пакета

```
IPHeader* RS_Sniff()
{
   IPHeader *hdr;
   int count = 0;
   count = recv(RS_SSocket, (char*)&RS_Buffer[0], sizeof(RS_Buffer), 0);
   if (count >= sizeof(IPHeader))
   {
      hdr = (LPIPHeader)malloc(MAX_PACKET_SIZE);
      memcpy(hdr, RS_Buffer, MAX_PACKET_SIZE);
      RS_UpdateNetStat(count, hdr);
      return hdr;
   }
   else
      return 0;
}
```

Тут все просто: одержуємо порцію даних за допомогою стандартної функції socket-функції recv, а потім копіюємо їх у структуру типу IPHeader.

Захоплюємо всі пакети, які потраплять на наш мережний інтерфейс

```
while (true)
{
```

```
    IPHeader* hdr = RS_Sniff();
    // обробка IP-пакета
    if (hdr)
    {
       // друкуємо заголовок у консолі
    }
}
```

Тут і далі в деяких важливих функцій і змінних автор зробив префкис RS_ (від Raw Sockets).

У принципі, можна піти далі, і описати заголовки всіх наступних протоколів, що перебувають вище. Для цього необхідно аналізувати поле protocol у структурі IPHeader.

Подивитесь на приклад коду (так, там повинен бути switch), де відбувається розфарбовування заголовка залежно від того, який протокол має пакет, інкапсульований в IP:

```
/*
*  Виділення пакета кольором
*/
void ColorPacket(const IPHeader *h, const u_long haddr, const u_long whost = 0)
{
   if (h-h->xsum)
      SetConsoleTextColor(0x17); // якщо пакет не порожній
   else
      SetConsoleTextColor(0x07); // порожній пакет

   if (haddr == h-h->src)
   {
```

```
            SetConsoleTextColor(BACKGROUND_BLUE                    |
/*BACKGROUND_INTENSITY |*/
                FOREGROUND_RED | FOREGROUND_INTENSITY); // "рідний"
пакет на віддачу
        }
        else if (haddr == h-h->dest)
        {
            SetConsoleTextColor(BACKGROUND_BLUE                    |
/*BACKGROUND_INTENSITY |*/
                FOREGROUND_GREEN | FOREGROUND_INTENSITY); //
"рідний" пакет на прийом
        }

        if (h-h->protocol == PROT_ICMP || h-h->protocol == PROT_IGMP)

        {
            SetConsoleTextColor(0x70); // ICMP-пакет
        }
        else if(h-h->protocol == PROT_IP || h-h->protocol == 115)
        {
            SetConsoleTextColor(0x4F); // in-IP-пакет, L2TP
        }
        else if(h-h->protocol == 53 || h-h->protocol == 56)
        {
            SetConsoleTextColor(0x4C); // TLS, IP with Encryption
        }

        if(whost == h-h->dest || whost == h-h->src)
        {
            SetConsoleTextColor(0x0A);
```

```
    }
}
```

Для нашого навчального приклада цілком достатньо буде подивитися ip-адреси хостів, з яких і на які йде трафік, і порахувати його кількість в одиницю часу.

Для того, щоб відобразити дані IP-заголовка, необхідно реалізувати функцію перетворення заголовка (але не даних) дейтаграми в рядок. Як приклад реалізації, можна запропонувати такий варіант:

***Перетворення IP-заголовка в рядок***

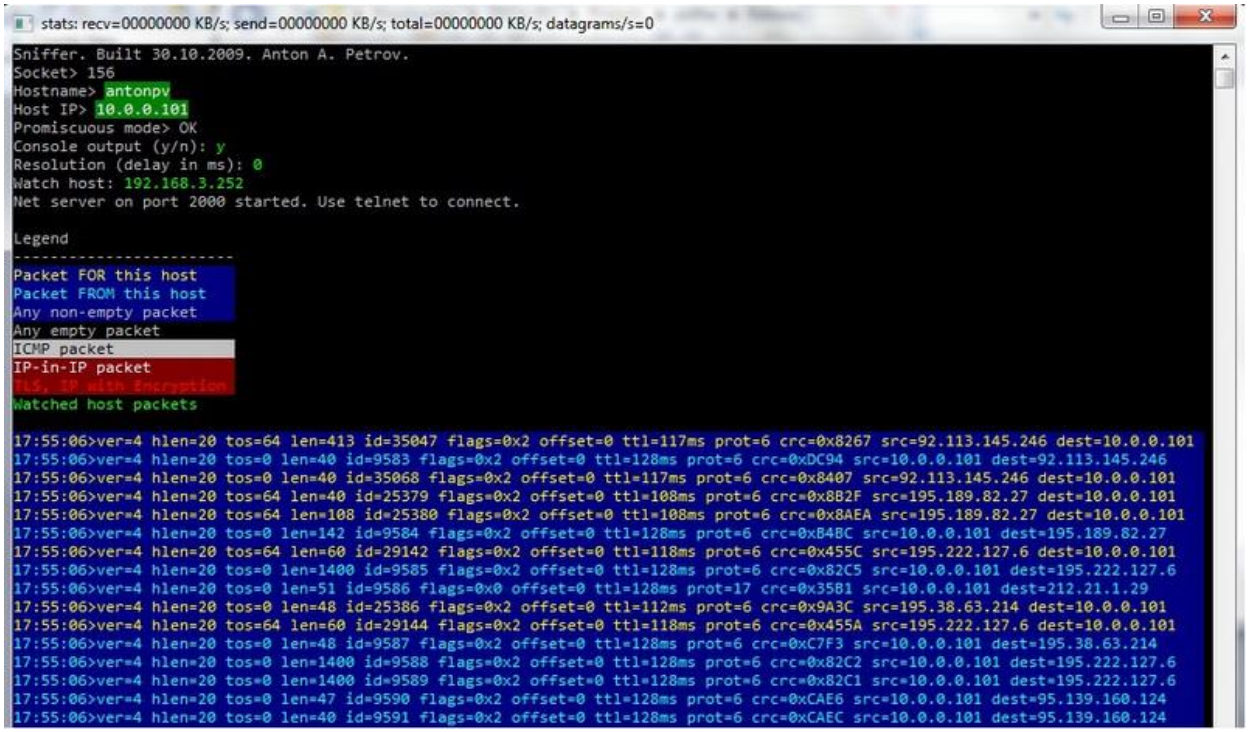
```
inline char* iph2str(IPHeader *iph)
{
    const int BUF_SIZE = 1024;
    char *r = (char*)malloc(BUF_SIZE);
    memset((void*)r, 0, BUF_SIZE);

    sprintf(r, "ver=%d hlen=%d tos=%d len=%d id=%d flags=0x%X offset=%d ttl=%dms prot=%d crc=0x%X src=%s dest=%s",

        BYTE_H(iph->ver_len),
        BYTE_L(iph->ver_len)*4,
        iph->tos,
        ntohs(iph->length),
        ntohs(iph->id),
        IP_FLAGS(ntohs(iph->flgs_offset)),
        IP_OFFSET(ntohs(iph->flgs_offset)),
        iph->ttl, iph->protocol,
```

```
            ntohs(iph->xsum), nethost2str(iph->src),
            nethost2str(iph->dest)
        );
    return r;
}
```

На підставі наведених вище базових відомостей, виходить от така невелика програма (ss, сокр. від англ. simple sniffer), що реалізує локальне прослуховування IP-трафіку. Інтерфейс її наведений нижче на рисунку.




Для компіляції буде досить навіть Visual Studio Express 2005.

*Що в нас вийшло в підсумку:*

– Сніффер працює в режимі користувача, однак вимагає привілею адміністратора.

– Пакети не фільтруються, відображаючись як є (можна додати налаштовуються фильтри).

– WiFi-трафік теж захоплюється (все залежить від конкретної моделі чипа), хоча є AirPcap, що чудово це вміє робити, але коштує грошей.

– Весь потік дейтаграм логується у файл.

Програма працює як сервер на порту 2000. Можна підключитися за допомогою утиліти telnet до хосту й зробити моніторинг потоків трафіку. Кількість підключень обмежена двадцятьома.

**Завдання :**

Реалізувати сніффер. Розроблене програмне забезпечення повинне включати функції перераховані у теоретичній частині.

## Лабораторна робота №4 Реалізація протоколу IPSec

**Ціль роботи: Розібратися з будовою протоколу IPSec й реалізувати свій такий протокол.**

### Теоретична частина

**IPsec** (скорочення від **IP Security**) – набір протоколів для забезпечення захисту даних, переданих по міжмережному протоколу IP. Дозволяє здійснювати підтвердження дійсності (автентифікацію), перевірку цілісності й/або шифрування IP-пакетів. IPsec також містить у собі протоколи для захищеного обміну ключами в мережі Інтернет. В основному, застосовується для організації VPN-з'єднань.

Розглянемо архітектуру сімейства протоколів IPSec. Ціль даного сімейства протоколів полягає в тому, щоб забезпечити різні сервіси безпеки на рівні IP для протоколів IPv4 і IPv6. Розглянемо сервіси безпеки, надавані протоколами IPSec, і використання цих протоколів у мережах TCP/IP.

Коли дані сервіси коректно встановлені, вони не заважають роботі користувачів, хостів і інших компонентів Інтернету, які не застосовують дані сервіси безпеки для захисту свого трафіку. Ці сервіси є алгоритмонезалежними. Це означає можливість додавання нових криптографічних алгоритмів без зміни самих протоколів. Наприклад, різні групи користувачів можуть використовувати різні набори алгоритмів.

Визначено стандартний набір алгоритмів за замовчуванням для забезпечення інтероперабельності у всьому Інтернеті. Використання цих алгоритмів разом із захистом трафіку, надаваний IPSec, і протоколами управління ключа дозволить розроблювачу систем і додатків забезпечити високий ступінь криптографічної безпеки.

IPSec може бути реалізований як в ОС, так і в маршрутизаторі або міжмережному екрані.

IPSec забезпечує конфіденційність, цілісність даних, керування доступом і автентифікацію джерела даних для IP-дейтаграм. Ці сервіси надаються за допомогою підтримки стану між джерелом і одержувачем IP-дейтаграм. Даний стан визначає конкретні сервіси забезпечення безпеки на рівні дейтаграми, використовувані криптографічні алгоритми для надаваних сервісів і ключі для цих алгоритмів.

***Перелічимо основні завдання протоколів IPSec:***

– Забезпечення криптографічного захисту на рівні IP для протоколів IPv4 і IPv6, а саме забезпечення конфіденційності й цілісності даних і цілісності деякої послідовності дейтаграм.

– Забезпечення прозорості для IP-трафіку, для якого не потрібне використання протоколів IPSec.

– Забезпечення розширюваності, тобто можливості додавати нові набори алгоритмів без зміни самого протоколу.

IPSec призначений для безпечної взаємодії з використанням криптографії для протоколів IPv4 і IPv6. Сервіси безпеки включають керування доступом, цілісність і конфіденційність даних і захист від replay-атак, що забезпечується гарантуванням цілісності деякої послідовності дейтаграм. Ці сервіси надаються на рівні IP, забезпечуючи захист для IP-протоколу й протоколів більше високого рівня.

*IPSec підтримує дві форми цілісності:*

– цілісність даних;

– цілісність визначеної послідовності дейтаграм.

Цілісність даних виявляє модифікацію конкретної IP-дейтаграми, безвідносно послідовності дейтаграм у потоці трафіку. Цілісність послідовності дейтаграм є анти-replay сервісом, за допомогою якого визначається одержання дублікатів IP-дейтаграм. Це відрізняється від забезпечення цілісності з'єднання, для якого існують більше строгі вимоги до цілісності трафіку, а саме, можливість визначення загублених або з повідомлень.

