Міністерство освіти і науки України
Вінницький національний технічний університет

**С. М. Захарченко, Т. І. Трояновська, О. В. Бойко**

# ОСНОВИ ПОБУДОВИ ЗАХИЩЕНИХ МЕРЕЖ НА БАЗІ ОБЛАДНАННЯ КОМПАНІЇ CISCO

**Навчальний посібник**

Вінниця
ВНТУ
2017

УДК 004.7(075)
З-38

Рекомендовано до друку Вченою радою Вінницького національного технічного університету Міністерства освіти і науки України (протокол № 16 від 22 червня 2017 р.)

Рецензенти:
**В. А. Лужецький,** доктор технічних наук, професор
**С. І. Перевозніков,** доктор технічних наук, професор
**Л. Б. Ліщинська,** доктор технічних наук, професор

**Захарченко, С. М.**

З-38 Основи побудови захищених мереж на базі обладнання компанії Cisco : навчальний посібник / С. М. Захарченко, Т. І. Трояновська, О. В. Бойко. – Вінниця : ВНТУ, 2017. – 136 с.

У навчальному посібнику розглянуто особливості побудови захищених комп'ютерних мереж різного типу на основі використання обладнання компанії Cisco.

Видання складається з восьми теоретичних розділів та лабораторного практикуму. Матеріал розташовано в логічній послідовності та із зазначенням необхідних команд для виконання налаштування обладнання компанії Cisco.

Навчальний посібник призначено для студентів спеціальностей 123 – «Комп'ютерна інженерія» та 125 – «Кібербезпека».

УДК 004.7 (075)

**ЗМІСТ**

# ВСТУП

Підготовка фахівців технічного напрямку, зокрема, професіоналів у галузі організаціїї безпеки комп'ютерних мереж, потребує змістовного теоретичного підґрунтя та водночас закріплення широкого спектру практичних навиків для успішного працевлаштування на сучасному ринку праці.

Комп'ютерні системи, мережі й технології забезпечують увесь інформаційний світ своїми послугами та потребують висококваліфіковаинх фахівців для роботи з обладнанням.

Компанія Cisco є світовим лідером у галузі мережевих технологій, що задовольняють потреби людського спілкування, корпоративного зв'язку, віддаленої співпраці.

Нині мережі стали найважливішим елементом бізнесу, освіти, державного управління та домашніх комунікацій. IP-рішення, програмні та апаратні засоби, що їх пропонує та застосовує компанія Cisco, можна впевнено назвати фундаментальною основою сучасних мереж.

Компанія Cisco Systems була заснована в 1984 році невеликою групою вчених зі Стенфордського уінверсітету. Із самого початку інженери Cisco стали лідерами з розробки мережевих технологій, заснованих на протоколі IP (Internet Protocol). А нині понад 60 тисяч співробітників компанії, які працюють по всьому світу, продовжують традиції новаторства і розробляють кращі в галузі продукти й рішення в базових для Cisco галузях (комутація та маршрутизація), а також у сфері сучасних технологій, до переліку яких входять IP-комунікації, безпровідні мережі LAN, мережі зберігання (SAN), домашні мережі, відеосистеми, прикладні мережеві послуги та мережева безпека.

Cisco була ініціатором багатьох історичних змін у сфері технологій, і сьогодні вона продовжує цю традицію. Сьогодні, коли галузь високих технологій знову переживає період великих змін, Cisco зберігає лідерство в різних сегментах таких, як: маршрутизація, комутація, уніфіковані комунікації, бездротовий зв'язок і безпека.

Цей навчальний посібник розраховано на студентів вищих навчальних закладів зі спеціальностями 123 – «Комп'ютерна інженерія» та 125 – «Кібербезпека» і допоможе ґрунтовно підготувати фахівців з мережевої безпеки для роботи на обладнанні компанії Cisco.

Видання містить актуальний теоретичний та практичний навчальний матеріал і складається з восьми розділів та лабораторного практикуму. Важливою особливістю цього посібника є послідовне викладення інформації та наведення достатнього обсягу команд для налаштування мережевого обладнання й операційної системи Cisco IOS.

# 1 ОСНОВИ МЕРЕЖЕВОЇ БЕЗПЕКИ

## 1.1 Еволюція мережевої безпеки

У липні 2001 черв'як Code Red здійснив глобальну атаку веб-серверів, інфікувавши близько 350 000 хостів. Він не тільки заблокував доступ до інфікованих серверів, а й заподіяв шкоду локальним мережам, у яких знаходились ці сервери, зробивши їх дуже повільними або взагалі непрацюючими. Code Red здійснив DoS атаку, результати якої відчули мільйони користувачів. Спеціалісти із захисту мереж, відповідальні за інфіковані Code Red сервери розробили і впровадили політику безпеки, відповідно до якої оновлення системи захисту має відбуватись постійно.

Мережева безпека безпосередньо стосується функціонування будь-яких бізнес структур. Недоліки в мережевій безпеці можуть зруйнувати електронну комерцію, призвести до втрати ділової та приватної інформації. Ці «дірки» у системі безпеки можуть призвести до втрати прибутків, викрадення інтелектуальної власності, і навіть становити загрозу державній безпеці.

Після появи перших вірусів та здійснення перших DoS атак змінились і функції спеціалістів з комп'ютерних мереж. Щоб задовольнити потреби кінцевих користувачів, мережеві фахівці засвоюють правила побудови захищених мереж. Основну увагу багатьох мережевих професіоналів тепер спрямовано не на проектування, побудову та масштабування мережі, а саме на захист існуючої мережі.

Сьогодні Internet дуже сильно відрізняється від тієї мережі, що була в 60-х роках. Робота мережевих професіоналів також передбачає перелік дій, які дають змогу гарантувати, що персонал (який відповідає за мережеву безпеку) є достатньо кваліфікованим і розуміється на сучасному обладнанні, технологіях і протоколах захисту даних. Важливо, щоб професіонал з мережевої безпеки вів постійний моніторинг мережі для постійного спостереження за появою нових загроз та небезпек.

Мережева безпека в сучасних мережах реалізовується спеціальними пристроями та засобами. Одним із перших засобів мережевої безпеки була система виявлення зловмисника (intrusion detection system (IDS) і була вперше розроблена SRI International у 1984. IDS у режимі реального часу забезпечує виявлення певного типу атак під час їхньої дії. Це дозволяє мережевим професіоналам оперативно реагувати і таким чином зменшувати негативний вплив цих атак на мережеві пристрої та користувачів. Наприкінці 90-х IDS стали замінювати на системи запобігання зловмисникам (intrusion prevention system (IPS). Ці системи дають можливість у режимі реального часу не тільки виявляти факт атаки, а й автоматично її блокувати.

На додаток до IDS та IPS були розроблені так звані firewall, які не пропускають у внутрішню мережу небажаний трафік, що надходить з небезпечних мереж, таким чином забезпечують захист периметра.

У 1988 році, Digital Equipment Corporation (DEC) створила перший мережевий firewall у вигляді пакетного фільтра. Ці перші firewall перевіряли пакети на предмет відповідності певному набору правил з подальшим просуванням або відкиданням пакета. Пакетні фільтри аналізували кожний пакет окремо, не враховуючи, що пакет передається в межах певного з'єднання. У 1989 році AT&T Bell Laboratories розробила так званий stateful firewall. Аналогічно firewall, що базується на фільтрації пакетів, він використовує набір попередньо визначених правил для фільтрації трафіка, однак, на відміну від останнього, він відслідковує встановленні з'єднання та визначає, чи належить пакет до існуючого потоку даних. Це дає змогу забезпечити вищий рівень безпеки і швидшу обробку.

Спочатку firewall реалізовувались програмно як додаткова функція в існуючому мережевому обладнанні, зокрема, у маршрутизаторах. Через певний час було розроблено самостійні пристрої, які звільняли маршрутизатори й комутатори від процедури фільтрації, що суттєво завантажувала ресурси останніх. В організаціях, які не потребують виділеного firewall, сучасні маршрутизатори Cisco Integrated Services Router (ISR) можуть використовуватись як stateful firewalls.

Окрім захисту мережі від зовнішніх загроз, мережеві професіонали мають бути готовими до загроз із середини мережі. Внутрішні загрози свідомі чи випадкові можуть нанести значно більшу шкоду, порівняно із зовнішніми, оскільки в цьому випадку є прямий доступ до інформації про структуру корпоративної мережі та корпоративних даних. Не дивлячись на це, засоби й технології запобігання внутрішнім загрозам з'явились майже на 20 років пізніше.

Дані, що передаються через бездротові канали, використовують різні методи криптографічного захисту. Обмін даними за допомогою IP-телефонії має також шифруватись. Файли на комп'ютері можуть також зберігатись у зашифрованому вигляді. На сьогоднішній день має місце тенденція, що всі комунікації мають бути захищеними за допомогою шифрування. Криптографія забезпечує конфіденційність даних, що є одним з трьох компонентів інформаційної безпеки: конфіденційності, цілісності, доступності. Інформаційна безпека має справу із захистом інформації та інформаційних систем від неавторизованого доступу, використання, розкриття, руйнування модифікації або знищення. Шифрування забезпечує конфіденційність шляхом приховування відкритих даних. Цілісність даних передбачає їхню незмінність протягом різних операцій і забезпечується механізмами хешування. Доступність даних – це гарантія доступу до даних, яка реалізується підвищенням надійності мережі та процедурами резервного копіювання даних.

### 1.2 Базові поняття інформаційної безпеки

Під інформаційною безпекою розуміють стан захищеності інформаційної системи, включаючи саму інформацію та інфраструктуру, що її підтримує. Інформаційна система перебуває у захищеному стані, якщо забезпечено її конфіденційність, доступність і цілісність.

**Конфіденційність** (confidentiality) – це гарантія того, що секретні дані будуть доступні тільки для тих користувачів, які мають дозвіл на доступ; такі користувачі називаються легальними або авторизованими.

**Доступність** (availability) – це гарантія того, що авторизовані користувачі завжди зможуть отримати доступ до даних.

**Цілісність** (integrity) – це гарантія того, що дані зберігають вірні значення. Ця гарантія забезпечується шляхом заборони неавторизованим користувачам будь-яким чином змінювати, модифікувати, знищувати або створювати дані.

Поняття конфіденційності, доступності і цілісності можна віднести не тільки до інформації, а й до інших ресурсів обчислювальної мережі – мережне обладнання або додатки.

Для опису дій, пов’язаних із несанкціонованим доступом до інформаційної системи використовуються такі поняття, як: загроза, атака та ризик.

**Загроза** – це будь-яка дія, що може бути спрямована на порушення інформаційної безпеки системи. **Атака** – реалізована загроза. **Ризик** – вірогідна оцінка розміру можливих збитків, яких може зазнати власник інформаційного ресурсу в результаті успішно проведеної атаки.

Загрози можуть надходити як від легальних користувачів мережі, так і від зовнішніх зловмисників. Останнім часом більше половини від загальної кількості найбільш серйозних інцидентів, пов’язаних з безпекою, становлять порушення з боку легальних користувачів мереж.