Розглянемо виконання протоколів IPSec, основні компоненти системи і їхня взаємодія для забезпечення сервісів безпеки.

IPSec виконується на хості (Host – H) або шлюзі безпеки (Security Gateway – SG), забезпечуючи захист IP-трафіку. Термін "шлюз безпеки" використовується для позначення маршрутизатора, що реалізує IPsec-протоколи.

Захист заснований на вимогах, визначених у базі даних політики безпеки (Security Policy Database – SPD), установлюваної й підтримуваної адміністратором. У загальному випадку пакети обробляються одним із трьох способів, заснованих на інформації IP-заголовка й транспортного рівня відповідно до записів в SPD. Кожний пакет або відкидається, або пропускається без обробки, або обробляється відповідно до запису SPD для даного пакета.

Можливі способи реалізації IPSec

Існує кілька способів реалізації IPSec на хості або разом з маршрутизатором або міжмережним екраном (для створення шлюзу безпеки).

– Інтеграція IPSec у конкретну реалізацію протоколу IP. Це вимагає доступу до вихідного коду IP і робиться як на хостах, так і на шлюзах безпеки.

– "in-the-stack" (BITS) реалізації, коли IPSec реалізований "унизу" існуючої реалізації стека IP-протоколів, вбудовуючи свою реалізацію між стандартною реалізацією IP-протоколів і локальних мережних драйверів. Доступу до вихідного коду стека IP у цьому випадку не потрібно. Даний підхід звичайно реалізується на хостах, коли IPSec реалізований у вигляді бібліотеки, яка підключається.

– Використання зовнішнього криптопроцесора. Звичайно це називається "in-the-wire" (BITW) реалізацією. Такі реалізації можуть використовуватися як на хостах, так і на шлюзах. Звичайно BITW-пристрої є IP-адресуемими.

***Можливі топології IPSec***

За допомогою протоколів IPSec можна реалізувати різні топології VPN. Основні топології дозволяють створювати наступні VPN:

– Шлюз безпеки – шлюз безпеки.

– Хост – шлюз безпеки.

– Хост – хост.

Приведемо чотири варіанти топологій VPN, створюваних між хостами й шлюзами безпеки, які реалізують IPSec. Уведемо наступні позначення:

| | |
|---|---|
| ——— | – SA (AH или ESP, транспортный или туннельный режим). |
| ——— | – незащищенная публичная сеть. |
| Host i | – хост № i. |
| SG i | – шлюз безопасности № i. |
| Rtr i | – маршрутизатор, не поддерживающий IPSec, через который должен проходить IPSec-трафик. |
| X* | – X (хост или шлюз безопасности) поддерживает IPSec. |

Рисунок 4.1 – Базові позначення

Розглянуті нижче безпечні асоціації можуть використовувати як AH-, так і ESP-протоколи. Режим (тунельний або транспортний) визначається характером кінцевих точок. Для Host – Host SA режим може бути як транспортним, так і тунельним. Для SG – SG SA режим швидше за все буде тунельним.

**Варіант 1**. Створення безпечного з'єднання між двома хостами через відкриту публічну мережу.

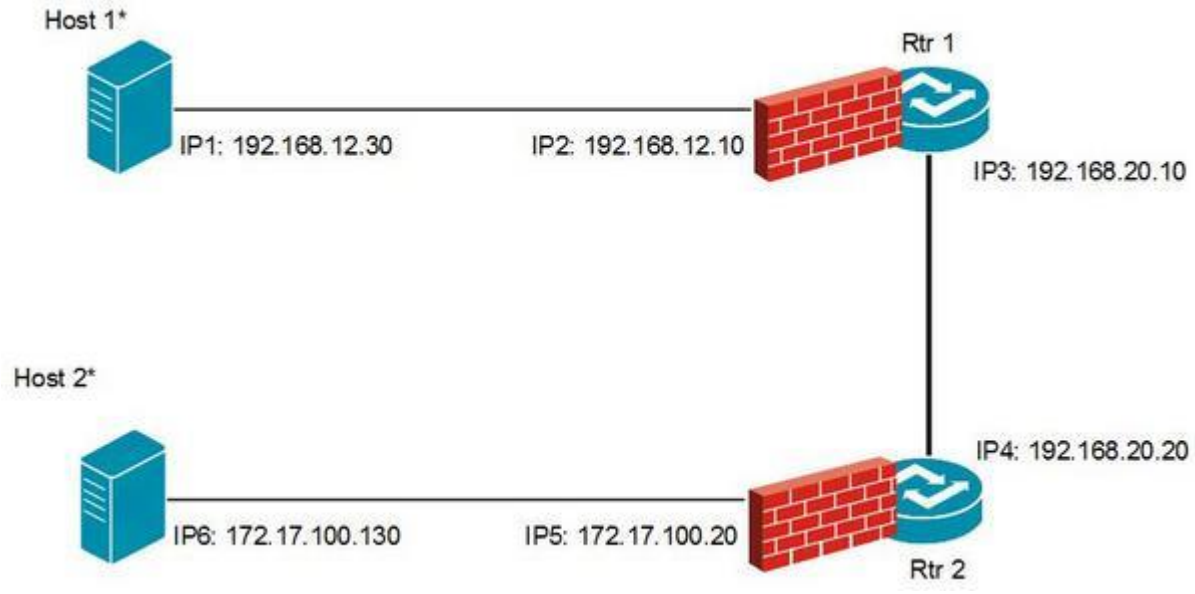



Рисунок 4.2 – Топологія мережі: VPN між двома хостами

У цьому випадку між хостами може використовуватися як транспортний, так і тунельний режим. Заголовки в пакеті між Host 1 і Host 2 виглядають у такий спосіб:

| Транспортный режим | Туннельный режим |
|---|---|
| [src:IP1; dst:IP6] [ESP]<br>[протокол более высокого уровня] | [src:IP3; dst:IP4] [ESP] [src:IP1; dst:IP6]<br>[протокол более высокого уровня] |

Рисунок 4.3 – Вкладеність заголовків при створенні VPN між двома хостами

У даній топології обидві кінцеві точки IP-з'єднання підтримують IPSec. Ці кінцеві точки можуть реалізовувати керування доступом на прикладному рівні, ґрунтуючись на автентифікації учасників. У транспортному режимі не існує внутрішнього IP-заголовка. У тунельному режимі внутрішній IP-заголовок існує, але, як правило, IP-адреси у внутрішньому й зовнішньому заголовках збігаються.

У даній топології маршрутизатори (Rtr 1 і Rtr 2) не підтримують IPSec, тобто не є шлюзами безпеки (SG), і не можуть аналізувати трафік, переданий між Host 1 і Host 2. Якщо ці маршрутизатори також виконують функції міжмережного екрана, то вони повинні пропускати весь IPSec-трафік, як трафік керування SA, так і трафік протоколів AH або ESP.

Трафік між Host 1 і Host 2 захищений сервісами безпеки як у публічній мережі, так і в обох локальних мережах. IP-адреси хостів видні як у публічній мережі, так і в обох локальних мережах.

**Варіант 2**. Створення віртуальної приватної мережі між двома віддаленими локальними мережами.

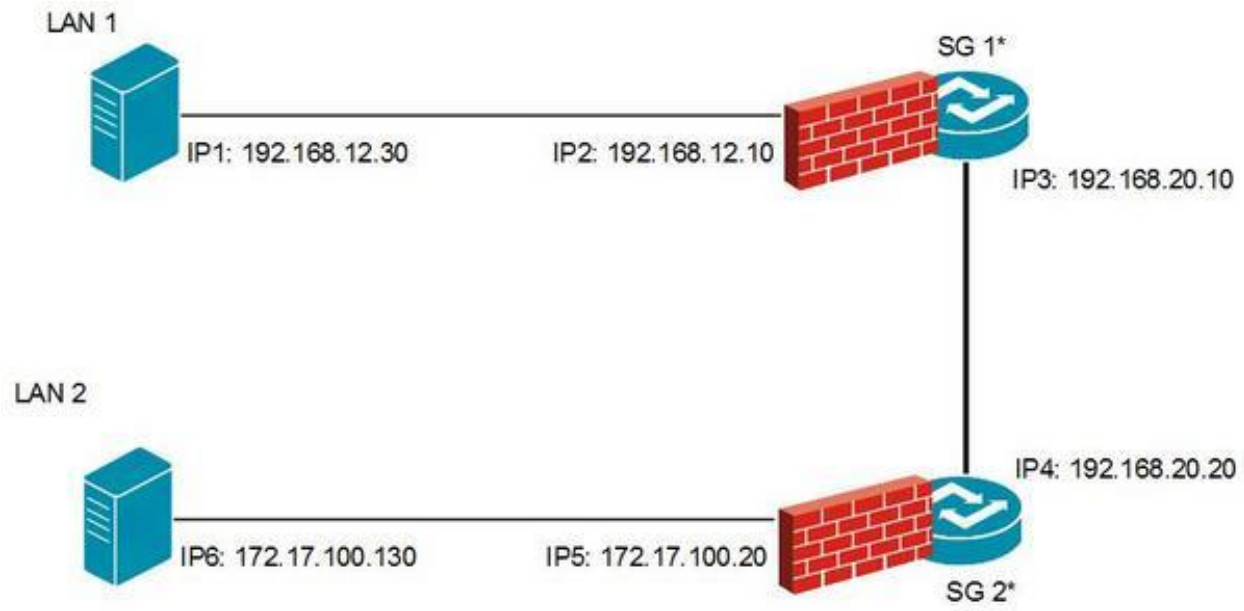



Рисунок 4.4 – Топологія мережі: VPN між двома локальними мережами

У цьому випадку, як правило, використовується тільки тунельний режим. При цьому заголовки в пакеті між SG1 і SG2 будуть виглядати в такий спосіб:

| Туннельный режим |
| --- |
| [src:IP3; dst:IP4] [ESP] [src: IP1; dst:IP6] [upper] |

Рисунок 4.5 – Вкладеність заголовків при створенні VPN між двома локальними мережами

У даній топології хости в локальних мережах не підтримують IPSec, і, як наслідок, трафік у локальних мережах не захищений від внутрішніх (insider) атак. Трафік у публічній мережі захищений. IP-адреси хостів у локальній мережі не видні в публічній мережі.

**Варіант 3**. Створення безпечного з'єднання між двома хостами з можливістю часткового аналізу й фільтрування трафіку на шлюзах безпеки.