Загрози з боку легальних користувачів поділяються на навмисні та ненавмисні. До навмисних загроз належать: моніторинг системи з метою отримання персональних даних інших співробітників (ідентифікаторів, паролів) або конфігураційних параметрів обладнання;отримання доступу до конфіденційних даних, що зберігаються на серверах і робочих станціях підприємства з метою їхнього викрадення, спотворення або знищення; виведення з ладу мережевого програмного забезпечення та обладнання; порушення персоналом правил, що регламентують роботу користувачів в мережі підприємства: відвідування заборонених веб-сайтів, винесення за межі підприємства змінних носіїв інформації, недбале зберігання паролів та інші схожі порушення режиму.

Під час проведення атаки зловмиснику важливо завдати шкоду атакованому об’єкту, але й знищити всі сліди своєї участі в цьому. Одним з основних прийомів, який використовується зловмисниками для приховування слідів, є підміна вмісту пакетів (spoofing), зокрема адресної частини.

### 1.3 Методи і засоби інформаційної безпеки

**Інформаційна безпека** – це стан захищеності потреб інформації особистістю, суспільством і державою, при якому забезпечується їхнє існування і прогресивний розвиток незалежно від наявності внутрішніх і зовнішніх інформаційних загроз.

Варто наголосити, що задоволення потреб в інформації призводить до оволодіння відомостями про навколишній світ і процеси, які є в ньому, тобто інформованості особистості, суспільства та держави. Стан інформованості визначає ступінь адекватності сприйняття суб'єктами навколишньої дійсності і, як наслідок, – обґрунтованість рішень та дій, що приймаються.

Інформаційну безпеку можна поділити на такі поняття щодо забезпечення стану захищеності: особистості, суспільства, держави від впливу неякісної інформації; інформації та інформаційних ресурсів від несанкціонованого впливу сторонніх осіб; інформаційних прав і свобод людини й громадянина.

**Інформаційна безпека** – це одна зі сторін розгляду інформаційних відносин у межах інформаційного законодавства з позицій захисту життєво важливих інтересів особистості, суспільства, держави та акцентування уваги на загрозах цим інтересам і на механізмах усунення або запобігання таким загрозам правовими методами.

**Інформаційне середовище** – сфера діяльності суб'єктів, пов'язана зі створенням, перетворенням і використанням інформації. Інформаційне середовище умовно поділяється на частини: створення і розповсюдження інформації; формування інформаційних ресурсів, підготовка інформаційних продуктів, надання інформаційних послуг; споживання інформації; створення і застосування інформаційних систем, інформаційних технологій і засобів їхнього забезпечення; створення і застосування засобів/механізмів інформаційної безпеки.

**Об'єктами інформаційної безпеки** є свідомість, психіка людей; інформаційні системи та мережі різного масштабу та різного призначення. До соціальних об'єктів інформаційної безпеки належать: особистість, колектив, суспільство, державу, світове товариство.

**Суб'єкти інформаційної безпеки** – держава, що здійснює свої функції через відповідні органи; громадяни; суспільні або інші організації та об'єднання, що володіють повноваженнями із забезпечення інформаційної безпеки відповідно до законодавства.

Інтереси особистості в інформаційній сфері полягають: у реалізації конституційних прав людини та громадянина на доступ до інформації, на використання інформації в інтересах здійснення незабороненої законом діяльності, фізичного, духовного та інтелектуального розвитку; у захисті інформації, що забезпечує особисту безпеку.

**Інформаційне забезпечення інформаційної безпеки** охоплює збирання (добування) відомостей про дестабілізуючі фактори та інформаційні загрози, їхнє оброблення, обмін інформацією між органами керування й силами та засобами системи інформаційної безпеки.

**Інформаційний захист** досягається шляхом внесення в порядку законодавчої ініціативи законопроектів, здійснення судового захисту, проведення оперативних заходів силами й засобами інформаційної безпеки.

**Інформаційна зброя** – сукупність спеціально організованої інформації та інформаційних технологій, яка дозволяє цілеспрямовано змінювати (знищувати, спотворювати), копіювати, блокувати інформацію, долати системи захисту, обмежувати допуск законних користувачів, здійснювати дезінформацію, порушувати функціонування носіїв інформації, дезорганізовувати роботу технічних засобів комп'ютерних систем та інформаційно-обчислювальних мереж, що застосовується під час інформаційної війни (боротьби) для досягнення поставлених цілей.

За метою використання така інформаційна зброя поділяється на інформаційну зброю атаки та інформаційну зброю забезпечення.

**Інформаційна зброя комп'ютерних систем та мереж** – інформаційна зброя, за допомогою якої здійснюється вплив на інформацію, що зберігається, оброблюється і передається в інформаційно-комунікаційних системах та мережах і (або) порушуються інформаційні технології, що застосовуються в комп'ютерних системах та мережах.

У складі інформаційної зброї комп'ютерних систем та мереж можна виділити чотири основних види засобів інформаційних впливів:

- засоби порушення конфіденційності інформації;
- засоби порушення цілісності інформації;
- засоби порушення доступності інформації;
- засоби психологічного впливу на абонентів мереж.

За способом реалізації інформаційну зброю поділяють на три великі класи: інформаційна алгоритмічна (математична) зброя; інформаційна програмна зброя; інформаційна апаратна зброя.

**Інформаційна алгоритмічна (математична) зброя** – це вид інформаційної зброї, до якого, звичайно, належать:

- алгоритми, що використовують сполучення санкціонованих дій для здійснення несанкціонованого доступу до інформаційних ресурсів;
- алгоритми застосування санкціонованого (легального) програмного забезпечення і програмні засоби несанкціонованого доступу для здійснення незаконного доступу до інформаційних ресурсів.

До **інформаційної програмної зброї** належать програми з потенційно небезпечними наслідками своєї роботи для інформаційних ресурсів мережі обміну інформацією.

Програми з потенційно небезпечними наслідками умовно поділяють на такі **класи**: комп'ютерні віруси; засоби несанкціонованого доступу; програмні закладки.

**Комп'ютерні віруси** (від лат. *virus* – отрута) – це спеціальні програми, які здатні самочинно розмножуватися, створюючи свої копії, і поширюватися, модифікуючи (заражаючи) інші програми шляхом приєднання до них для подальшого одержання управління та відтворення нових копій.

Для комп'ютерних вірусів принципове значення мають такі класифікаційні ознаки: об'єкт впливу (зараження); спосіб зараження об'єкта; принцип маскування; деструктивні можливості.

За видом об’єкта зараження комп’ютерні віруси поділяються на завантажувальні віруси, файлові віруси, завантажувально-файлові віруси, макровіруси. За способом зараження комп’ютерні віруси поділяються на резидентні й нерезидентні. За способом маскування – на поліморфні віруси, віруси-невидимки (стелс-віруси) та комбіновані віруси. За деструктивними можливостями – на безпечні віруси й віруси, що виконують деструктивні функції. Комп’ютерні віруси можуть розмножуватися, упроваджуватися в програми, передаватися лініями зв’язку, мережами обміну інформацією, виводити з ладу системи керування та ін.

**Засоби несанкціонованого доступу** – клас програм з потенційно небезпечними наслідками, для яких обов’язковим є виконання таких функцій:

- руйнування (спотворення довільним чином) кодів програм в оперативній пам’яті;
- збереження фрагментів інформації з оперативної пам’яті в деякій ділянці зовнішньої пам’яті прямого доступу (локальної або віддаленої);
- спотворення довільним чином, блокування і (або) підміни масивів інформації, що виводиться у зовнішню пам’ять або в канал зв’язку, утворених у результаті роботи прикладних програм або масивів даних, що уже знаходяться у зовнішній пам’яті;
- нейтралізація роботи тестових програм і систем захисту інформаційних ресурсів.

**Програмні закладки** належать до програм з потенційно небезпечними наслідками:

- руйнування (спотворення довільним чином) кодів програм в оперативній пам’яті;
- збереження фрагментів інформації з оперативної пам’яті в ділянці зовнішньої пам’яті прямого доступу (локальної або віддаленої);
- спотворення довільним чином, блокування і (або) підміни масивів інформації, що виводиться у зовнішню пам’ять або в канал зв’язку, утворених у результаті роботи прикладних програм або масивів даних, що уже знаходяться у зовнішній пам’яті. Характерною ознакою (відносно засобів несанкціонованого доступу) є відсутність функцій подолання захисту.

Виділяють декілька видів програмних закладок: троянські програми; логічні бомби; програмні пастки; програмні хробаки.

До **троянських програм** належать програмні закладки, які мають законний доступ до системи, проте виконують також і приховані функції.

Так, троянські програми в доповнення до основних (проектних і документованих) надають додаткові, але не описані в документації функціональні можливості, спрямовані на те, щоб обійти контроль доступу й призвести до несанкціонованого знищення, блокування, модифікації або копіювання інформації, порушення роботи комп’ютерної мережі.

Ці можливості можуть самоліквідуватись, що робить неможливим їхнє виявлення, або ж можуть реалізуватися постійно, але існувати потай. Найбільш небезпечним є опосередкований вплив, при якому «Троянський кінь» діє в межах повноважень одного користувача, але в інтересах іншого користувача, установити особу якого інколи неможливо. Упровадження троянських програм в автоматизовані системи керування інформаційними системами та мережами здійснюється засобами: використанням віддалених атак; упровадженням програмних закладок в операційні системи та програмне забезпечення, що постачаються на експорт; агентурним шляхом.

Програмні закладки можна також класифікувати відповідно до мети створення та за способом доставки в систему. За метою створення: програмні закладки класу «дослідник»; програмні закладки класу «перехоплювач»; програмні закладки класу «руйнівник»; програмні класу «активна завада». За способом доставки в систему: закладки, асоційовані з програмно-апаратним середовищем (BIOS).

У цьому випадку під асоціюванням (від лат. *associatio* – сполучення, з'єднання, від *associo* – з'єдную) розуміють інтеграцію коду програми з потенційно небезпечними наслідками або її частини в код іншої програми таким чином, щоб при деяких умовах керування передавалося на код програми з потенційно небезпечними наслідками;

- закладки, асоційовані з програмами первинного завантаження (знаходяться в Master Boot або Record BOOT – секторах активних розділів);
- закладки, асоційовані із завантаженням драйверів, командного інтерпретатора, мережних драйверів (завантаженням операційної системи);
- закладки, асоційовані з прикладним програмним забезпеченням загального призначення (вбудовані в клавіатурні й екранні драйвери, програми тестування ПЕОМ, утиліти й оболонки типу NORTON);
- модулі, що виконуються, які містять тільки код закладки (як правило, упроваджуються в пакетні файли типу BAT);
- модулі-імітатори, що співпадають з деякими програмами, що потребують введення конфіденційної інформації, за зовнішнім виглядом;
- закладки, що маскуються під програмні засоби оптимізаційного призначення (архіватори, прискорювачі та ін.);
- закладки, що маскуються під програмні засоби ігрового й розважального призначення (як правило, використовуються для первинного впровадження закладок типу «дослідник»).