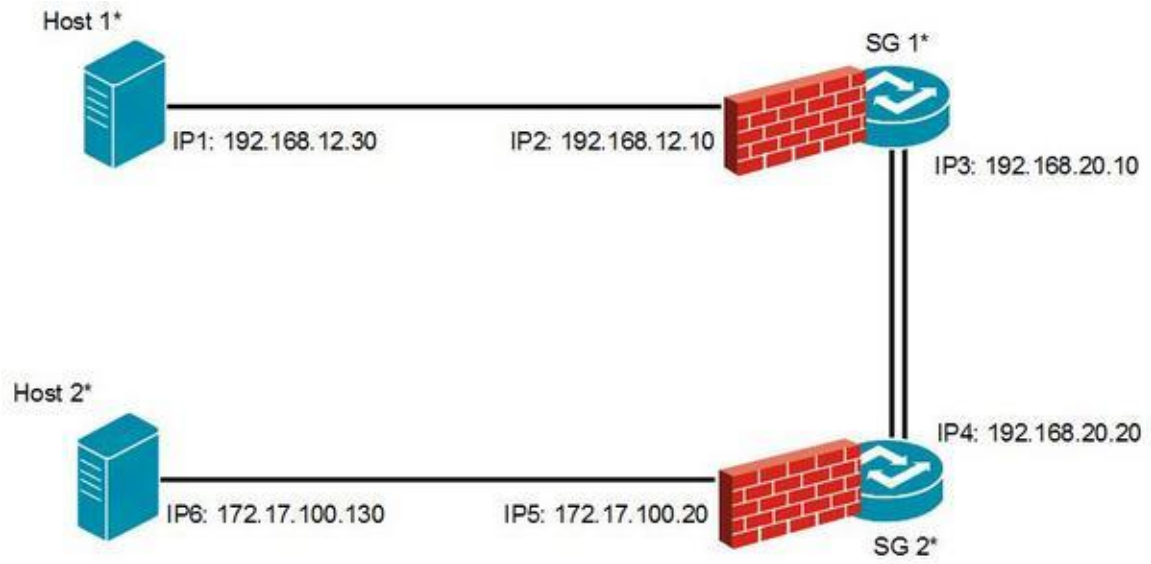



Рисунок 4.6 – Топологія мережі: VPN з можливістю аналізу трафіку на шлюзі безпеки

Створюються дві вкладені SA. Одна між хостами Host 1 і Host 2, інша між шлюзами безпеки SG 1 і SG 2. У цьому випадку трафік буде захищений як у публічній, так і в локальній мережах, і шлюзи безпеки зможуть частково аналізувати й фільтрувати трафік, переданий в і з локальних мереж.

На шлюзах безпеки повинен використовуватися тунельний режим. На хостах вірніше використовувати транспортний режим.

**Варіант 4**. Безпечне підключення віддаленого користувача до локальної мережі організації з можливістю часткового аналізу й фільтрування трафіку на шлюзі безпеки (Рисунок 4.7).

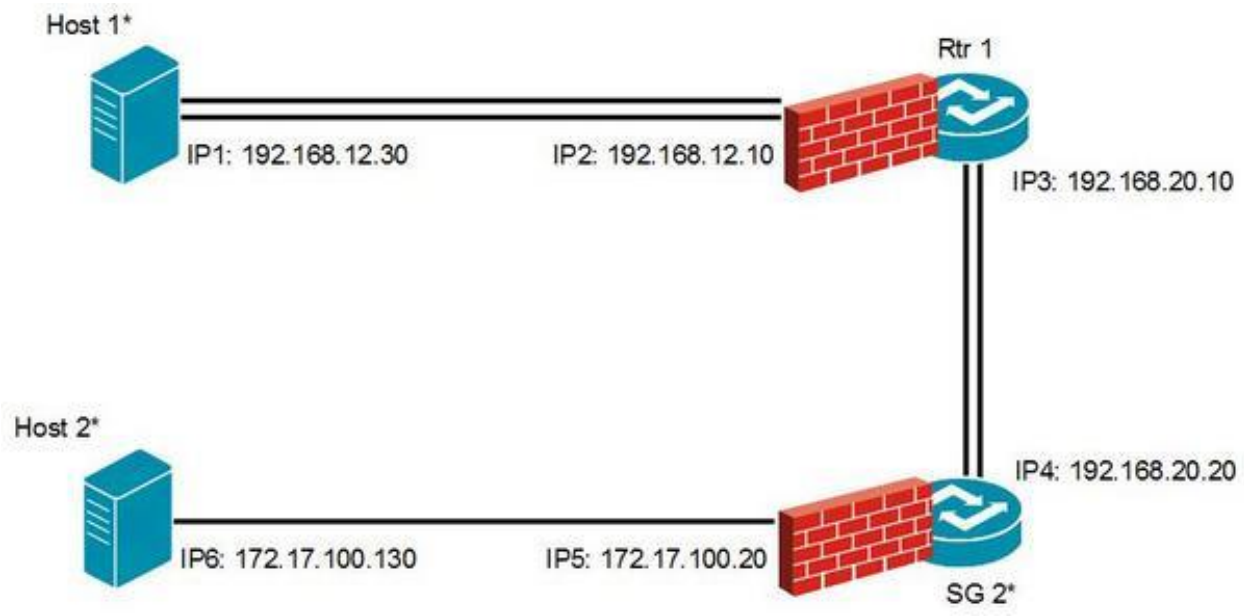



Рисунок 4.7 – Топологія мережі: захищений доступ користувача в локальну мережу

У цьому випадку створюються дві вкладені SA: одна між віддаленим хостом Host 1 і хостом у локальній мережі Host 2 (SA 1), друга між віддаленим хостом Host 1 і шлюзом безпеки SG 2 (SA 2). У результаті трафік захищений як в Інтернеті (SA 2), так і в локальній мережі (SA 1). Віддалений хост (Host 1) використовує Інтернет для досягнення міжмережного екрана організації (SG 2) і потім одержує доступ до деякому хосту (Host 2) у локальній мережі. Між Host 1 і SG 2 використовується режим тунелювання. Для SA між SG 2 і Host 2 можливий як транспортний, так і тунельний режими.

У даній топології кінцева точка, яка захищається (звичайно портативний переносний комп'ютер) з'єднується зі своєю корпоративною мережею через IPSec-тунель. Кінцева точка використовує даний тунель для доступу в корпоративну мережу, після цього трафік може тунелюватися через локальну мережу, щоб захистити його й у локальній мережі. У цьому випадку існує можливість фільтрування трафіку корпоративним міжмережним екраном. Кінцева точка повинна мати IP-адресу, відому міжмережному екрану, щоб він міг пропускати пакети через міжмережний екран і тунелювати їх далі. Дана IP-

адреса може бути статичнию або може задаватися динамічно якою-небудь із технологій, аналогічних DHCP. Для підтримки другого варіанта існує механізм, що дає можливість Ініціаторові запитувати IP-адресу, що належить міжмережному екрану для використання з SA, створюваної в локальній мережі.

***Ступінь деталізації керування трафіком***

IPSec дозволяє управляти деталізацією, з якої надається сервіс безпеки. Наприклад, можна створити єдиний зашифрований тунель між двома локальними мережами, або для кожного TCP– і UDP-з'єднання може бути створений окремий зашифрований тунель між парою хостів. IPSec дозволяє вказувати наступні параметри:

Необхідний рівень деталізації застосовуваного захисту. Варто помітити, що сильно деталізовані SA звичайно є більше уразливими для аналізу трафіку, чим слабко деталізовані.


Рисунок 4.8 – Приклад веб-інтерфейсу для вказівки ступеня деталізації створення SA

Використовувані алгоритми в протоколах забезпечення безпеки IP-трафіку.


Рисунок 4.9 – Приклад веб-інтерфейсу для вказівки необхідних алгоритмів у протоколах захисту трафіку

Використовувані алгоритми в протоколах керування SA.


Рисунок 4.10 – Приклад веб-інтерфейсу для вказівки необхідних алгоритмів у протоколах керування SA

**Завдання:**

Реалізувати передачу даних за протоколом IPSec між хостами, або візуалізувати роботу протоколу IPSec.

## Лабораторна робота №5 Реалізація протоколу TLS/SSL

**Ціль роботи:** Розібратися з будовою протоколу TLS/SSL й реалізувати цей протокол.

### Теоретична частина

**SSL** (англ. Secure Sockets Layer – рівень захищених сокетів) – криптографічний протокол, що має на увазі більше безпечний зв'язок. Він використовує асиметричну криптографію для автентифікації ключів обміну, симетричне шифрування для збереження конфіденційності, коди автентифікації повідомлень для цілісності повідомлень. Протокол широко використовувався для обміну миттєвими повідомленнями й передачі голосу через IP (англ. Voice over IP – VoIP), у таких додатках, як електронна пошта, інтернет-факс і др. У цей час відомо, що протокол не є безпечним. **SSL повинен бути виключений з роботи на користь TLS** (див. CVE-2014-3566).

SSL споконвічно розроблений компанією Netscape Communications для додавання протоколу HTTPS у свій веб-браузер Netscape Navigator. Згодом, на підставі протоколу SSL 3.0 був розроблений і прийнятий стандарт RFC, що одержав ім'я TLS.

**TLS** (англ. Transport Layer Security) – безпека транспортного рівня, як і його попередник SSL (англ. Secure Sockets Layer – рівень захищених сокетів) – криптографічні протоколи, що забезпечують захищену передачу даних між вузлами в мережі Інтернет. TLS і SSL використовують:

– асиметричну криптографію для автентифікації;

– симетричне шифрування для конфіденційності;

– коди автентичності повідомлень для збереження цілісності повідомлень.

Даний протокол широко використовується в додатках, що працюють із мережею Інтернет, таких як веб-браузери, робота з електронною поштою, обмін миттєвими повідомленнями й IP-телефонія (VoIP)

TLS-протокол заснований на специфікації протоколу SSL версії 3.0, розробленою компанією Netscape Communications. Зараз розвитком стандарту TLS займається IETF. Розходження між даним протоколом і SSL 3.0 несуттєві, але важливо помітити, що TLS 1.0 і SSL 3.0 несумісні, хоча в TLS 1.0 передбачений механізм, що дозволяє TLS мати зворотну сумісність із SSL 3.0.

**Основна функція протоколу SSL/TLS** складається в забезпеченні конфіденційності й цілісності даних прикладного рівня, переданих між двома взаємодіючими додатками, один з яких є клієнтом, а інший – сервером.

Протокол складається із двох рівнів. Нижнім рівнем, розташованим вище деякого надійного протоколу (а саме, протоколу TCP) є протокол Запису. Протокол Запису забезпечує безпеку з'єднання, що заснована на наступних двох властивостях:

Конфіденційність з'єднання. Для захисту даних використовується один з алгоритмів симетричного шифрування. Ключ для цього алгоритму створюється для кожної сесії й заснований на секреті, про яке домовляються в протоколі Рукостискання. Протокол Запису також може використовуватися без шифрування.

Цілісність з'єднання. Забезпечується перевірка цілісності повідомлення за допомогою MAC із ключем. Для обчислення MAC використовуються хеш-функції SHA-1 і MD5. Протокол Запису може виконуватися без обчислення MAC.

Одна з переваг TLS полягає в тому, що він незалежний від прикладного протоколу.

***Протокол Запису***

Протокол Запису складається з декількох рівнів. Протокол Запису фрагментує повідомлення на блоки потрібної довжини, здійснює стиск даних, обчислює HMAC і зашифровує їх. На іншому кінці з'єднання отримані дані розшифровуються, перевіряється їхня цілісність, далі вони декомпрессуються, дефрагментуються й передаються протоколам більше високого рівня.

Вище протоколу Запису можуть розташовуватися наступні протоколи:

– протокол Рукостискання;

– Alert-протокол;

– протокол зміни шифрування;

– прикладний протокол, безпека якого забезпечується.

***Стан з'єднання***

У протоколі вводиться поняття стану з'єднання, що визначає параметри виконання протоколу Запису. Такими параметрами є:

– алгоритм стиску;

– алгоритм шифрування;

– MAC-алгоритм;

– параметри цих алгоритмів, тобто секрети MAC, ключі алгоритму шифрування й ініціалізаційні вектора.

Для кожного напрямку (відповідно читання або запис) параметри з'єднання можуть розрізнятися.

Існує чотири стани з'єднання:

– поточні стани читання й запису;

– очікувані стани читання й запису.

Параметри безпеки для очікуваних станів установлюються протоколом Рукостискання, а протокол зміни шифрування робить очікуваний стан поточним, у результаті чого відповідні параметри поточного стану скидаються й заміняються параметрами очікуваного стану. Параметри очікуваного стану ініціалізується порожніми значеннями. Початковий поточний стан завжди визначається без використання шифрування, стиску й MAC.

Визначено наступні параметри стану:

Таблиця 5.1.

| | |
|---|---|
| Кінець з'єднання | Кожний учасник є або "клієнтом", або "сервером" |
| MAC алгоритм | Алгоритм, використовуваний для перевірки цілісності повідомлення, секрет MAC. |

| | |
|---|---|
| Алгоритм симетричного шифрування | Алгоритм, використовуваний для симетричного шифрування, і його параметри – довжина ключа алгоритму, довжина блоку алгоритму, ключ шифрування, ініціалізаційний вектор (IV) і ін. |
| Алгоритм стиску | Алгоритм, використовуваний для стиску даних. |
| Майстер-Секрет | 48-байтний секрет, поділюваний обома учасниками з'єднання. |
| Випадкове число клієнта SecurityParameters.client_random | 32-байтне значення, створюване клієнтом у протоколі Рукостискання. |
| Випадкове число сервера SecurityParameters.server_random | 32-байтне значення, створюване сервером у протоколі Рукостискання. |
| Послідовний номер | Кожний стан з'єднання містить послідовний номер, що обчислюється незалежно для станів читання й запису. Послідовний номер повинен установлюватися в нуль при ініціалізації стану. Послідовні номери не можуть бути більше $2^{64} – 1$. Послідовний номер зростає після створення чергового запису. |

З майстер-секрету створюються шість ключів:

– client write MAC secret;

– server write MAC secret;

– client write key;

– server write key;

– client write IV;

– server write IV.

Ключі client write використовуються сервером, коли він одержує повідомлення й клієнтом, коли той посилає їх. Ключі server write використовуються сервером, коли він посилає повідомлення й клієнтом, коли він одержує їх. Після того як параметри безпеки встановлені й ключі створені, очікувані стани з'єднання робляться поточними.

**Обчислення ключів**

Протокол Запису використовує наступний алгоритм для створення ключів, ініціалізаційних векторів і секретів MAC із параметрів безпеки, створюваних протоколом Рукостискання.

**HMAC і псевдовипадкова функція**

Для забезпечення цілісності використовується HMAC із хеш-функціями MD5 і SHA-1, позначуваними як:

– HMAC_MD5 (secret,data);

– HMAC_SHA (secret, data).

В алгоритмі визначена псевдовипадкова функція PRF, що розширює секрет до потрібної довжини для створення всіх необхідних ключів. Ця функція одержує як вхід секрет, "зерно" (seed – значення, що з однієї сторони є випадковим, а з іншої сторони не є секретним, тобто може стати відомо опонентові) і стандартне значення, і створює вихід необхідної довжини.

Спочатку визначається функція розширення даних P_hash (secret, data), що використовує хеш-функцію для розширення секрету до потрібної довжини в такий спосіб:

P_hash (secret, seed) = HMAC_hash (secret, A(1) || seed) ||
HMAC_hash (secret, A(2) || seed) || HMAC_hash (secret, A(3) || seed) ||

A (i) визначається в такий спосіб:

A (0) = seed

A (i) = HMAC_hash (secret, A (i – 1))

P_hash може мати стільки ітерацій, скільки необхідно для створення даних необхідної довжини. Наприклад, якщо P_ SHA-1 використовується для створення 64 байтів даних, то кількість ітерацій повинне бути дорівнює 4, при цьому буде створено 80 байтів даних; останні 16 байтів заключної ітерації будуть відкинуті, щоб залишити тільки 64 байта вихідних даних.

Для одержання ключового матеріалу потрібної довжини секрет, обчислений у протоколі Рукостискання, ділиться на дві половини, одна половина використовується для створення даних за допомогою P_MD5, а інша – для створення даних за допомогою P_ SHA-1.

PRF визначається як результат додавання по модулі 2 результатів виконання P_MD5 і P_ SHA-1.

PRF (secret, label, seed) = P_MD5 (S1, label + seed) ⊕

P_ SHA-1 (S2, label + seed)

Label є фіксованим текстовим рядком.

Помітимо, що оскільки MD5 створює 16-байтні значення, а SHA-1 створює 20-байтні значення, то кількість ітерацій кожної з функцій буде різним. Наприклад, для створення 80-байтного значення необхідно виконати 5 ітерацій P_MD5 і 4 ітерації P_ SHA-1.

Для створення ключів обчислюється наступне значення:

key_block = PRF (SecurityParameters.master_secret, "key expansion",

SecurityParameters.server_random + SecurityParame-ters.client_random)

Кількість ітерація в PRF визначається сумарною довжиною ключів. Потім key_block розбивається на блоки для одержання необхідних ключів.

***Протокол Рукостискання***

Протокол Рукостискання складається із трьох підпротоколів, використання яких дозволяє учасникам погодити криптографічні алгоритми, автентифікувати один одного, і повідомити один одному про виникнення тих або інших помилок.

У результаті виконання протоколу Рукостискання будуть створені наступні елементи сесії:

Таблиця 5.2.

| | |
|---|---|
| Ідентифікатор сесії | Довільна послідовність байтів, обирана сервером для ідентифікації активного або поновлюваного стану сесії. |
| Сертифікат учасника | X.509 v3 сертифікат учасника. Цей елемент може бути нульовим. |
| Метод стиску | Алгоритм, використовуваний для стиску даних перед шифруванням. |
| Набір алгоритмів | Алгоритм симетричного шифрування даних (наприклад, NULL, DES, AES і т.д.), MAC-алгоритм (такий як MD5 або SHA-1) і параметри цих алгоритмів. |
| Майстер-Секрет | 48-байтний секрет, поділюваний клієнтом і сервером. |
| Поновлювано | Прапор, що визначає, чи може дана сесія використовуватися для створення нового TCP-з'єднання. |

**Протокол зміни шифрування**

Протокол складається з єдиного повідомлення, що зашифроване й стисле, як визначено в поточному стані з'єднання.

Повідомлення про зміну шифрування посилає як клієнтом, так і сервером для повідомлення протилежної сторони про те, що наступні записи будуть захищені алгоритмами й ключами, про які сторони тільки що домовилися. При одержанні даного повідомлення протокол Запису копіює очікуваний стан читання в поточний стан читання. Відразу після посилки даного повідомлення відправник копіює очікуваний стан запису в поточний стан запису. Повідомлення про зміну шифрування посилає при Рукостисканні після того, як параметри безпеки погоджені, але перед тим як посилає заключне веріфікуюче повідомлення.

**Alert-протокол**

Одним із протоколів, що виконуються вище протоколу Запису, є протокол Alert. Умістом протоколу є або фатальне, або попереджуюче повідомлення. Фатальне повідомлення повинне приводити до негайного розриву даного TCP-з'єднання. У цьому випадку інші з'єднання, що відповідають даної сесії, можуть бути продовжені, але ідентифікатор сесії повинен бути позначений як недійсний для запобігання використання даної сесії для встановлення нових з'єднань. Подібно іншим повідомленням, повідомлення Alert зашифровані й стислі, як визначено в поточному стані з'єднання.

**Протокол Рукостискання**

Криптографічні параметри сесії створюються протоколом Рукостискання, що виконується вище протоколу Запису. Коли клієнт і сервер починають взаємодіяти, вони погоджують версію протоколу, вибирають криптографічні алгоритми, можуть автентифікувати один одного, використовуючи технологію з відкритим ключем. Для створення поділюваного секрету також використовується технологія з відкритим ключем.

Протокол Рукостискання складається з наступних кроків:

– Обмін повідомленнями Hello для узгодження алгоритмів, обміну випадковими значеннями й перевірки поновлюємості сесії.

– Обмін необхідними криптографічними параметрами, які дозволяють клієнтові й серверу погодити премайстер-секрет.

– Обмін сертифікатами й криптографічною інформацією, що дозволяє клієнтові й серверу автентифікувати один одного.

– Надання параметрів безпеки на рівень Запису.

Можливість клієнтові й серверу перевірити, що вони обчислили ті самі параметри безпеки й що Рукостискання відбулося без втручання зловмисника.

Протокол розроблений для мінімізації ризику атак "зустріч посередині", але захист від атак, при яких зловмисник може блокувати доступ до порту, не передбачений.

Клієнт посилає повідомлення ClientHello, на яке сервер повинен відповісти повідомленням ServerHello або фатальною помилкою й розривом з'єднання. ClientHello і ServerHello використовуються для визначення максимального рівня безпеки між клієнтом і сервером.

Client Hello і Server Hello установлюють наступні атрибути: Protocol Version, Session ID, Cipher Suite і Compression Method. Додатково створюються й передаються два випадкових значення: ClientHello.random і ServerHello.random.

Автентифікація й обмін загальним секретом здійснюються в чотирьох повідомленнях: сертифікат сервера, обмін ключа сервера, сертифікат клієнта й обмін ключа клієнта. Загальний секрет повинен бути досить більшим; поточні методи розподілу ключа обмінюються секретами, довжина яких перебуває в діапазоні від 48 до 126 байт.

Після повідомлень Hello сервер посилає сертифікат, за допомогою якого клієнт виконує автентифікацію сервера. Додатково може бути послане повідомлення обміну ключа сервера, якщо сервер не має сертифіката або його сертифікат може використовуватися тільки для перевірки підпису. Якщо сервер автефікований, він може запросити сертифікат клієнта, якщо того вимагає встановлена політика безпеки на стороні сервера. Після цього сервер посилає повідомлення Server Hello Done, що вказує на те, що фаза Hello-повідомлень рукостискання завершена. Потім сервер чекає відповіді клієнта. Якщо сервер послав повідомлення запиту сертифіката, клієнт повинен послати повідомлення Certificate. Після цього посилає повідомлення обміну ключа клієнта. Уміст цього повідомлення залежить від обраного алгоритму виробітку загального секрету. Якщо клієнт посилав свій сертифікат, то він посилає повідомлення, що містить цифровий підпис для перевірки всіх повідомлень Рукостискання.

У даній точці клієнтом посилає повідомлення про зміну стану, і клієнт копіює очікуваний стан у поточний стан. Після цього клієнт посилає заключне повідомлення з використанням нових алгоритмом, ключів і секретів. У відповідь сервер посилає своє повідомлення про зміну стану, перетворить

очікуваний стан у поточний стан і посилає заключне повідомлення з використанням нових алгоритмів і ключів. Після цього рукостискання вважається виконаним, і клієнт і сервер можуть починати обмін даними прикладного рівня.


Рисунок 5.1. Послідовність повідомлень при повнім Рукостисканні

Коли клієнт і сервер вирішують відновити попередню сесію або дублювати існуючу (замість того щоб вести нові переговори про параметри безпеки), виконується так зване скорочене рукостискання:

Клієнт посилає Client Hello, використовуючи Session ID поновлюваної сесії.

Сервер шукає відповідний ідентифікатор сесії у своєму кеші сесій. Якщо ідентифікатор існує, і сесія позначена як поновлювана, сервер установлює з'єднання з параметрами зазначеної сесії, після чого посилає Server Hello із цим значенням Session ID. Якщо відповідний Session ID не знайдений, сервер створює новий ID сесії, і клієнт і сервер виконують повне рукостискання.

Після цього й клієнт, і сервер повинні послати повідомлення про зміну стану, потім відразу послати завершальні повідомлення.

І клієнт, і сервер починають обмін даними прикладного рівня.