**Інформаційна боротьба** – це боротьба з використанням спеціальних способів і засобів для впливу на інформаційну сферу (середовище) конфронтуючої сторони, а також для захисту власної інформаційної сфери в інтересах досягнення поставленої мети. Інформаційна боротьба може бути як самостійним видом, так і складовою частиною будь-якого іншого різно-

виду боротьби (збройної, ідеологічної, економічної тощо). Вона ведеться постійно як у мирний, так і у воєнний час. Масштаби інформаційної боротьби настільки великі, що її підготовка і ведення мають містити плановий, систематичний характер, заснований на глибоких знаннях законів і закономірностей інформаційної боротьби.

**Теорія інформаційної боротьби** – система знань про характер, закони, закономірності; ця база є головною під час розробки фундаментальних положень теорії інформаційної боротьби та визначення напрямків її розвитку. Оскільки основними завданнями інформаційної боротьби є ураження об'єктів інформаційного середовища противника та захист власної інформації, то структура теорії інформаційної боротьби має охоплювати загальні основи теорії інформаційної боротьби, теорію ураження інформації й теорію захисту інформації.

**Теорія захисту інформації** – складова частина теорії інформаційної боротьби охоплює загальні положення, що визначають: предмет, завдання і зміст теорії; об'єкти й елементи захисту інформації; основні фактори, що впливають на зміст й ефективність захисту інформації, а також визначає та вивчає загрози інформації й методологічні основи її захисту, систему показників оцінки ефективності захисту інформації, загальну математичну модель захисту інформації, організаційно-технічні й правові основи захисту інформації.

### 1.4 Політика безпеки та її базові принципи

Фундаментальним поняттям інформаційної безпеки є **політика безпеки** (ПБ) або політика захисту. Важливість цього поняття важко переоцінити – існують ситуації, коли правильно сформульована політика є чи не єдиним механізмом захисту від несанкціонованого доступу (НСД).

З ПБ пов'язується поняття оптимальності рішень з організації та підтримки системи захисту. Іноді вдається досягти загальноприйнятого розуміння оптимальності прийнятого рішення і навіть довести його існування. Однак коли розв'язок багатоальтернативний, то загальноприйнятого розуміння оптимальності немає, а в тих випадках, коли розглядається питання про оптимальний у певному сенсі розв'язок, то його існування, зазвичай, можна довести лише в окремих задачах. Подібна ситуація існує і в задачах захисту інформації, оскільки неоднозначним є рішення про те, що система захищена. Крім того, система захисту – не самоціль, а має лише підпорядковане значення з підпорядкованою функцією, порівняно з головною метою обчислювального процесу.

Під поняттям **політики безпеки інформації** розуміється організована сукупність документованих керівних рішень, спрямованих на захист ін-формації й асоційованих із нею ресурсів. ПБ відображає систему поглядів, основних принципів, практичних рекомендацій і вимог, що закладаються в основу реалізованої в системі комплексної системи захисту

інформації. Формування ПБ є складним аналітичним важко формалізованим процесом. Існують різні типи конкретних політик, причому деякі з них припускають достатньо високий рівень формалізації. Окрім того, існують точні доказові методи оцінки ПБ. Дотримання ПБ має забезпечувати виконання такого компромісу між альтернативами, який обрали власники цінної інформації для її захисту. Вочевидь, що будучи результатом компромісу, ПБ ніколи не задовольнить усі сторони, що беруть участь у взаємодії з інформацією, яка захищається. Водночас, вибір ПБ – це кінцеве вирішення проблеми під час роботи з цінною інформацією. Після прийняття такого рішення можна будувати захист, тобто систему підтримки виконання правил ПБ. Таким чином, побудована система захисту інформації вдала, якщо вона надійно підтримує виконання правил ПБ. І навпаки, система захисту інформації – невдала, якщо вона ненадійно підтримує ПБ. Таке вирішення проблеми захищеності інформації та проблеми побудови системи захисту дозволяє залучити точні математичні методи. Тобто довести, що система в заданих умовах підтримує ПБ. У цьому і є суть доказового підходу до захисту інформації, який дозволяє говорити про «гарантовано захищену систему». Сенс «гарантованого захисту» у тому, що при дотриманні початкових умов завідомо виконуються усі правила ПБ. Термін **«гарантований захист»** уперше зустрічається в стандарті міністерства оборони США на вимоги до захищених систем («Оранжева книга»).

**Побудова політики безпеки**, звичайно, відповідає таким крокам:

1)В інформацію вноситься структура цінностей і проводиться аналіз ризику.

2)Визначаються правила для будь-якого процесу користування цим видом доступу до елементів інформації, що має таку оцінку цінностей.

Однак реалізація цих кроків є складною задачею. Результатом помилкового чи бездумного визначення правил ПБ, зазвичай, є руйнування цінності інформації без порушення політики. Таким чином, навіть вдала система захисту може бути «прозорою» для зловмисника за умов невдалої ПБ.

Під ПБ інформації варто розуміти набір законів, правил, рекомендацій тощо, які регламентують порядок обробки інформації та спрямовані на її захист від певних загроз. Що дрібніший об’єкт, відносно якого застосовується цей термін, то конкретнішими і формальнішими стають правила. ПБ інформації в автоматизованій системі є частиною загальної політики безпеки організації і може успадковувати, зокрема, положення державної політики у галузі захисту інформації. Для кожної автоматизованої системи ПБ інформації може бути індивідуальною і може залежати від технології обробки інформації, що реалізується, особливостей операційної системи (ОС), фізичного середовища і від багатьох інших чинників. Крім того, одна й та ж автоматизована система може реалізовувати декілька різноманітних технологій обробки інформації. Тоді і ПБ інформації в такій системі буде

складеною і її частини, що відповідають різним технологіям, можуть істотно відрізнятись. ПБ інформації, що реалізуються різними комп'ютерними системами будуть відрізнятися не тільки тим, що реалізовані в них функції захисту можуть забезпечувати захист від різних типів загроз, а й у зв'язку з тим, що ресурси комп'ютерної системи можуть істотно відрізнятись. Так, якщо операційна система оперує файлами, то СУБД має справу із записами, розподіленими в різних файлах.

Для визначення і формалізації процесу розробки ПБ у деякій організації необхідно розробляти два комплекти документів: узагальнена політика **(program-level)**; проблемно-орієнтована (окрема) політика **(issue-specific)**.

**Основною функцією узагальненої ПБ** є визначення програми захисту інформації, призначення відповідальних за її виконання осіб, формулювання цілей і об'єктів захисту, а також вироблення схеми для змушування дотримання розроблених правил і вказівок. Компонентами узагальненої ПБ вважаються призначення, сфера розповсюдження, визначення цілей захисту інформації, розподіл відповідальності за виконання й методи змушування дотримання правил.

**Проблемно-орієнтована ПБ** необхідна для виділення певних проблемних частин мережі і визначення позицій організації щодо них.

Якщо узагальнена ПБ описує глобальні аспекти захисту інформації і її схему, то окремі ПБ розроблюються для деяких видів діяльності й у деяких випадках для конкретних систем (наприклад, для захисту електронної кореспонденції). Таким чином, окремі ПБ стандартизують роботу і зменшують потенційний ризик, що виникає при некоректному використанні інформаційних ресурсів. Проблемно-орієнтована ПБ частково визначає керівні принципи під час створення функціональних інструкцій для співробітників організації. Формуючи окрему ПБ, виділяють такі її компоненти: формулювання проблеми, визначення позиції організації, визначення сфери розповсюдження, ролей і відповідальності, а також призначення осіб для контактів щодо цього питання.

Частина ПБ, яка регламентує правила доступу користувачів і процесів до ресурсів комп'ютерної системи, містить правила розмежування доступу (ПРД) **(access mediation rules)**.

**Види політик безпеки.** Як було сказано раніше, для дослідження ПБ почали широко використовуватися математичні методи, що дозволило розробити математичні моделі та чітко класифікувати найбільш відомі та розповсюджені ПБ. Саме завдяки назвам математичних моделей ПБ і виникла певна їхня класифікація. Крім того, основним змістом таких моделей є опис та дослідження правил розмежування доступу суб'єктів до об'єктів системи. На сьогодні відомі три типи моделей ПБ, які досить детально досліджені та широко використовуються. Це – дискреційна, мандатна та рольова ПБ. Перші дві досить давно відомі й досліджені, а рольова політика є недавнім досягненням теорії та практики захисту інформації.

Основою **дискреційної політики безпеки** (ДПБ) є дискреційне управління доступом (Discretionary Access Control – DAC), що визначається двома властивостями: усі суб'єкти і об'єкти мають бути однозначно ідентифіковані; права доступу суб'єкта до об'єкта системи визначаються на основі деякого зовнішнього відносно системи правила, яка базується на використанні атрибутів доступу.

Назва пункту є дослівним перекладом Discretionary policy, ще одним варіантом перекладу є **розмежувальна політика**. Ця політика – одна з найрозповсюдженіших у світі, у системах за замовчуванням мається на увазі саме ця політика. ДПБ реалізується за допомогою матриці доступу (access matrix), яка фіксує множину кожного суб'єкта до доступних йому об'єктів та суб'єктів. Існує декілька варіантів задання матриці доступу.

1) Листи можливостей (privilege list, profile): для кожного суб'єкта створюється список (файл) усіх об'єктів, до якого він має доступ.

2) Листи контролю доступу (access control list): для кожного об'єкта створюється список усіх суб'єктів, що мають доступи до нього.

До переваг ДПБ можна віднести досить просту реалізацію відповідних механізмів захисту. Саме цим зумовлений той факт, що більшість розповсюджених сьогодні захищених АС забезпечують виконання положень ДПБ. Крім того, при її реалізації досягається велика економія пам'яті, оскільки матриця доступів, звичайно, буває дуже розрядженою.

Однак багатьох проблем захисту ця політика вирішити не може. Наведемо найбільш суттєві вади ДПБ.

1) Одним із суттєвих недоліків цього класу політик є те, що вони не витримують атак за допомогою «Троянського коня». Це, зокрема, означає, що системи захисту інформаціїї, яка реалізує ДПБ, погано захищає від проникнення вірусів у систему й інших засобів прихованої руйнівної дії.

2) Ще однією проблемою ДПБ є автоматичне визначення прав. Так як об'єктів багато і їхня кількість безперервно змінюється, то задати заздалегідь вручну перелік прав кожного суб'єкта на доступ до об'єктів неможливо. Тому матриця доступу різними способами агрегується, наприклад, як суб'єкти залишаються тільки користувачі, а у відповідну клітинку матриці записують формули функцій, обчислення яких визначає права доступу суб'єкта, породженого користувачем, до об'єкта. Звичайно, ці функції можуть змінюватися за часом. Зокрема, можливе вилучення прав після виконання події. Можливі модифікації, які залежать від інших параметрів.

3) Ще однією з найважливіших проблем під час використання ДПБ є проблема контролю розповсюдження прав доступу. Найчастіше буває, що власник файла передає вміст файла іншому користувачу і той, таким чином, набуває права власника на цю інформацію. Отже, права можуть розповсюджуватися, навіть якщо перший власник не бажав передавати доступ іншому суб'єкту до своєї інформації, то після декількох кроків передача прав може відбуватися незалежно від його волі. Виникає задача про умови,

за якими в такій системі деякий суб'єкт рано чи пізно отримає необхідний йому доступ.