Потік повідомлень при скороченому Рукостисканні

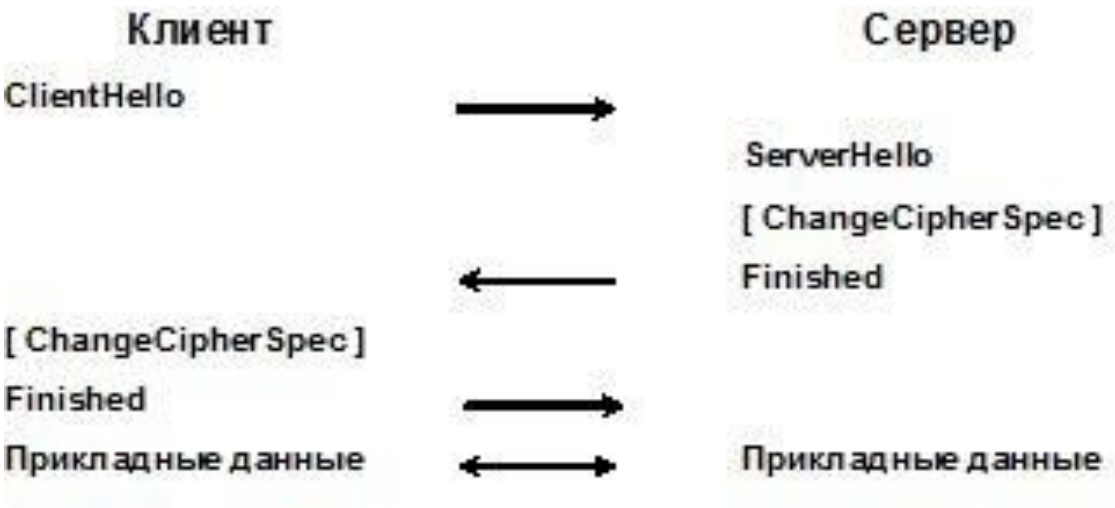



Рисунок 5.2. Послідовність повідомлень при скороченому Рукостисканні

Варто помітити, що, тому що Session ID передається без шифрування й забезпечення цілісності, він не містить конфіденційну інформацію. Уміст усього Рукостискання, включаючи Session ID, захищено Finished-повідомленнями, якими учасники обмінюються наприкінці рукостискання.

Список Cipher Suite, переданий від клієнта серверу в повідомленні Client Hello, містить перелік криптографічних алгоритмів, підтримуваних клієнтом, упорядкований по перевагах клієнта. Сервер вибирає по одному алгоритму з кожної категорії, що він підтримує. Якщо такого алгоритму не існує, сервер повертає фатальний Alert і закриває з'єднання.

Після посилки повідомлення Client Hello клієнт чекає повідомлення Server Hello. Будь-яке інше повідомлення, що повертається сервером, за винятком Hello Request, трактується як фатальна помилка.

Сервер посилає Server Hello у відповідь на повідомлення Client Hello, для того щоб вибрати конкретний набір алгоритмів. Якщо для якогось типу алгоритмів клієнт і сервер не мають однакового алгоритму, TCP-з'єднання буде скинуто.

**Повідомлення** Certificate (сервера)

Сервер повинен посилати сертифікат, якщо метод обміну ключів не є анонімним. Дане повідомлення завжди треба відразу за повідомленням Server Hello.

Тип сертифіката повинен відповідати обраному алгоритму обміну ключа. Звичайно це сертифікат X.509v3. Він повинен містити ключ, що відповідає методу обміну ключа.

Повідомлення сервера обміну ключа посилає сервером тільки тоді, коли повідомлення Server Certificate (якщо воно послано) не містить досить даних для того, щоб клієнт міг здійснити обмін премайстер-секретом.

Дане повідомлення передає криптографічну інформацію, що дозволяє клієнтові передавати премайстер-секрет: премастер-секрет шифрується або відкритим ключем RSA, або відкритим ключем Діффі-Хеллмана.

**Повідомлення** Certificate Request

Неанонімний сервер може додатково запросити сертифікат клієнта, якщо це потрібно політикою безпеки сервера.

**Повідомлення** Server Hello Done

Повідомлення Server Hello Done посилає сервером як ознака закінчення фази Server Hello.

**Повідомлення** Certificate (клієнта)

Це перше повідомлення, що клієнт посилає після одержання повідомлення Server Hello Done. Воно посилає тільки в тому випадку, якщо сервер запросив автентифікацію клієнта.

**Повідомлення** Client Key Exchange

Дане повідомлення посилає клієнтом завжди. Воно треба відразу за повідомленням Client Certificate, якщо воно посилало. У противному випадку

це перше повідомлення, послане клієнтом після одержання повідомлення Server Hello Done.

Після одержання даного повідомлення сервер може обчислити премайстер-секрет, що передається або за допомогою RSA шифрування, або обчислюється по алгоритму Діффі-Хеллмана. У кожному разі кожна сторона обчислює той самий премайстер-секрет.

**Перевірка цілісності за допомогою сертифіката клієнта**

Дане повідомлення використовується для виконання перевірки цілісності переданих і отриманих повідомлень Рукостискання й автентифікації клієнта. Воно посилає тільки в тому випадку, якщо алгоритм відкритого ключа, для якого створений сертифікат клієнта, має можливість підписування. Це означає, що виключенням є сертифікати, створені для відкритого ключа алгоритму Діффі-Хеллмана.

**Повідомлення** Finished

Повідомлення Finished завжди посилає безпосередньо після повідомлення Change Cipher Spec для перевірки успішного виконання обміну ключа й процесів автентифікації. Повідомлення Change Cipher Spec повинне бути отримане після інших повідомлень Рукостискання й перед Finished-повідомленням.

Finished-повідомлення є першим повідомленням, захищеним за допомогою тільки що обговорених алгоритмів і ключів. Одержувачі Finished-повідомлення повинні переконатися, що його вміст коректно. Після того як одна сторона послала своє Finished-повідомлення, одержала й перевірила Finished-повідомлення іншої сторони, вона може починати посилати й одержувати прикладні дані по цьому з'єднанню.

**Обчислення майстер-секрету**

Незалежно від методів обміну ключа використовується наступний алгоритм для перетворення премайстер-секрету в майстра-секрет. Премайстер-Секрет повинен бути вилучений після того, як обчислений майстер-секрет.

master_secret = PRF(pre_master_secret, "master secret",
ClientHello.random+ServerHello.random) [0..47]

Довжина майстер-секрету завжди дорівнює 48 байтам. Довжина премайстер-секрету змінюється залежно від методу обміну ключа.

**Завдання:**

Реалізувати передачу даних за протоколом TLS/SSL між хостами, або візуалізувати роботу протоколу TLS/SSL.

## Лабораторна робота №6 Реалізація системи виявлення вторгнень

**Ціль роботи: Розібратися з будовою системи виявлення вторгнень й реалізувати таку систему.**

### Теоретична частина

***Виявлення вторгнень*** – це ще одне завдання, виконуване співробітниками, відповідальними за безпеку інформації в організації, при забезпеченні захисту від атак. Виявлення вторгнень – це активний процес, при якому відбувається виявлення хакера при його спробах проникнути в систему. Периметр захисту мережі являє собою віртуальний периметр, усередині якого перебувають комп'ютерні системи. Цей периметр може визначатися міжмережними екранами, точками поділу з'єднань або настільних комп'ютерів з модемами

**Система виявлення вторгнень (IDS)** призначена для розмежування авторизованого входу й несанкціонованого проникнення, що реалізується набагато складніше.

***Визначення типів систем виявлення вторгнень***

Існують два основних типи IDS:

– вузлові (HIDS);

– мережні (NIDS).

Система HIDS розташовується на окремому вузлі й відслідковує ознаки атак на даний вузол.

На рисунку 6.1 показані два типи IDS, які можуть бути присутні у мережному середовищі.

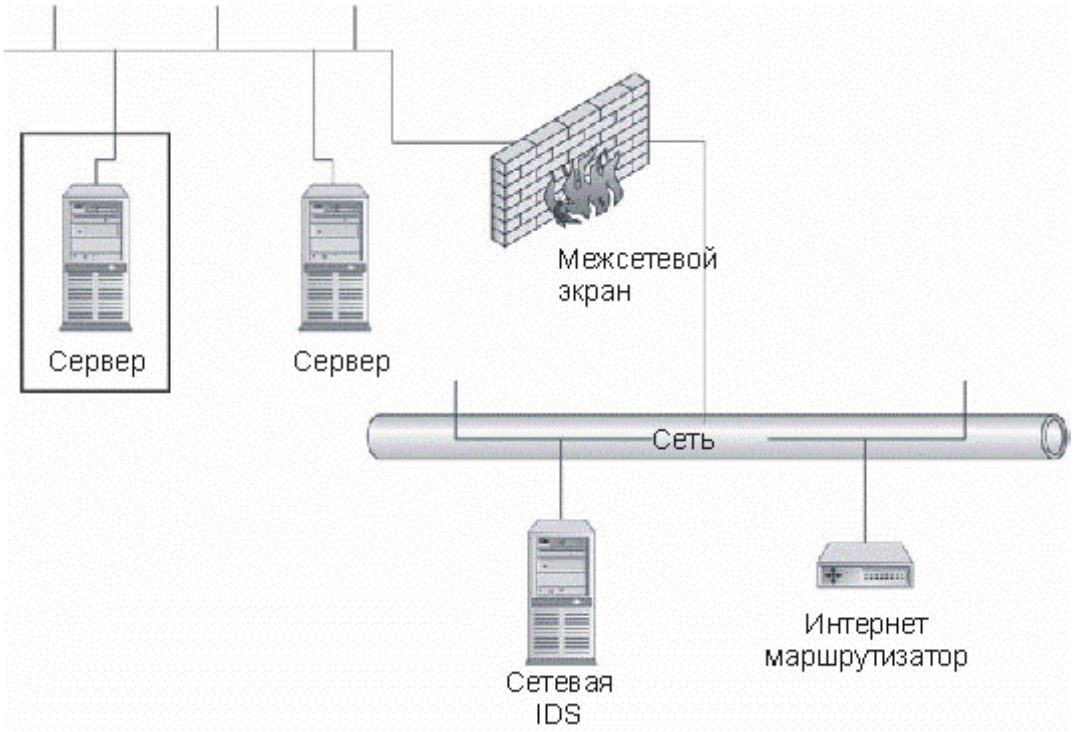



Рисунок 6.1 – Приклади розміщення IDS у мережному середовищі

***Вузлові IDS***

Вузлові IDS (HIDS) являють собою систему датчиків, що завантажуються на різні сервера організації й керованих центральним диспетчером. Датчики відслідковують різні типи подій і вживають визначені дії на сервері або передають повідомлення. Датчики HIDS відслідковують події, пов'язані із сервером, на якому вони завантажені. Сенсор HIDS дозволяє визначити, чи була атака успішною, якщо атака мала місце на тій же платформі, на якій установлений датчик.

***Мережні IDS***

NIDS являє собою програмний процес, що працює на спеціально виділеній системі. NIDS перемикає мережну карту в системі в нерозбірливий режим роботи, при якому мережний адаптер пропускає весь мережний трафік (а

не тільки трафік, спрямований на дану систему) у програмне забезпечення NIDS. Після цього відбувається аналіз трафіку з використанням набору правил і ознак атак для визначення того, чи представляє цей трафік який-небудь інтерес. Якщо це так, то генерується відповідна подія.