4) При використанні ДПБ виникає питання визначення правил розповсюдження прав доступу і аналізу їхнього впливу на безпеку системи. Загалом при використанні ДПБ органом, який її реалізує і який при санкціонуванні доступу суб'єкта до об'єкта керується деяким набором правил, постає задача, яку алгоритмічно неможливо розв'язати: перевірити, призведуть його дії до порушень безпеки чи ні.

Отже, матриця доступів не є тим механізмом, який дозволив би реалізувати ясну й чітку систему захисту інформації. Більш досконалою ПБ виявилася мандатна ПБ.

Основу **мандатної (повноважної) політики безпеки** (МПБ) становить мандатне управління доступом (Mandatory Access Control – MAC), яке має на увазі, що:

- усі суб'єкти й об'єкти мають бути однозначно ідентифіковані;
- заданий лінійно упорядкований набір міток секретності;
- кожному об'єктові системи привласнена мітка секретності, яка визначає цінність інформації, що міститься в ньому, – його рівень секретності в системі;
- кожному суб'єктові системи привласнена мітка секретності, яка визначає рівень довіри до нього в системі,– максимальне значення мітки секретності об'єктів, до яких суб'єкт має доступ; мітка секретності суб'єкта називається його рівнем доступу.

Основна мета МПБ – запобігання витоку інформації від об'єктів з високим рівнем доступу до об'єктів з низькими рівнем доступу, тобто протидія виникненню в мережі інформаційних каналів зверху вниз. Вона оперує, таким чином, поняттями інформаційного потоку й цінності (певним значенням мітки секретності) інформаційних об'єктів.

Цінність інформаційних об'єктів, зазвичай, досить важко визначити. Однак досвід показує, що в будь-якій системі майже завжди для будь-якої пари об'єктів X та Y можна сказати, який з них більш цінний. Тобто можна вважати, що таким чином фактично визначається деяка однозначна функція c(X), яка дозволяє для будь-яких об'єктів X і Y зазначити, що коли Y більш цінний об'єкт, ніж X, то $c(Y)>c(X)$. І навпаки, у силу однозначності, якщо $c(Y)>c(X)$, то Y – більш цінний об'єкт, ніж X. Тоді потік інформації від X до Y дозволяється, якщо $c(X)<c(Y)$, і не дозволяється, якщо $c(X)>c(Y)$.

Таким чином, МПБ має справу з множиною інформаційних потоків, яка ділиться на дозволені й недозволені дуже простою умовою – значенням наведеної функції. Іншими словами, управління потоками інформації здійснюється через контроль доступів. МПБ у сучасних системах захисту на практиці реалізується мандатним контролем. Він реалізується на найнижчому апаратно-програмному рівні, що дозволяє досить ефективно будувати захищене середовище для механізму мандатного контролю.

Пристрій мандатного контролю називають монітором звернень. Мандатний контроль ще називають обов'язковим, так як його має проходити кожне звернення суб'єкта до об'єкта, якщо вони знаходяться під захистом СЗІ. Організовується він так: кожний об'єкт має мітку з інформацією про свій рівень секретності; кожний суб'єкт також має мітку з інформацією про те, до яких об'єктів він має право доступу. Мандатний контроль порівнює мітки і приймає рішення про допуск або ні.

Найчастіше МПБ описують у термінах, поняттях і визначеннях властивостей моделі Белла-Лападула. У межах цієї моделі доводиться важливе твердження, що вказує на принципову відмінність систем, що реалізують мандатний захист, від систем з дискреційним захистом: **якщо початковий стан системи безпечний і всі переходи системи зі стану в стан не порушують обмежень, сформульованих ПБ, то будь-який стан системи безпечний.**

Наведемо низку переваг МПБ, порівняно з ДПБ.

1) Для систем, де реалізовано МПБ, є характерним вищий ступінь надійності. Це пов'язано з тим, що, за правилами МПБ, відстежуються не тільки правила доступу суб'єктів системи до об'єктів, а й стан самої системи. Отже, канали витоку в системах такого типу не закладені первісно (що є в положеннях ДПБ), а можуть виникнути тільки під час практичної реалізації систем унаслідок помилок розробника.

2) Правила МПБ більш ясні й прості для розуміння розробниками та користувачами системи, що також є фактором, який позитивно впливає на рівень безпеки.

3) МПБ стійка до атак типу «Троянський кінь».

4) МПБ допускає можливість точного математичного доказу, що така система в заданих умовах підтримує ПБ.

Розглянемо ще один вид ПБ – рольову ПБ. **Рольову політику безпеки**, РПБ, (Role Base Access Control – RBAC) не можна віднести ані до дискреційної, ані до мандатної, тому що керування доступом здійснюється як на основі матриці прав доступу для ролей, так і за допомогою правил, що регламентують призначення ролей користувачам та їхню активацію під час сеансів. Отже, рольова модель є цілком новим типом політики, яка базується на компромісі між гнучкістю керування доступом, що є характерною для ДПБ, і жорсткістю правил контролю доступу, яка притаманна МПБ.

У РПБ класичне поняття **суб'єкт** заміщується поняттями **користувач** і **роль**. Користувач – це людина, котра працює із системою і виконує певні службові обов'язки. Роль – це активно діюча в системі абстрактна суттєвість, з якою пов'язаний обмежений та логічно пов'язаний набір повноважень, що необхідні для здійснення певної діяльності.

РПБ розповсюджена досить широко, тому що вона, на відміну від інших більш суворих і формальних політик, є дуже близькою до реального життя. Дійсно, користувачі, що працюють у системі, діють не від свого

особистого імені – вони завжди здійснюють певні службові обов'язки, тобто виконують деякі ролі, що аж ніяк не пов'язані з їхньою особистістю.

Тому цілком логічно здійснювати керування доступом і призначати повноваження не реальним користувачам, а абстрактним (не персоніфікованим) ролям, які представляють учасників певного процесу обробки інформації. Такий підхід до ПБ дозволяє урахувати розділ обов'язків і повноважень між учасниками прикладного інформаційного процесу, оскільки з точки зору РПБ має значення не особистість користувача, котрий здійснює доступ до інформації, а те, які повноваження йому необхідні для виконання його службових обов'язків. Наприклад, у реальній системі обробки інформації можуть працювати системний адміністратор, менеджер баз даних і прості користувачі.

У такій ситуації РПБ дозволяє розподілити повноваження між цими ролями, відповідно до їхніх службових обов'язків: ролі адміністратора призначаються спеціальні повноваження, які дозволять йому контролювати роботу системи й керувати її конфігурацією; роль менеджера баз даних дозволяє здійснювати керування сервером БД, а права простих користувачів обмежуються мінімумом, необхідним для запуску прикладних програм. Крім того, кількість ролей у системі може не відповідати кількості реальних користувачів – один користувач, якщо він має різні повноваження, може виконувати (водночас або послідовно) декілька ролей, а декілька користувачів можуть користуватись однією і тією ж роллю, якщо вони виконують однакову роботу.

При використанні РПБ керування доступом здійснюється у дві стадії: по-перше, для кожної ролі вказується набір повноважень, що представляють набір прав доступу до об'єктів; по-друге, кожному користувачу призначається список доступних йому ролей. Повноваження призначаються ролям відповідно до принципу найменших привілеїв, з якого випливає, що кожний користувач повинен мати тільки мінімально необхідні для виконання своєї роботи повноваження.

У моделі РПБ визначаються такі множини: U – множина користувачів; R – множина ролей; P – множина повноважень на доступ до об'єктів, що представляється, наприклад, у вигляді матриці прав доступу; S – множина сеансів роботи користувачів із системою.

Для перелічених множин визначаються відношення: відображає множину повноважень на множину ролей, установлюючи для кожної ролі набір наданих їй повноважень; відображає множину користувачів на множину ролей, визначаючи для кожного користувача набір доступних ролей.

Правила керування доступом рольової політики безпеки визначаються такими функціями:

$$User: S \rightarrow U, user\ (s) = u \qquad (1.1)$$

Тобто для кожного сеансу *s* функція (1.1) визначає користувача *u*, який здійснює цей сеанс роботи із системою: *user(s)=u*.

$$Roles: S \rightarrow R, roles\ (s) = \{r|(user(s), r)\} \quad (1.2)$$

Для кожного сеансу *s* функція (1.2) визначає набір ролей з множини *R*, що можуть бути одночасно доступні користувачу *u* у цьому сеансі.

$$Permitions: S \rightarrow P, permitions(s) = \{p|(p, r)\} \quad (1.3)$$

Для кожного сеансу *s* функція (1.3) задає набір доступних у ньому повноважень, що визначається як сукупність повноважень усіх ролей, які беруть участь у цьому сеансі. Як критерій безпеки рольової моделі використовується правило: **система вважається безпечною, якщо будь-який користувач системи, котрий працює в сеансі *s*, може здійснити дії, які вимагають повноважень *p* тільки тоді, коли *p permissions(s)*.** З формулювання критерію безпеки моделі РПБ виникає, що управління доступом здійснюється, переважно, не за допомогою призначення повноважень ролям, а шляхом задання відношення, яке призначає ролі користувачам, і функції *roles*, що визначає доступний у сеансі набір ролей. Тому численні інтерпретації рольової моделі відрізняються видом функцій *user*, *roles* і *permission*.

Ієрархічна організація ролей є найбільш розповсюдженим типом рольової моделі, оскільки вона досить точно відображає реальне відношення підпорядкованості між учасниками процесів обробки інформації і розподілом між ними сфер відповідальності. Ролі в ієрархії упорядковуються за рівнем наданих повноважень. Що вища роль в ієрархії, то більше з нею пов’язано повноважень, оскільки вважається: коли користувачу надано деяку роль, то йому автоматично призначаються і всі підпорядковані їй за ієрархією ролі. Ієрархічна організація ролей є характерною для систем військового призначення.

# 2 КЛАСИФІКАЦІЯ ТА РІЗНОВИДИ АТАК

Нині зловмисники використовують велике різноманіття атак, які можна систематизувати за різними параметрами.

За **характером впливу на мережу** атаки поділяють на **пасивні** та **активні**. Пасивні атаки не порушують роботу інформаційної системи, але порушують політику безпеки. До пасивних атак можна віднести прослуховування каналів даних, перехоплення трафіка і т. д. Пасивні атаки досить часто передують активним. Для зловмисника корисною інформацією, на отримання якої і спрямовано пасивну атаку, є типи операційних систем і програм, що працюють у мережі, IP-адреси, номери портів клієнтських частин програм, імена і паролі користувачів. Активні атаки безпосередньо впливають на роботу мережі або окремих її компонентів за рахунок модифікування конфігураційних файлів, установлення додаткових програмних компонентів тощо.

Залежно від **мети**, яку ставить зловмисник, атаку може бути спрямовано на **порушення конфіденційності**, **цілісності інформації** або **порушення працездатності системи загалом**. Порушення конфіденційності можливе у разі отримання зловмисником несанкціонованого доступу до інформації. Порушення цілісності передбачає спотворення інформації, а порушення працездатності системи – створення умов, за яких система не може виконувати свої функції.

За **місцем розташування** зловмисника атаки поділяються на **внутрішні** та **зовнішні**. Внутрішня атака – це реалізована загроза з боку легальних користувачів мережі. Зовнішня атака – відповідно реалізована загроза зовнішніх зловмисників.

За **рівнем кваліфікації** зловмисника атаки поділяються на **структуровані** і **неструктуровані**. Структуровані атаки, як правило, проводяться професійними хакерами і складаються з кількох етапів. На першому етапі різноманітними засобами збирається інформація про об’єкт атаки з метою отримання хоча б якогось доступу до системи. На другому етапі зловмисник збільшує свій рівень привілеїв доти, доки їх не буде достатньо для реалізації спланованих дій. Після цього атака переходить в основну фазу. Неструктуровані атаки, як правило, здійснюються хакерами-початківцями з метою набуття досвіду або перевірки тієї чи іншої вразливості системи.

За **кількістю атакувальників** атаки поділяються на **одноосібні** та **розподілені**. У першому випадку атака здійснюється з одного комп’ютера, розташованого у внутрішній чи зовнішній мережі. Метою одноосібних атак, зазвичай, є порушення конфіденційності або цілісності інформації. Розподілена атака передбачає участь кількох комп’ютерів, на які попередньо встановлюється необхідне програмне забезпечення. Досить часто зловмисник «готує» учасників розподіленої атаки, взламуючи систему захисту персональних комп’ютерів, під’єднаних до мережі, та встановлюючи на них шкідливе програмне забезпечення. Розподілені атаки, як правило, спрямовані на порушення функціонування системи-жертви.

За **рівнем еталонної моделі OSI** дії зловмисника поділяються на атаки **фізичного, канального, мережевого, транспортного, сеансового, представницького рівня** та **рівня додатків**. Атака того чи іншого рівня використовує вразливості протоколів, що працюють на відповідному рівні моделі OSI.

Сьогодні в арсеналі зловмисників велике різноманіття програмних та апаратних засобів, які дають змогу реалізовувати найрізноманітніші атаки, найбільш поширеними серед яких є: атаки розвідницького типу (Reconnaissance attacs); переспрямування трафіка; атаки, спрямовані на отримання доступу до системи (Access attacks); атаки відмови в обслуговуванні (Denial of Service attacks); занесення шкідливого програмного забезпечення (malware).

## 2.1 Атаки розвідницького типу

Атаки розвідницького типу спрямовані на збирання інформації про систему (mapping). Існують різноманітні методи та засоби збирання інформації, найбільш відомими з яких є: засоби для перехоплення й аналізу трафіка (Packet sniffers); засоби для знаходження IP-адрес працюючих комп'ютерів (Ping sweeps); засоби для сканування відкритих TCP та UDP портів (Port scans); методи соціальної інженерії.

**Засоби перехоплення й аналізу трафіка** базуються на використанні мережевого адаптера, що працює в невпорядкованому режимі, та програмному забезпеченні, що допомагає розпізнати вміст захоплених пакетів. У такому разі мережевий адаптер перехоплює увесь трафік, який на нього потрапляє. Одним з популярних аналізаторів трафіка є безкоштовна програма Wireshark. Використання комутаторів замість концентраторів у локальних мережах суттєво обмежило обсяг трафіка, що потрапляє на окрему робочу станцію, тому зловмисники використовують додаткові засоби, метою яких є спрямовування трафіка на комп'ютер зловмисника. Методи переспрямування трафіка будуть розглянуті нижче.

Для **знаходження IP-адреси працюючих комп'ютерів** використовується діагностичний протокол ICMP. Зловмисник відправляє echo-запити на IP-адреси певного діапазону. Комп'ютери, що надіслали echo-відповіді, можуть стати об'єктами подальших дій зловмисника.

**Сканування відкритих портів** – це подальший етап після визначення IP-адреси жертви. Зловмисник використовує особливості реалізації транспортного рівня протокольного стеку TCP/IP, зокрема механізм обміну даних через, так звані, протокольні порти. Кожний TCP або UDP порт асоціюється з певним мережевим додатком, запущеним на комп'ютері, до того ж номер порту ідентифікує тип мережевого сервісу. Маючи інформацію про вразливості певних мережевих сервісів та їхній перелік на комп'ютері-жертві, зловмисник може переходити до подальшої фази атаки.

**Методи соціальної інженерії** – це спосіб отримати необхідну інформацію безпосередньо у користувачів комп'ютерної мережі. Для цього зловмисник, телефонуючи користувачу, представляється адміністратором або іншим технічним спеціалістом і просить для «здійснення чергового обліку», «оновлення інформації про користувачів та комп'ютери» тощо назвати певну конфіденційну інформацію, відому користувачеві.

## 2.2 Переспрямування трафіку

Однією з актуальних задач зловмисника є спрямовування трафіка, що його цікавить, на іншу (хибну) адресу, якою може бути або адреса зловмисника, або адреса третьої сторони. Зловмисник організовує транзит трафіка через власний комп'ютер. Кожний перехоплений пакет аналізується на атакуючому вузлі, а після цього переспрямовується на «справжній» сервер. Таким чином, увесь трафік між клієнтом і сервером проходить через комп'ютер зловмисника.

Найпростіший варіант переспрямування трафіка в локальній мережі може бути реалізований шляхом відправлення в мережу **хибної ARP-відповіді**. У такому разі схема є очевидною: отримавши широкомовний ARP-запит щодо деякої IP-адреси, зловмисник відправляє хибну ARP-відповідь, у якій повідомляється, що цій IP-адресі відповідає його власна MAC-адреса.

Для перехоплення й переспрямування трафіка в локальній мережі теоретично також може використовуватись **протокол ICMP**. ICMP-повідомлення про зміну маршруту маршрутизатор за замовчуванням відправляє на хост безпосередньо приєднаної локальної мережі, якщо цей маршрут відмовив або тоді, коли виявляє, що для певної адреси призначення хост використовує нераціональний маршрут.

Ще один спосіб перехоплення трафіка базується на використанні **хибних DNS-відповідей**. Зловмисник знає, що клієнт звертається до сервера, вказуючи його символьне DNS-ім'я www.server.ua. Також йому відомо, що перед тим, як надіслати пакет серверу, програмне забезпечення клієнтської машини спрямовує запит DNS-серверу, щоб дізнатися, яка IP-адреса відповідає цьому імені. Мета зловмисника – випередити відповідь DNS-сервера і нав'язати клієнту свій варіант відповіді, у якому замість IP-адреси корпоративного сервера зловмисник вказує IP-адресу власного хоста.

## 2.3 Атаки, спрямовані на отримання доступу до системи

Найбільш поширеними атаками, спрямованими на отримання доступу до системи є: атаки на пароль; атаки використання довіри; людина посередині; переповнення буферу.

**Атаки на пароль** спрямовані на підбирання паролю. Ефективність таких атак пояснюється тим фактом, що користувачі досить часто вико-

ристовують занадто прості паролі, які містять ім'я або прізвище користувача, імена дітей або інших родичів, імена домашніх улюбленців тощо. Інший варіант простих паролів – це використання звичайних слів (на такі паролі спрямовуються словникові атаки).

**Атаки використання довіри** (Trust exploitation) базуються на такій схемі: А довіряє В, а В довіряє С, водночас, між С та А безпосередньо немає довірливих відносин. Така схема є дуже поширеною при реалізації захищеного периметру з використанням, так званої, демілітаризаційної зони (DMZ). Атака складається з двох етапів: С, скориставшись довірою, отримує привілейований доступ до системи В, а вже через систему В отримує доступ до системи А. Різновидом атаки використання довіри є атака переспрямування портів (Port redirection).

**Атака «Людина посередині»** (Man-in-the-middle) полягає в тому, що зловмисник маскується під сервер адресата, передаючи клієнту ті «картинку» і повідомлення, які той очікує. Таким чином, зловмисник може імітувати для користувача-жертви процедуру логічного входу, отримуючи при цьому його ідентифікатор і пароль. У подальшому ці дані можуть використовуватись для несанкціонованого доступу до сервера підприємства чи банку, які і є основною ціллю атаки.

**Атака переповнення буферу** базується на тому факті, що сучасне програмне забезпечення використовує спільний адресний простір в оперативній пам'яті для зберігання програмного коду й інформації, що потрапляє у вхідний буфер з мережі. Зловмисник свідомо спрямовує у вхідний буфер більше інформації, ніж той може вмістити, унаслідок чого частина даних, що не уміщується в буфер, потрапляє в зону пам'яті, призначену для зберігання програмного коду. Таким чином, зловмисник фактично модифікує код програми, що знаходиться в оперативній пам'яті і відкриває доступ до системи.

## 2.4 Атаки відмови в обслуговуванні

Атаки відмови в обслуговуванні (Denial of Service, DoS), зазвичай, спрямовуються на інформаційні сервери підприємства, функціонування яких є вкрай важливою умовою для працездатності всього підприємства. Найчастіше об'єктами DoS-атак стають основні веб-сервери, файлові й поштові сервери підприємства, а також кореневі сервери системи DNS.

Для проведення DoS-атак зловмисники часто координують «роботу» кількох комп'ютерів (як правило, без відома користувачів даних комп'ютерів). Це, так звана, розподілена атака відмови в обслуговуванні (Distributed Denial of Service, DDoS). Захопивши управління групою віддалених комп'ютерів, зловмисник «змушує» їх відсилати пакети на адресу вузла-жертви. Загальний потік пакетів «затоплює» атакований комп'ютер. Це викликає його перевантаження і, як наслідок, робить його недоступним. Блокування відбувається в результаті вичерпання ресурсів процесора, оперативної пам'яті або каналу зв'язку (смуги пропускання).

Приклад проведення DoS-атаки, у якій використовуються особливості протоколу TCP, показано на рис. 2.1. Для встановлення логічного з'єднання за протоколом TCP вузли мають обмінятися трьома пакетами: спочатку ініціатор з'єднання надсилає SYN-пакет, на який сервер відповідає пакетами ACK і SYN. Завершує процедуру ACK-пакет від вузла-ініціатора.

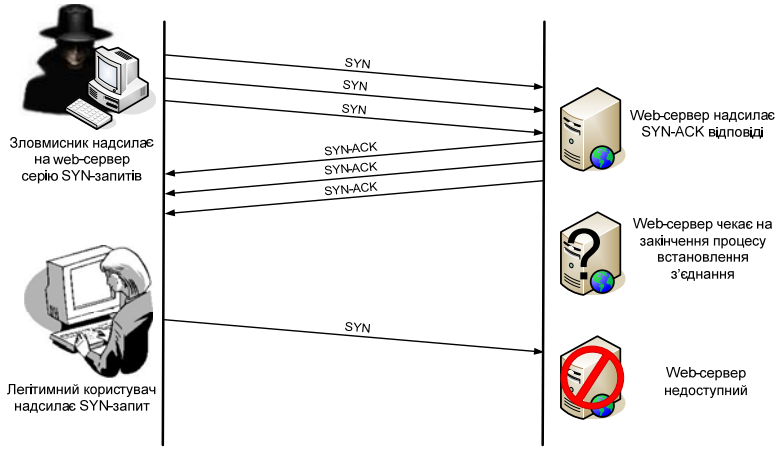


Рисунок 2.1 – Приклад організації DoS-атаки

Для здійснення атаки зловмисник організовує передавання на сервер масового потоку SYN-пакетів, кожний з яких ініціює створення нового TCP-з'єднання. Отримавши SYN-пакет, сервер виділяє для нового з'єднання необхідні ресурси і, відповідно до протоколу, відповідає клієнту пакетами ACK і SYN. Після цього, установивши тайм-аут, він очікує від клієнта завершальний ACK-пакет, який так і не надходить. Аналогічним чином створюється безліч інших «недовстановлених» з'єднань. Унаслідок цього виникає перевантаження сервера, оскільки всі його ресурси використовуються для підтримки тих з'єднань, процедури встановлення яких залишились незавершеними. У такому стані сервер не спроможний відповідати на запити, що надходять від легітимних користувачів. Отже, зловмисник досягає своєї мети.