Найчастіше при застосуванні NIDS використовуються дві мережні карти (див. Рисунок 6.2). Одна карта використовується для моніторингу мережі. Ця карта працює в "схованому" режимі, тому вона не має IP-адреси й, отже, не відповідає на вхідні з'єднання.

У схованої карти відсутній стек протоколів, тому вона не може відповідати на такі інформаційні пакети, як пінг-запити. Друга мережна карта використовується для з'єднання із системою керування IDS і для відправлення сигналів тривоги. Ця карта приєднується до внутрішньої мережі, невидимої для тієї мережі, у відношенні якої виробляється моніторинг.


Рисунок 6.2. Конфігурація NIDS із двома мережними картами

**Про що може повідомити система IDS**

Система виявлення вторгнень може тільки видавати звіти про ті події, на виявлення яких вона налаштована. Конфігурація IDS складається із двох компонентів. Першою з них є ознаки атак, запрограмовані в системі. Другий компонент – будь-яким додаткові, визначені адміністратором, події, що також представляють інтерес. Серед цих подій можуть бути визначені типи трафіку або повідомлень журналу.

За допомогою включення в кінцевий продукт ознак атак постачальник або розроблювач системи по-своєму інтерпретує рівень важливості зазначених подій. Ступінь важливості, що привласнюється визначеним подіям у тій або іншій організації, може бути зовсім інший, ніж той, який передбачив розроблювач. Може знадобитися змінити параметри за замовчуванням для деяких ознак або просто відключити ознаки, не застосовні до організації.

Варто мати на увазі, що система IDS буде видавати попередження тільки про ті події, які вона виявить. Якщо на системі, що відслідковується датчиком HIDS, не заносяться в журнал визначені події, то датчик HIDS не буде їх розпізнавати. Аналогічно, якщо датчик NIDS не може "бачити" визначений трафік, він не видасть попередження навіть у тому випадку, якщо подія відбудеться.

З умовою правильної конфігурації IDS можна привести чотири типи подій, про які буде повідомляти система IDS:

– Події дослідження.

– Атаки.

– Порушення політики.

– Підозрілі або непояснені події.

Більша частина часу буде приділятися дослідженню підозрілих подій.

**Події дослідження**

Події дослідження являють собою спроби атакуючого зібрати дані про систему перед безпосереднім проведенням атаки. Ці події можна розділити на п'ять категорій.

1."Сховане" сканування.

2. Сканування портів.

3. Сканування "троянських коней".

4. Сканування уразливостей.

5. Відстежування файлів.

Більша частина цих подій відбувається в мережі, в основному, вони виходять із Інтернету й спрямовані на системи із зовнішніми адресами.

Події дослідження виявляють собою спроби збору інформації про системи. Вони не є подіями, що впливають на систему. Деякі комерційні IDS сприймають події дослідження як події високого пріоритету. З обліком того, що ці події не наносять збитку системі, такий підхід можна порахувати нерозумним.

Джерелом подібного трафіку може бути й інша система-жертва, захоплена зловмисником, тому дану інформацію варто повідомляти системним адміністраторам цього вузла.

***Сховане сканування.*** Сховане сканування – це спроби ідентифікації систем, що є присутні у мережі, з метою запобігти виявленню системи, з якої буде проводитися атака. Цей тип сканування буде визначатися датчиками NIDS як половинчасте сканування IP або сховане сканування IP, і, як правило, таке сканування спрямоване на велике число IP-адрес. Відповідною реакцією є ідентифікація джерела й інформування власника системи-джерела про те, що його система, швидше за все, піддалася впливу зловмисника.

***Сканування портів.*** Сканування портів використовується для визначення служб, що працюють на системах мережі. Системи виявлення вторгнень виявляють сканування портів у випадку, коли визначене число портів (відповідне граничному значенню) на одній системі відкривається протягом невеликого проміжку часу. Датчики NIDS і деякі датчики HIDS забезпечують ідентифікацію даного типу сканування й становлять відповідні звіти. Відповідні дії на сканування даного типу ідентичні відповідним діям на сховане сканування.

***Сканування "троянських коней".*** Існує безліч шкідливих програм типу "троянський кінь". Датчики NIDS містять ознаки, що визначають багато які з цих програм. На жаль, трафік, спрямований на "троянські" програми, як правило, визначається кінцевим портом пакета. Ця обставина викликає велику кількість помилкових спрацьовувань системи виявлення вторгнень. У випадку виникнення події "Trojan" варто перевіряти вихідний порт трафіку. Приміром, трафік, що виходить із порту 80, як правило, надходить із веб-сайту.

Одним з найпоширеніших типів "троянського" сканування є сканування BackOrifice. Програма BackOrifice використовує порт 31337, і дуже часто зловмисники здійснюють сканування діапазону адрес для цього порту. Консоль BackOrifice також містить функцію "ping host" (відправлення пінг-запитів на вузли), що здійснює сканування автоматично. Турбуватися не про що, поки не буде зафіксований вихідний трафік із внутрішньої системи. Знов-таки, у цьому випадку потрібно зв'язатися із власником системи-джерела, так як вона, імовірно, піддалася впливу зловмисника.

***Сканування уразливостей.*** Сканування уразливостей розпізнається системою IDS при виявленні великого набору різних ознак атак. Як правило, таке сканування спрямоване на кілька систем. Рідкі випадки, коли сканування уразливостей виробляється стосовно діапазону адрес без активних систем.

***Сканування уразливостей***, здійснюване хакерами, неможливо відрізнити від сканування уразливостей, проведеного компаніями, які перевіряють рівень захищеності інформаційних систем (у багатьох випадках у цих компаніях використовуються ті ж самі засоби). Так чи інакше, саме по собі сканування не заподіює системі якої-небудь шкоди, однак якщо атакуючий виконав сканування, у результаті якого виявилися системи з уразливостями до атаки, йому після цього стає відомо, які системи можна атакувати. Для забезпечення відповідності комп'ютерних систем актуальним проблемам безпеки варто контактувати із власником системи-джерела й перевіряти внутрішні системи організації на наявність самих останніх надбудов безпеки й відновлень.

Як правило, складно відрізнити сканування уразливостей від атаки, так як IDS в обох випадках ініціює ті самі події. Різниця тут полягає в кількості подій. Сканування уразливостей, як правило, супроводжується великим числом різних подій за дуже малий відрізок часу, у той час як при проведенні атак відбуваються події одного типу.

***Відстежування файлів***. Відстеження файлів або перевірка файлових дозволів, як правило, здійснюється внутрішнім користувачем. Користувач намагається визначити, до яких файлів можна здійснити доступ і що ці файли можуть містити. Даний тип розвідки розпізнається тільки датчиком HIDS і тільки в тому випадку, якщо в системі ведеться журнал спроб несанкціонованого доступу. Окремі події подібного роду, як правило, являють собою безневинні помилки, однак якщо простежується визначена схема, те варто зв'язатися з користувачем і з'ясувати, що ж відбулося.

**Атаки**

Події атак вимагають найшвидшої відповідної реакції. В ідеальному випадку IDS повинна бути налаштована тільки на ідентифікацію подій високого пріоритету у випадку використання відомої внутрішньої уразливості. У цьому випадку повинна бути негайно застосована процедура обробки інциденту.

Майте на увазі, що IDS не розпізнає різницю між безпосередньою атакою й скануванням уразливостей, що виглядає як атака. Адміністратор системи IDS повинен проводити оцінку інформації, представленою системою IDS, для визначення того, чи є подія атакою. По-перше, необхідно з'ясувати число подій. Якщо протягом короткого проміжку часу спостерігався набір ознак різних атак, то це, швидше за все, сканування уразливостей, а не безпосередня атака. Якщо ж виявлена одна ознака атаки, спрямована на одну або кілька систем, то ця подія може являти собою справжню атаку.

**Порушення політики**

Більша частина систем IDS поставляється з ознаками наступних подій:

– Загальний доступ до файлів (Gnutella, Kazaa і т.д.).

– Обмін миттєвими повідомленнями.

– Сеанси Telnet.

– Команди "r" (rlogin, rsh, rexec).

У більшій частині організацій використання такого трафіку є порушенням політики безпеки. На жаль, такі порушення політики можуть представляти для організації більшу небезпеку, ніж безпосередні атаки. У більшості випадків подія відбувається в дійсності. Таким чином, відкривається доступ до файлів, і системи налаштовуються на дозвіл виконання команди rlogin.

Вибір методу обробки різних порушень політики залежить від внутрішніх політик і процедур, що мають місце в організації. Проте, необхідно роз'яснити всі моменти системному адміністраторові або відповідальній особі, щоб йому стала ясна суть політик організації.

**Підозрілі події**

Події, що не відповідають повністю ні однієї з інших категорій, заносяться в категорію підозрілих подій. Підозрілою подією називається подія, що не вдалося розпізнати. Наприклад, ключ реєстру Windows NT був змінений з незрозумілої причини. Це не схоже на атаку, але в той же час не ясно, які причини змінився ключ. Як інший приклад можна привести пакет із прапорами заголовка, що порушують стандарт протоколу. Це може бути спроба розвідувального сканування, результат несправності мережної карти системи або пакет, при передачі якого виникли помилки. У даних, видаваних системою IDS, не надається досить відомостей для чіткого визначення конкретної ситуації й з'ясування того, що відбулося – необразлива помилка або атака.

Не менш підозрілим може виявитися несподіваний мережний трафік, що з'явився у внутрішній мережі. Якщо робоча станція починає запитувати SNMP-дані з інших систем, то це може бути як наслідком атаки, так і неправильної конфігурації. Підозрілі події необхідно досліджувати настільки, наскільки дозволяють це робити наявні ресурси.

Дослідження підозрілих подій може бути дуже трудомістким завданням. Нерідко представляється розумним пропустити деякі із цих подій або просто передати інформацію мережному або системному адміністраторові.

**Дослідження підозрілих подій**

При виникненні підозрілих дій варто виконати процедуру, що складається з наступних кроків, щоб визначити, чи є дана дія вдалим вторгненням або спробою проникнення, або вона носить нешкідливий характер. Отже, потрібно виконати наступні кроки.

1. Ідентифікувати систему.

2. Записувати в журнал відомості про додатковий трафік між джерелом і пунктом призначення.

3. Записувати в журнал весь трафік, що виходить із джерела.

4. Записувати в журнал уміст пакетів із джерела.

При виконанні кожного кроку необхідно визначати, чи досить очевидних ознак для з'ясування того, чи є дана дія атакою. У наступних розділах приводиться опис даних кроків.

При дослідженні події необхідно мати на увазі наступний момент. Якщо подія відбувається один раз і більше не повторюється, то дуже важко одержати яку-небудь додаткову інформацію (крім того, звідки надійшов трафік). Одиночні аномалії досліджувати практично неможливо.

**1. Ідентифікація систем**

Першим кроком при дослідженні підозрілої активності є ідентифікація систем, що беруть участь у дії. Ця процедура може полягати в перетворенні IP-адрес в імена вузлів. У деяких випадках ім'я вузла знайти не вдається (система не має записи DNS; це клієнт DHCP; віддалений DNS-сервер перебуває в неактивному стані й т.д.). Якщо пошук DNS кінчається невдачею, то варто спробувати ідентифікувати вузол іншими способами, наприклад, пошуком у реєстрі American Registry of Internet Numbers (ARIN) за адресою http://www.arin.net/, в Internic за адресою http://www.networksolutions.com/ або в інших каталогах Інтернету. Утиліти, такі як Sam Spade (перебувають за

адресою http://samspade.org/), також допоможуть у цьому випадку. Неможливість ідентифікації джерела або пункту призначення підозрілих дій не є достатнім доказом того, що подія в дійсності є атакою. Аналогічно, успішна ідентифікація систем не є достатнім доказом "необразливості" виявлених дій.