## 2.5 Занесення шкідливого програмного забезпечення

Значна група атак пов'язана із занесенням в комп'ютери шкідливих програм (malware), до яких належать: троянські та шпіонські програми, черв'яки, віруси, спам, логічні бомби та деякі інші види програм, спрямовані на порушення інформаційної безпеки.

Ці програми можуть проникати на атаковані комп'ютери різними шляхами. Найчастіше це відбувається, коли користувач завантажує файли із неперевірених джерел (змінних носіїв чи веб-сайтів) або безпечно відкриває підозрілий файл, який надходить йому на електронну пошту. Існують і більш небезпечні представники шкідливих програм, що мають власні

механізми «розмноження», копії таких програм розповсюджуються на комп'ютери в мережі без участі користувачів.

**Троянські програми**, або трояни (trojan), – це різновид шкідливих програм, які наносять шкоду системі, маскуючись під корисні програми.

Троянські програми можуть використовувати як прикриття знайомі для користувача програми, з якими він працював і раніше. В іншому випадку, відповідно до стародавньої легенди, троянська програма набуває вигляду нової програми, яка намагається зацікавити користувача-жертву якимись своїми псевдокорисними функціями.

**Мережеві черв'яки** (worms) – це програми, що здатні самостійно розповсюджувати свої копії як в межах локальної мережі, так і через глобальні зв'язки, пересуваючись від одного комп'ютера до іншого без будь-якої участі користувачів мережі.

Головна мета і результат діяльності черв'яка полягає у передачі своєї копії на максимально можливу кількість комп'ютерів. Для пошуку комп'ютерів – нових потенційних жертв – черв'яки запускають у дію вбудовані в них засоби. Стандартна програма-черв'як не видаляє і не спотворює файли користувача та системні файли, не перехоплює електронну пошту користувачів, не псує вміст баз даних, а завдає шкоду атакованим комп'ютерам шляхом споживання їхніх ресурсів. Черв'як складається з двох основних функціональних компонентів: атакуючого блоку й блоку пошуку цілей.

**Вірус** (virus) – це шкідливий програмний фрагмент, який може вбудовуватись в інші файли. Вірус може вбудовувати свої фрагменти в різні типи файлів, зокрема, у файли програм, що виконуються. При цьому можливі найрізноманітніші варіанти: заміщення коду, коли розмір інформаційного файлу не змінюється, вставка вірусного коду повністю на початок або в кінець вихідної програми, заміна фрагментів програмного коду фрагментами вірусу з перестановкою заміщених фрагментів і без перестановки тощо. Код вірусу може бути зашифрований з метою ускладнення його виявлення антивірусними програмами.

**Шпигунські програми** (spyware) – це такий тип шкідливих програм, які таємно (зазвичай, віддалено) установлюються зловмисниками на комп'ютери жертв, щоб відстежувати й фіксувати всі їхні дії. До переліку таких дій може входити введення імені й пароля під час логічного входу в систему, відвідування тих чи інших веб-сайтів, обмін інформацією із внутрішніми та зовнішніми користувачами мережі тощо. Зібрана інформація передається зловмиснику, котрий використовує її для злочинних дій.

**Спам** – це атака, що здійснюється шляхом зловживання можливостями електронної пошти. Спам забирає час і ресурси на перегляд і видалення повідомлень, при цьому помилково можуть бути видалені листи із надзвичайно важливою інформацією, особливо велика вірогідність цього виникає під час автоматичної фільтрації листів. Зайва пошта не тільки знижує ефективність роботи підприємства, а й досить часто є засобом надходження

шкідливих програм. Крім того, спам, зазвичай, є елементом різноманітних схем махінацій, жертвами яких можуть стати як окремі співробітники, так і все підприємство. Спамери, тобто особи, що розсилають спам, використовують для своїх цілей різноманітні, іноді досить складні, методи й засоби. Так, наприклад, для поповнення баз даних адрес вони можуть виконувати автоматичне сканування сторінок Інтернету, а для організації масового розсилання вони можуть використовувати розподілені атаки, коли вражені за допомогою черв'яків комп'ютери «бомбардують» величезну кількість користувачів мережі.

## 2.6 Соціальна інженерія

Соціальна інженерія – це метод несанкціонованого доступу до інформації або системи без використання технічних засобів. Метод заснований на використанні слабкостей людського фактора і є дуже ефективним. Зловмисник отримує інформацію, наприклад, шляхом збору інформації про службовців об'єкта атаки, за допомогою звичайного телефонного дзвінка або шляхом проникнення в організацію під виглядом її службовця. Зловмисник може подзвонити працівникові компанії (під виглядом технічної служби) і вивідати пароль, посилаючись на необхідність вирішення невеликої проблеми в комп'ютерній системі. Дуже часто подібна афера вдається. Найсильніша зброя в цьому випадку – приємний голос і акторські здібності зловмисника. Імена службовців вдається дізнатися після низки дзвінків і вивчення імен керівників на сайті компанії та інших джерел відкритої інформації (звітів, реклами тощо). Використовуючи реальні імена в розмові зі службою технічної підтримки, зловмисник розповідає вигадану історію, що не може потрапити на важливу нараду на сайті зі своїм обліковим записом віддаленого доступу. Допоміжним підходом в такому методі є дослідження сміттєвих контейнерів організацій, віртуальних сміттєвих кошиків, крадіжка портативного комп'ютера та інших носіїв інформації. Такий метод використовується, коли зловмисник вибрав як жертву конкретну компанію.

### 2.6.1 Техніки соціальної інженерії

**Фішинг** (англ. *phishing*, від *fishing* – риболовля, видобування) – це вид інтернет-шахрайства, метою якого є отримання доступу до конфіденційних даних користувачів – логінів і паролів. Мабуть, це найпопулярніша схема соціальної інженерії на сьогоднішній день. Жоден великий витік персональних даних не обходиться без хвилі фішингових розсилок, що передують їй. Найбільш яскравим прикладом фішингової атаки може бути повідомлення, відправлене жертві на електронну пошту, і підроблене під офіційний лист – від банку або платіжної системи – вимагає перевірки певної інформації або здійснення певних дій. Причини можуть називатися найрізноманітніші. Це може бути втрата даних, поломка в системі тощо. Такі

листи, зазвичай, містять посилання на фальшиву веб-сторінку, цілком схожу на офіційну, і містять форму, що вимагає ввести конфіденційну інформацію.

Найчастіше фішингові повідомлення містять:

- відомості, що викликають занепокоєння, або загрози, наприклад, закриття призначених для користувача банківських рахунків;
- обіцянки грошового призу з мінімальними зусиллями або зовсім без них;
- запити про добровільні пожертвування від імені благодійних організацій;
- граматичні, пунктуаційні та орфографічні помилки;
- імітацію пошкодженого або неправильно кодування;
- повідомлення, які надходять з неіснуючої поштової скриньки.

**Плечовий серфінг** (англ. *shoulder surfing*) охоплює спостереження особистої інформації про жертву через її плече. Цей тип атаки поширений у таких громадських місцях, як: кафе, торговельні центри, аеропорти, вокзали, а також у громадському транспорті.

Опитування ІТ-фахівців про безпеку показало:

- 85% опитаних зізналися, що бачили конфіденційну інформацію, яку їм не належало знати;
- 82% зізналися, що інформацію, яка відображається на їхньому екрані, могли б бачити сторонні особи;
- 82% не впевнені в тому, що в їхній організації будь-хто буде захищати свій екран від сторонніх осіб.

**Квід про кво** (від лат. *quid pro quo* – «то за це») – в англійській мові цей вислів, зазвичай, використовується в значенні «послуга за послугу». Цей вид атаки використовує звернення зловмисника в компанію за допомогою корпоративного телефону або електронної пошти. Найчастіше зловмисник представляється співробітником технічної підтримки, котрий повідомляє про виникнення технічних проблем на робочому місці співробітника і пропонує допомогу в їхньому усуненні. У процесі «вирішення» технічних проблем, зловмисник змушує ціль атаки здійснити дії, які дозволяють атакуючому запускати команди або встановлювати програмне забезпечення на комп’ютері «жертви».

**Збір інформації з відкритих джерел.** Застосування технік соціальної інженерії вимагає не тільки знання психології, а й уміння збирати про людину необхідну інформацію. Відносно новим способом отримання такої інформації став її збір з відкритих джерел, головним чином із соціальних мереж. Як правило, їх користувачі не приділяють належної уваги питанням безпеки, залишаючи у вільному доступі дані і відомості, які можуть бути використані зловмисником. Навіть обмеживши доступ до інформації на своїй сторінці в соціальній мережі, користувач не може бути точно впевнений, що вона ніколи не потрапить до рук шахраїв.

**«Дорожнє яблуко»** – цей метод атаки є адаптацію троянського коня, і полягає у використанні фізичних носіїв. Зловмисник підкидає «інфіковані» носії інформації в місцях загального доступу, де ці носії можуть бути легко знайдені, наприклад, місця паркування, столові, або робочі місця співробітників. Носії оформляються як офіційні для компанії, яку атакують, або супроводжуються підписом, покликаного викликати цікавість. Наприклад, зловмисник може підкинути CD, забезпечений корпоративним логотипом і посиланням на офіційний сайт компанії, забезпечивши його написом «Заробітна плата керівного складу». Диск може бути залишений на підлозі ліфта або у вестибюлі. Співробітник через незнання може підібрати диск і вставити його в комп'ютер, щоб задовольнити свою цікавість.

**Зворотна соціальна інженерія.** Про зворотну соціальну інженерію згадують тоді, коли жертва сама пропонує зловмиснику потрібну йому інформацію. Це може здатися абсурдним, але насправді особи, що мають авторитет у технічній або соціальній сфері, часто отримують ідентифікатори й паролі користувачів та іншу важливу особисту інформацію просто тому, що ніхто не сумнівається в їхній порядності. Наприклад, співробітники служби підтримки ніколи не запитують у користувачів ідентифікатор або пароль, їм не потрібна ця інформація для вирішення проблем. Однак, багато користувачів заради швидкого усунення проблем добровільно повідомляють ці конфіденційні відомості.

### 2.6.2 Способи захисту від соціальної інженерії

Основним способом захисту від соціальної інженерії є навчання. Оскільки той, хто попереджений – той озброєний. І незнання, передусім, не звільняє від відповідальності. Усі працівники компанії мають знати про небезпеку розкриття інформації та способи її запобігання. Крім того, співробітники компанії повинні мати чіткі інструкції про те, як та на які теми розмовляти зі співрозмовником, яку інформацію для точної аутентифікації співрозмовника їм необхідно у нього отримати. Ось деякі правила, що будуть корисні:

1) Усі призначені користувачу паролі є власністю компанії. Усім співробітникам має бути роз'яснено в день прийому на роботу, що ті паролі, які їм видали, не можна використовувати в будь-яких інших цілях, наприклад, для авторизації на інтернет-сайтах (відомо, що людині важко тримати в голові всі паролі та коди доступу, тому вона часто користується одним паролем для різних ситуацій).

2) Усі співробітники мають бути проінструктовані про правила поведінки із відвідувачами. Необхідні чіткі правила для встановлення особи відвідувача і його супроводу. З відвідувачем завжди повинен знаходитися співробітник компанії.

3) Має існувати правило коректного розкриття необхідної інформації телефоном і під час особистої розмови.

Усі описані заходи досить прості, однак більшість співробітників забувають про це і про той рівень відповідальності, що на них покладено під час підписання зобов'язань про нерозголошення комерційної таємниці.

### 2.6.3 Багаторівнева модель забезпечення безпеки

Для захисту великих компаній та їхніх співробітників від шахраїв, котрі використовують техніки соціальної інженерії, часто застосовуються комплексні багаторівневі системи безпеки. Нижче перераховані деякі особливості й обов'язки таких систем.

- *Фізична безпека.* Бар'єри, що обмежують доступ у будівлі компанії і до корпоративних ресурсів. Не варто забувати, що ресурси компанії, наприклад, сміттєві контейнери, розташовані поза територією компанії, фізично не захищені.
- *Дані.* Ділова інформація: облікові записи, поштова кореспонденція тощо. При аналізі загроз і плануванні заходів щодо захисту даних потрібно визначити принципи поводження з паперовими й електронними носіями даних.
- *Додатки.* Програми, що запускаються користувачами. Для захисту середовища необхідно врахувати, як зловмисники можуть використовувати у своїх цілях поштові програми, служби миттєвої передачі повідомлень та інші додатки.
- *Комп'ютери.* Сервери та клієнтські системи, які використовуються в організації. Захист користувачів від прямих атак на їхні комп'ютери шляхом визначення суворих принципів, які вказують, що програми можна використовувати на корпоративних комп'ютерах.
- *Внутрішня мережа.* Мережа, за допомогою якої взаємодіють корпоративні системи. Вона може бути локальною, глобальною або бездротовою. Протягом останніх років через зростання популярності методів віддаленої роботи, кордони внутрішніх мереж стали багато в чому умовними. Співробітникам компанії потрібно роз'яснити, що вони мають робити для організації безпечної роботи в будь-якому мережевому середовищі.
- *Периметр мережі.* Кордон між внутрішніми мережами компанії та зовнішніми: Інтернет або мережі партнерських організацій.

# 3 МЕТОДИ КРИПТОГРАФІЧНОГО ЗАХИСТУ ІНФОРМАЦІЇ

**Шифрування** – це засіб забезпечення конфіденційності даних, що зберігаються в пам'яті комп'ютера або передаються через провідну чи безпровідну мережу. Будь-яка процедура шифрування, що перетворює інформацію зі звичайного «зрозумілого» вигляду в «нечитабельний» зашифрований, звичайно має бути доповнена процедурою дешифрування, яка, у разі застосування до зашифрованого тексту, знову переводить його до зрозумілого вигляду.

**Пара процедур – шифрування і дешифрування – називається криптосистемою.** Зазвичай, криптосистема передбачає наявність спеціального параметра – секретного ключа. Криптосистема вважається розкритою, якщо знайдена процедура, що дозволяє підібрати ключ за реальний проміжок часу. Складність алгоритму розкриття є однією із важливих характеристик криптосистеми і називається криптостійкістю.

У криптографії прийнято правило Керкгоффа, суть якого полягає в тому, що стійкість шифру має визначатися тільки секретністю ключа. Усі стандартні алгоритми шифрування (наприклад, DES, 3DES, AES, PGP) є загальновідомими, але від цього їхня ефективність не зменшується. Система залишається захищеною, якщо зловмиснику відомо все про алгоритм шифрування, але він не знає секретний ключ.

Існує два класи криптосистем – **симетричні** і **асиметричні**. У симетричних схемах шифрування секретний ключ шифрування збігається із секретним ключем дешифрування. У асиметричних системах шифрування відкритий ключ шифрування й секретний ключ дешифрування є різними.

## 3.1 Симетричні криптосистеми

На рис. 3.1 наведено класичну модель симетричної криптосистеми, теоретичні основи якої вперше були викладені в 1949 році в роботі Клода Шеннона. У цій моделі є три учасники: відправник, отримувач і зловмисник. Задачею відправника є передача через відкритий канал деякого повідомлення у захищеному вигляді. Для цього він зашифровує відкритий текст m ключем K і передає шифрований текст K(m). Задача отримувача – розшифрувати K(m) і прочитати повідомлення m. Припускається, що відправник має своє джерело ключа. Згенерований ключ завчасно передається отримувачу через надійний канал. Задача зловмисника полягає в тому, щоб перехопити і прочитати передані повідомлення, а також зімітувати хибні повідомлення.

Найбільш популярним стандартним симетричним алгоритмом шифрування даних є DES (Data Encryption Standard). Алгоритм розроблений фірмою IBM і в 1976 році був рекомендований Національним бюро стандартів для використання у відкритих секторах економіки.

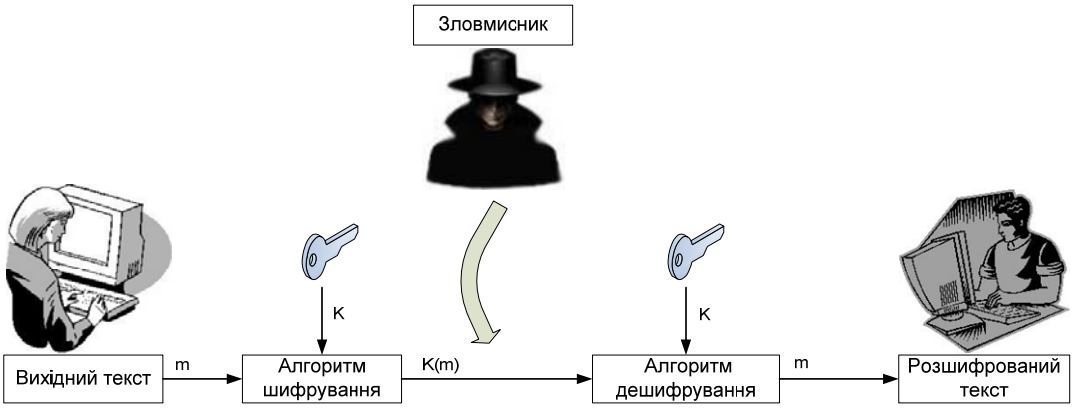


Рисунок 3.1 – Модель симетричного шифрування

Алгоритм DES широко використовується в різноманітних технологіях і продуктах, пов’язаних із безпекою інформаційних систем. Щоб підвищити криптостійкість алгоритму DES, іноді застосовують його підсилений варіант, що має назву «потрійний алгоритм DES» і поєднує в собі триразове шифрування з використанням двох різних ключів. Потрійний алгоритм DES вимагає в три рази більше часу на реалізацію, ніж «звичайний».

У 2001 році Національне бюро стандартів США прийняло новий стандарт симетричного шифрування, який отримав назву AES (Advanced Encryption Standard). AES забезпечує кращий захист, оскільки використовує 128-бітні ключі (підтримує також 192- і 256-бітні) і має більш високу швидкість роботи, кодуючи за один цикл 128-бітний блок на відміну від 64-бітного блоку DES.

У симетричних алгоритмах головною проблемою є ключі. По-перше, криптостійкість багатьох симетричних алгоритмів залежить від якості ключа, це висуває підвищені вимоги до служби генерації ключів. По-друге, принциповою є надійність каналу передачі ключа іншому учаснику секретних переговорів. Несиметричні алгоритми, в основі яких є використання відкритих ключів, усувають цю проблему.

### 3.2 Несиметричні алгоритми шифрування

У 1976 році вчені Уітфілд Діффі і Мартін Хеллман у роботі «Нові напрямки в сучасній криптографії» описали принципово інший підхід до шифрування. Особливість шифрування з відкритим ключем полягає в тому, що одночасно генерується унікальна пара таких ключів, що текст, зашифрований одним ключем, можна розшифрувати тільки з використанням другого ключа, і навпаки.

У моделі криптосхеми з відкритим ключем також три учасники: відправник, отримувач і зловмисник. Задача відправника – через відкритий канал зв’язку передати деяке повідомлення у захищеному вигляді. Отри-

мувач генерує на своєму клієнтському боці два ключі: відкритий $K^+$ і закритий $K^-$. Закритий або особистий ключ $K^-$ абонент має зберігати у захищеному місці, а відкритий ключ $K^+$ він може передати всім, з ким хоче підтримувати захищені відносини. Для шифрування тексту слугує відкритий ключ, але розшифрувати цей текст можна тільки за допомогою закритого ключа. Тому відкритий ключ передається відправнику в незахищеному вигляді. Відправник, використовуючи відкритий ключ отримувача, шифрує повідомлення m і передає його отримувачу. Отримувач розшифровує повідомлення своїм закритим ключем $K^-$ (рис. 3.2).

Хоча інформація про відкритий ключ не є секретною, її потрібно захищати від підробок, щоб зловмисник під ім'ям легального користувача не зміг нав'язати свій відкритий ключ, після чого з допомогою свого закритого ключа він зможе розшифрувати всі повідомлення, адресовані легальному користувачу, і надсилати свої повідомлення від його імені. Вирішенням проблеми є технологія цифрових сертифікатів – електронних документів, які пов'язують конкретних користувачів з конкретними відкритими ключами.

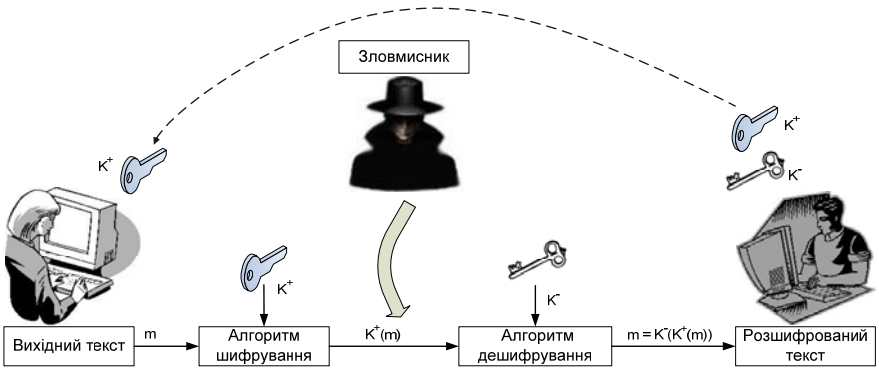


Рисунок 3.2 – Модель криптосистеми з відкритим ключем

### 3.3 Однобічні функції шифрування

У багатьох базових технологіях безпеки використовується ще один підхід шифрування – шифрування за допомогою однобічної функції (one-way function), яка також називається хеш-функцією (hash function) або дайджест-функцією (digest function). Результатом застосування такої функції під час шифрування даних є, так званий, **дайджест**. Довжина дайджесту має фіксоване значення й не залежить від довжини вихідних даних. Знання дайджесту не дозволяє і навіть не припускає відновлення вихідних даних.