Джерело підозрілого трафіку може не бути безпосереднім джерелом атаки. Спроби проведення атаки на відмову в обслуговуванні, як правило, проводяться з підміненими вихідними адресами, і спроби несанкціонованого доступу або зондування можуть виходити з інших систем, захоплених зловмисником.

**2. Запис у журнал додаткового трафіку між джерелом і пунктом призначення**

Одна-єдина окрема подія (наприклад, порушення протоколу IP) може не представляти повну інформацію про трафік між двома системами. Іншими словами, необхідно розуміти контекст підозрілих дій. Гарним прикладом тут служить ознака атаки Sendmail WIZ. Ця ознака ідентифікує спробу використання команди WIZ у програмі Sendmail. Дана подія безпеки ідентифікує будь-яке входження команди WIZ у повідомленні. Якщо команда WIZ є присутнім у тілі повідомлення, то це виразно не спроба вторгнення. Розуміння контексту події допомагає визначати помилкові спрацьовування.

Налаштуйте IDS на відстеження всього трафіку між джерелом підозрілої активності й пунктом призначення. Як приклад використовуйте таблицю 6.1.

Таблиця 6.1 – Приклад конфігурації IDS із записом у журнал усього трафіку між двома системами

| **Ім'я події** | **Дія** | **IP-адреса джерела** | **IP-адреса пункту призначення** | **Протокол** | **Порт джерела** | **Кінцевий порт** |
|---|---|---|---|---|---|---|
| SUS_ACT | Повідомлення, занесення в журнал | Джерело підозрілої активності | Пункт призначення підозрілої активності | TCP,UDP і/або ICMP, залежно від типу виявленої | Будь-який | Будь-який |

| | | | | | | |
|---|---|---|---|---|---|---|
| | | | | активності | | |

Тепер задамося питанням, що ж це все нам дає. По-перше, ми одержуємо подання про інший трафік, що має місце між джерелом і пунктом призначення. Якби пакет WIZ був єдиним трафіком між двома системами, із цього можна було зробити вивід про те, що це схоже на спробу проникнення в систему. Навпроти, якщо спостерігається велике число трафіку SMTP (пошти) між двома системами, то, швидше за все, це звичайний легітимний поштовий трафік.

**3. Запис у журнал усього трафіку із джерела**

Маючи на увазі, що даних, фіксуємих за допомогою запису всього трафіку між двома системами, недостатньо для визначення того, чи є активність легітимною, можна почати збір іншого трафіку, що надходить із джерела. Майте на увазі, що обсяг цього трафіку може бути обмеженим. Якщо джерело підозрілої активності перебуває в деякій віддаленій мережі, то буде спостерігатися тільки трафік, що надходить на ваш вузол. Якщо ж джерело локальне, то можливий збір усього трафіку з даного комп'ютера, що дасть набагато більше широке подання про те, що ж насправді відбувається.

Щоб почати збір усього трафіку із джерела, налаштуйте детектор IDS на збір всієї інформації з підозрілого джерела. Приклад такої конфігурації наведений у таблиці 6.2.

Таблиця 6.2 – Приклад конфігурації IDS, призначеної для занесення в журнал усього трафіку, що виходить із визначеної адреси джерела

| **Ім'я події** | **Дія** | **IP-адреса джерела** | **IP-адреса пункту призначення** | **Протокол** | **Порт джерела** | **Кінцевий порт** |
|---|---|---|---|---|---|---|

| | | | | | | |
|---|---|---|---|---|---|---|
| SUS_SRC | Повідомлення, запис у журнал | Джерело підозрілих дій | Будь-який | TCP,UDP і/або ICMP, залежно від типу виявленої активності | Будь-який | Будь-який |

Така конфігурація, як правило, генерує деяку інформацію, що не представляє якої-небудь цінності для дослідження. Доти, поки можливо об'єктивну оцінку інформації, даний журнал можна використовувати для складання докладної картини взаємодій, що відбуваються, що мають місце між джерелом і пунктом призначення. Спробуйте вникнути в суть спостережуваної активності. Чи є спостережувана активність веб-трафіком? чи Виходить трафік з підозрілого джерела, або ж його джерелом є ваш вузол?

На даному етапі дослідження повинна бути відома наступна інформація.

Ім'я системи-джерела.

Тип і частота трафіку, обмін яким відбувається між джерелом і пунктом призначення.

Тип і частота трафіку, обмін яким відбувається між джерелом і будь-якими іншими системами вашого вузла.

Ця інформація забезпечує досить докладне подання про природу підозрілого трафіку. Проте, ступінь очевидності що відбуває може не сказати про те, чи є спостережувана активність спробою атаки.

**4. Запис у журнал умісту пакетів із джерела**

Кінцевим кроком проведеного дослідження є запис у журнал умісту пакетів, що виходять із джерела. Варто помітити, що даний підхід корисний тільки при роботі з текстовими протоколами, такими як telnet, FTP, SMTP і HTTP (до деякої міри). Якщо використовуються двійкові протоколи або протоколи із шифруванням, даний підхід зовсім марний. Для реалізації

описаного методу необхідно змінити конфігурацію IDS, як показано в таблиці 6.3.

За допомогою занесення в журнал вмісту пакетів можна скласти повний запис сеансу, а також зафіксувати команди, призначення, що відправляються безпосередньо в пункт.

Після фіксування деяких даних необхідно переглянути знайдену інформацію. Чи позначає сеанс потенційну атаку, або ж все виглядає в межах припустимого? Скомбінувавши ці дані з іншою знайденою інформацією, можна знайти відповідь на це питання. Якщо цього зробити не вдалося, спробуйте знайти людину, у якого є досвід роботи з досліджуваним протоколом.

Таблиця 6.3 – Приклад конфігурації IDS, що здійснює перехоплення вмісту пакетів

| **Ім'я події** | **Дія** | **IP-адреса джерела** | **IP-адреса пункту призначення** | **Протокол** | **Порт джерела** | **Кінцевий порт** |
|---|---|---|---|---|---|---|
| SUS_ACT | Повідомлення, запис у журнал умісту пакета | Джерело підозрілої активності | Пункт призначення підозрілої активності | TCP або UDP | Будь-який | Порт, на який спрямований підозрілий трафік |
| SUS_ACT | Повідомлення, запис у журнал умісту пакета | Пункт призначення з підозрілої активності | Джерело підозрілої активності | TCP або UDP | Порт, на який спрямований підозрілий трафік | Будь-який |

**Система запобігання вторгнень** (англ. *Intrusion Prevention System*) – програмна або апаратна система мережної й комп'ютерної безпеки, що виявляє вторгнення або порушення безпеки й автоматично захищаюча від них.

У цілому, IPS аналогічні IDS. Головна ж відмінність полягає в тому, що вони функціонують у реальному часі й можуть в автоматичному режимі блокувати мережні атаки. Кожна IPS містить у собі модуль IDS.

**HIPS (***Host-Based Intrusion Prevention System, англ. система запобігання вторгнень***)** – проактивна технологія захисту, побудована на аналізі поводження.

**Принцип роботи**

У силу того що HIPS є засобом проактивного захисту, програми даного класу не містять бази даних сигнатур вірусів (однак можуть їх задіяти, скажемо HIPS у складі антивірусу може блокувати запуск відомих шкідливих програм незалежно від включення або відключення файлового монітора) і не здійснює їх детектування. HIPS-продукти здійснюють аналіз активності програмного забезпечення й всіх модулів системи й блокування потенційне небезпечних дій у системі користувача. Аналіз активності здійснюється за рахунок використання перехоплювачів системних функцій або установці т.зв. міні-фільтрів. Слід зазначити, що ефективність HIPS може доходити до 100%, однак більшість програм цього класу жадають від користувача високого рівня кваліфікації для грамотного керування антивірусним продуктом.

***Види HIPS:***

Класичні HIPS. Класичні HIPS-продукти надають користувачеві інформацію про активність того або іншого додатка, однак рішення про дозвіл/заборони тої або іншої операції повинен приймати користувач, таким чином класичні HIPS-продукти дозволяють користувачеві тонко налаштувати ті або інші правила контролю, але створення правил вимагає високої кваліфікації користувача.

Експертні HIPS. На відміну від класичних HIPS-продуктів, експертні HIPS можуть самостійно ухвалювати рішення щодо блокуванні тієї або іншої активності, виходячи із правил і алгоритмів, закладених розроблювачем продукту. Для використання експертних HIPS-продуктів користувачеві не обов'язково мати визначену кваліфікацію, однак експертні HIPS-продукти в ряді випадків можуть блокувати легітимну активність користувальницького програмного забезпечення, або можуть не визнати дану активність програми за

**Яким чином можна запобігти вторгненням за допомогою системи IDS**

Щоб запобігти вторгненню, необхідно або зупинити здійснювану атаку перед її досягненням системи-жертви, або зупинити дію атаки перед виконанням на системі-жертві коду, що використовує уразливість.

Механізм запобігання атаки легше всього розглядати на вузлі, що використовує HIDS. Наприклад, можна використовувати аналізатори системних викликів або поводження додатка. Якщо виклик додатка схожий на атаку, аналізатор системних викликів запобіжить виконанню виклику операційною системою. Якщо додаток намагається виконати неавторизовану операцію, аналізатор поводження додатка запобіжить її виконанню. В обох випадках HIDS запобігає атаці.

Процес запобігання атаки за допомогою NIDS є більше складним. У стандартній конфігурації NIDS датчик розташовується в тому місці, з якого він може відслідковувати трафік (див. Рисунок 8.2, лекція 8). При надходженні через канал зв'язку даних атаки, датчик перехоплює пакет і починає його аналізувати. У деякий момент датчик визначає, що пакет являє собою атаку, і вживає дію. Ця дія, як правило, полягає в закритті з'єднання (тільки якщо атака проводиться через з'єднання TCP) або в переналаштовуванні міжмережного екрана для блокування подальшого трафіку із джерела.

На жаль, у випадку з NIDS час працює не на користь досягнення цілі. Під час аналізу пакета датчиком пакет продовжує свій рух по мережі. У більшості випадків пакет досягає цілі ще перед закриттям з'єднання або виконанням дій

по переналаштовуванню міжмережного екрана. Отже, найчастіше атака випереджає дії датчика по її запобіганню.

Закриття з'єднання або блокування трафіку з атакуючої системи може знизити рівень ушкодження системи, але не запобіжить впливу на неї зловмисника.

Для запобігання за допомогою NIDS успішного проведення атак на систему рішення по пакеті повинне прийматися до того, як пакет досягне системи-цілі. Це означає, що архітектуру системи NIDS потрібно змінити таким чином, щоб датчик NIDS був розташований на одному каналі зв'язку із трафіком (як міжмережний екран), а не просто стежив за минаючим мимо трафіком (див. рисунок 6.3).

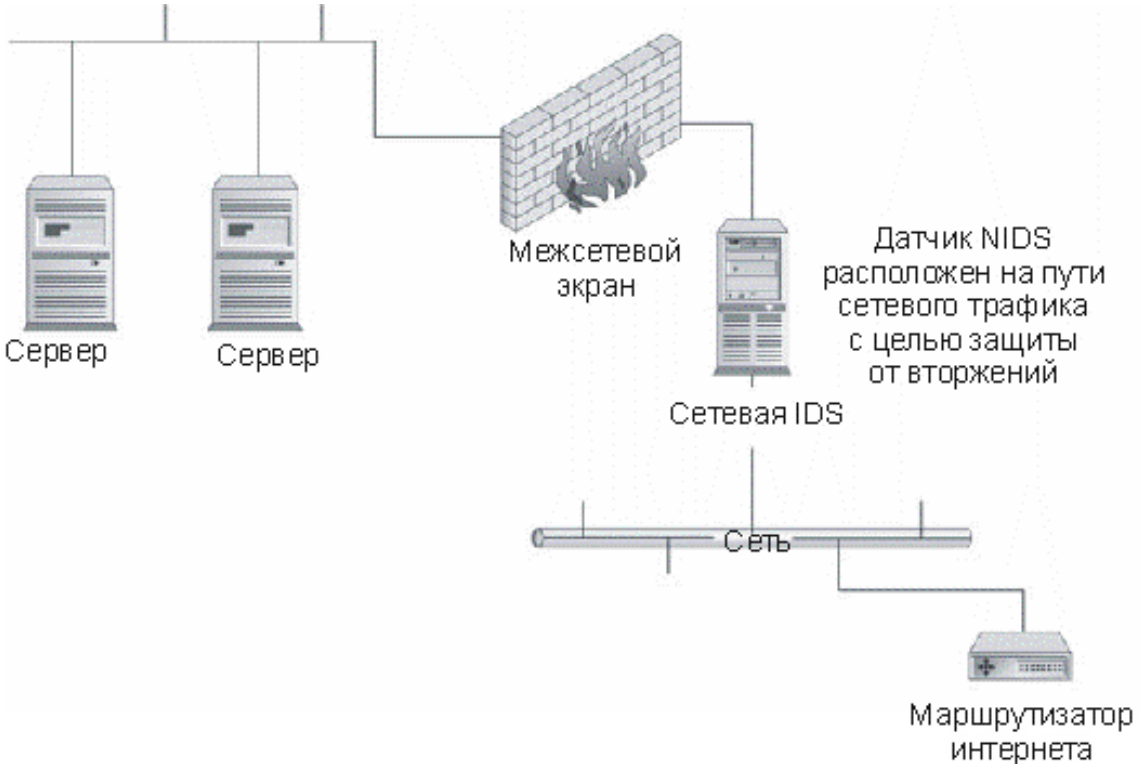



Рисунок 6.3 – Конфігурація, необхідна для запобігання атак датчиком NIDS

Розглянута архітектура не є єдино можливою. Також можливо розташувати датчик NIDS на міжмережному екрані або реалізувати його тісний взаємозв'язок з міжмережним екраном, щоб останній не пропускав трафік без дозволу датчика NIDS.

**Проблеми, пов'язані з виявленням вторгнень**

Заміна реактивної природи IDS на превентивну створює деякі проблеми. Дійсно, після цієї зміни виникають два серйозних питання: потенційна можливість відмови в обслуговуванні й недостатній середній рівень доступності.

**Відмова в обслуговуванні**

При запобіганні вторгнень головним механізмом обробки більше не є повідомлення системи, мережі й системних адміністраторів. Тепер "ядром" системи є блокування спроби виконання дії. Коли IDS блокує атаку, вона запобігає виконанню дії, будь то системний виклик, операція додатка або мережне з'єднання. Дане блокування запобігає атаці. Очевидно, при цьому мається на увазі коректна ідентифікація системою IDS дії як атаки.

Якщо дія, спроба якого була здійснена, насправді не було атакою, а IDS заблокувала його, те, можливо, IDS заблокувала законну дію, виконувана в інформаційному середовищі. Внаслідок цього IDS може викликати відмову в обслуговуванні. Якщо дію, що викликала проблему, являло собою деяку аномалію (наприклад, пакет з помилками), то повторна передача пакета або повторна установка з'єднання, як правило, здійснюються успішно. Проте, якщо IDS некоректно ідентифікує легітимні дії або трафік, приймаючи їх за атаки, то швидше за все, відмова в обслуговуванні буде відбуватися й надалі.

**Завдання:**

Використовуючи програмне забезпечення, розроблене у ході виконання лабораторних робіт № 2 та № 3, реалізувати систему виявлення вторгнень у мережу або на хост, або візуалізувати роботу системи виявлення вторгнень.

## Список використаної літератури


1. Хорошко В.А. Методы и средства защиты информации / Хорошко В.А., Чекатков А.А. / Под ред. Ю.С. Ковтанюка. – К.: Юниор, 2003. – 504 с.
2. Термінологічний довідник з питань технічного захисту інформації / Коженевський С.Р., Кузнецов Г.В., Хорошко В.О., Чирков Д.В. / За ред. проф. В.О. Хорошка. – К.: ДУІКТ, 2007. – 365 с.
3. Макс Ронге. Разведка и контрразведка / М. Ронге /. – К.: СИНТО, 1993. – 239 с.
4. Мухачев В.А. Методы практической криптографии / Мухачев В.А., Хорошко В.А./ . – К.: ПолиграфКонсалтинг, 2005. – 214 с.
5. Мицан И.Б. Стеганографические методы сокрытия информации / Мицан И.Б. // Специальная техника и вооружение. Научно-методическое издание. – К., № 1 – 5, 2001. – С. 28 – 32.
6. Хорошко В.О. Основи комп'ютерної стеганографії. Навчальний посібник / В.О. Хорошко, О.Д. Азаров, М.Є. Шелест, Ю.Є. Яремчик /. – Вінниця: ВДТУ, 2003. – 143 с.
7. Конахович Г.Ф. Компьютерная стеганография. Теория и практика / Конахович Г.Ф., Пузыренко А.Ю. /. – К.: «МК-Пресс», 2006. – 288 с.
8. Безопасность информационных технологий. Методология создания систем защиты/ В.В. Домарев. – К.: ООО "ТИД "ДС", 2001. – 688 с.
9. Энциклопедия промышленного шпионажа/ Под общ. ред. Е.В. Куренкова – С.Петербург: ООО "Изд-во Полигон", 1999. – 512 с.
10. Хорев А;А. Способы и средства защиты информации. – М.: МО РФ, 2000. -316 с.
11. А.Ю.Щербаков. Введение в теорию и практику компьютерной безопасности. -М.: издатель Молгачева С.В., 2001.
12. Бармен, Скотт. Разработка правил информационной безопасности.: Пер. с англ. – М.: Издательский дом "Вильямс", 2002.
13. С.Л.Емельянов Основы информационной безопасности.– Одесса:

Юридична література, 2003.-198с.

14. Про державну таємницю. Закон України №3855-XII від 21.01.1994 р. (в редакції Закону № 1079-14 від 21.09.99).

15. Про затвердження зводу відомостей, що становлять державну таємницю. Наказ Голови Служби безпеки України від 12.08.2005 р. № 440.

16. НД ТЗІ 1.1 – 002 – 99. Общие положения по защите информации в компьютерных системах от несанкционированного доступа. Нормативний документ ДСТЗИ СБ Украйни. Киев, 1999.

17. Про інформацію. Закон України № 2657-ХП від 02.10.92 р.

18. Концепція технічного захисту інформації в Україні. Постанова КМУ №1126 8.10.97.

19. Положення про технічний захист інформації в Україні. Указ Президента України №1229/99 від 27.09.99.

20. Тимчасові рекомендації з технічного захисту інформації вид витоку каналами побічних електромагнітних випромінювань і наводок. (ТР ТЗІ-ПЕМВН-95). Затверджені наказом ДСТЗІ від 09.06.95р. № 25.

21. НД ТЗІ 1.4-001-2000. Типове положення про службу захисту інформації в автоматизованій системі.

22. НД ТЗІ 2.5-005-99. Класифікація автоматизованих систем і стандартні функціональні профілі захищеності оброблюваної інформації від несанкціонованого доступу.

23. НД ТЗІ 2.1-001-2001. Створення комплексів технічного захисту інформації. Атестація комплексів. Основні положення.

24. НД ТЗІ 3.7-001-99. Методичні вказівки щодо розробки технічного завдання на створення комплексної системи захисту інформації в автоматизованій системі (Зі зміною № 1, затвердженою наказом ДСТСЗІСБУ 18.06.02 № 37).

25. НД ТЗІ 2.5-010-2003. Вимоги до захисту інформації \№ЕВ-сторінки від несанкціонованого доступу.

26. 13.ГОСТ Р 51275-99. Защита информации. Объект информатизации.

Факторы, воздействующие на информацию. Общие положения.

27. http://www.intuit.ru/department/security/secbasics/

28. Галатенко В.А. Основы информационной безопасности Интернет-университет информационных технологий – ИНТУИТ.ру, 2008

29. Лапонина О.Р. Основы сетевой безопасности: криптографические алгоритмы и протоколы взаимодействия Интернет-университет информационных технологий – ИНТУИТ.ру, 2005

30. Галатенко В.А. Стандарты информационной безопасности Интернет-университет информационных технологий – ИНТУИТ.ру, 2005

31. Э. Мэйволд Безопасность сетей Эком, 2006

32. Хаулет Т. Защитные средства с открытыми исходными текстами БИНОМ. Лаборатория знаний, Интернет-университет информационных технологий – ИНТУИТ.ру, 2007

33. Department of Defense Trusted Computer System Evaluation Criteria DoD 5200.28-STD, 1993.

34. Information Technology Security Evaluation Criteria (ITSEC). Harmonized Criteria of France – Germany – the Netherlands – the United Kingdom Department of Trade and Industry, London, 1991.

35. Security Architecture for Open Systems Interconnection for CCITT Applications. Recommendation X.800 CCITT, Geneva, 1991.

36. Site Security Handbook. Holbrook P., Reynolds J., Request for Comments: 1244, 1991.

37. James Nechvatal, Elaine Barker, Lawrence Bassham, William Burr, Morris Dworkin, James Foti, Edward Roback Report on the Development of the Advanced Encryption Standard (AES) Computer Security Division Information Technology Laboratory National Institute of Standards and Technology Technology Administration U.S. Department of Commerce. 2000г. 116с.

38. Государственный Стандарт Российской Федерации Информационная технология. Криптографическая защита информации. Процедуры выработки и

проверки электронной цифровой подписи на базе асимметричного криптографического алгоритма 1994г.

39. Государственный Стандарт Российской Федерации Информационная технология. Криптографическая защита информации. Функция хэширования 1994г.

40. RFC 3280 Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile 2002г. 129с.

41. RFC 3281 An Internet Attribute Certificate Profile for Authorization 2002г. 40с.

42. RFC 2510 Internet X.509 Public Key Infrastructure Certificate Management Protocols 1999г. 72с.

43. RFC 2511 Internet X.509 Certificate Request Message Format 1999г. 25с.

44. RFC 3369 Cryptographic Message Syntax 2002г. 60с.

45. RFC 2560 X.509 Internet Public Key Infrastructure Online Certificate Status Protocol – OCSP 1999г. 23с.

46. RFC 2797 Certificate Management Messages over CMS 2000г. 47с.

47. RFC 3379 Delegated Path Validation and Delegated Path Discovery Protocol Requirements 2002г. 15с.

48. RFC 2633 S/MIME Version 3 Message Specification 1999г. 32с.

49. RFC 2632 S/MIME Version 3 Certificate Handling 1999г. 13с.

50. Security Architecture for the Internet Protocol RFC 2401 1998г. 66с.

51. Internet Security Association and Key Management Protocol (ISAKMP) RFC 2408 1998г. 86с.

52. The Internet Key Exchange (IKE) RFC 2409 1998г. 41с.

53. RFC 2412 The OAKLEY Key Determination Protocol 1998г. 55с.