Припустимо, що необхідно забезпечити цілісність повідомлення m, яке передається в мережі. Відправник та отримувач домовились, яку хеш-

функцію (ХФ) і з яким значенням параметра – секретного ключа $K^-$ – вони будуть використовувати для вирішення цієї задачі. Перед тим, як надіслати повідомлення m, відправник обчислює для нього дайджест H(m) і надсилає його разом з повідомленням адресату (рис. 3.3, а). Адресат, отримавши дані, застосовує ХФ до переданого у відкритому вигляді повідомлення. Якщо значення обчисленого локально і отриманого з мережі дайджестів збігаються, значить, вміст повідомлення під час передавання не був змінений (рис. 3.3, б).

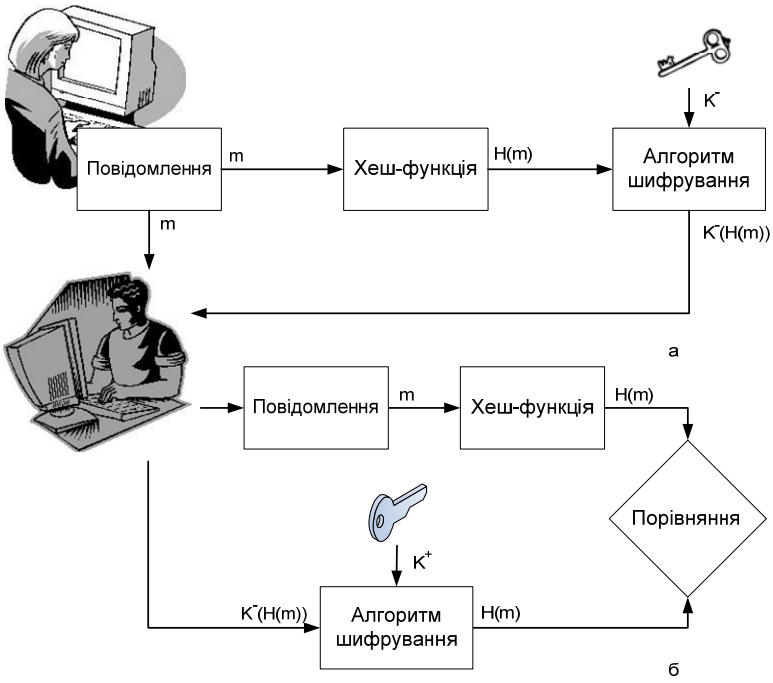


Рисунок 3.3 – Використання хеш-функції для забезпечення цілісності

Окрім забезпечення цілісності повідомлень, дайджест може бути використаний як електронний підпис для аутентифікації документа, що передається. Найпопулярнішою в системах безпеки сьогодні є серія хеш-функцій MD2, MD4, MD5. Усі вони генерують дайджести фіксованою довжиною 16 байтів. Також застосовується більш захищена хеш-функція SHA, яка генерує дайджест довжиною 20 байтів.

# 4 АУТЕНТИФІКАЦІЯ

Термін «аутентифікація» (authentication) походить від латинського слова *authenticus* (справжній, достовірний, такий, що відповідає самому собі). Аутентифікація, або, іншими словами, процедура встановлення аутентичності (достовірності), може застосовуватись як до людей, так і до інших об'єктів, зокрема, до програм, пристроїв, документів. **Аутентифікація користувача – це процедура доведення користувачем того, що він саме той, за кого себе видає**. У процедурі аутентифікації беруть участь дві сторони: одна сторона доводить свою аутентичність, надаючи деякі відомості, інша сторона – аутентифікатор – перевіряє ці відомості і приймає рішення. Для доведення аутентичності застосовуються найрізноманітніші прийоми, наприклад, той, хто проходить аутентифікацію, може: продемонструвати знання певного спільного для обох сторін секрету – пароля; продемонструвати, що він володіє певним унікальним предметом (фізичним ключем), функцію якого може виконувати, наприклад, електронна магнітна карта; довести свою аутентичність, використовуючи власні біологічні характеристики: зображення райдужної оболонки ока або відбитки пальців, які попередньо були внесені в базу даних аутентифікатора.

## 4.1 Мережеві служби аутентифікації

Мережеві служби аутентифікації будуються на основі всіх цих прийомів, але найчастіше для доведення ідентичності користувача застосовують паролі. Простота і логічна ясність механізмів аутентифікації на основі паролів, певною мірою, компенсує відомі слабкості паролів. Для зменшення рівня загрози розкриття паролів адміністратори мережі, як правило, застосовують вбудовані програмні засоби для формування політики призначення і використання паролів.

Як уже зазначалось, об'єктами, що потребують аутентифікації, можуть бути не тільки користувачі, а й різноманітні програми, пристрої, текстова та інша інформація. Наприклад, користувач, котрий звертається із запитом до корпоративного веб-сервера, повинен довести йому свою легальність, але й сам він має упевнитись в тому, що веде діалог саме з веб-сервером свого підприємста. Тобто сервер і клієнт повинні пройти процедуру взаємної аутентифікації. Це приклад аутентифікації на програмному рівні.

Під час встановлення зв'язку між двома пристроями також часто передбачається процедура взаємної аутентифікації пристроїв на нижчому канальному рівні. Під аутентифікацією даних розуміють доведення цілісності цих даних, а також того, що вони надійшли саме від тієї людини, яка повідомила про це. Для цього використовується механізм електронного підпису.

## 4.2 Дайджест-аутентифікація

Дайджест-аутентифікація є досить поширеною і використовується в протоколі CHAP (Challenge Handshake Authentication Protocol), що належить до сімейства протоколів PPP, у протоколах аутентифікації операційної системи Windows тощо. Цей підхід використовується, наприклад, під час аутентифікації віддалених користувачів, підключених до Інтернету по комутованому каналу. Аутентифікатором є сервер провайдера, а об'єктом аутентифікації – комп'ютер клієнта. Під час укладання договору клієнт отримує від провайдера пароль P (рис. 4.1).

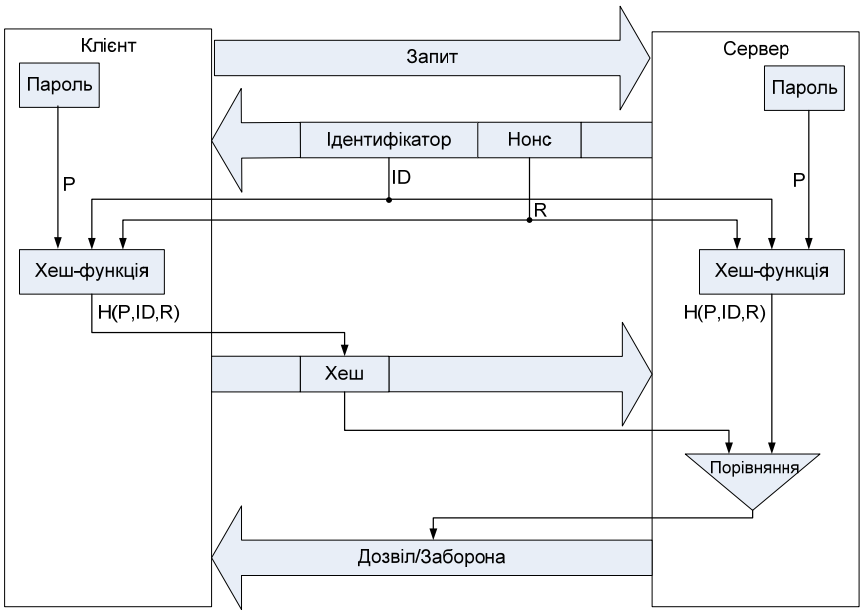


Рисунок 4.1 – Схема дайджест-аутентифікації

Аутентифікація виконується у такій послідовності:

1. Клієнт активує програму віддаленого доступу до сервера провайдера і відправляє запит на встановлення з'єднання.

2. Сервер, отримавши запит від клієнта, генерує псевдовипадкове слово – нонс (R) і передає його клієнту разом зі значенням, що ідентифікує повідомлення в межах цього сеансу (ID). Для захисту від перехоплення відповіді аутентифікатор має використовувати різні значення нонсу під час кожної процедури аутентифікації.

3. Програма клієнта, отримавши такий пакет, вилучає з нього нонс (R) та ідентифікатор і додає пароль (P), а потім за допомогою функції MD5 знаходить дайджест H(P,ID,R). Результат клієнт надсилає на сервер.

4. Сервер порівнює отриманий з мережі дайджест з тим значенням, яке він отримав, локально застосувавши ту ж хеш-функцію до набору аналогічних компонентів.

5. Якщо результати збігаються, то аутентифікація вважається успішною і аутентифікатор відсилає партнеру дозвіл на з'єднання, в іншому випадку – заборону.

## 4.3 Аутентифікація на основі одноразового паролю

Алгоритми аутентифікації, що базуються на багаторазових паролях, є не досить надійними. Паролі можна підглянути, розгадати або просто вкрасти. Надійнішими є схеми з одноразовими паролями. До того ж одноразові паролі набагато дешевші й простіші, ніж біометричні системи аутентифікації, наприклад, сканери сітківки ока або відбитків пальців. Усе це робить системи, що базуються на використанні одноразових паролів, дуже перспективними. Однак потрібно враховувати, що зазвичай системи аутентифікації на основі одноразових паролів розраховані на перевірку тільки віддалених, а не локальних користувачів.

**Генерація одноразових паролів може виконуватися або програмно, або апаратно**. Апаратні реалізації систем доступу на основі одноразових паролів називають апаратними ключами. Вони є мініатюрними пристроями із вбудованим мікропроцесором, схожі або на звичайні пластикові картки, які використовуються для доступу до банкоматів, або на кишенькові калькулятори з клавіатурою і маленьким дисплейним вікном. Також існують і програмні реалізації засобів аутентифікації на основі одноразових паролів – програмні ключі, що розташовуються на змінному магнітному носії у вигляді звичайної програми, важливою частиною якої є генератор одноразових паролів.

## 4.4 Аутентифікація на основі сертифікатів

Аутентифікація з використанням цифрових сертифікатів є альтернативою використанню паролів і є природним рішенням в умовах, коли кількість користувачів мережі (нехай навіть потенційних) вимірюється мільйонами. За таких обставин процедура попередньої реєстрації користувачів, пов’язана з призначенням і зберіганням їхніх паролів, стає вкрай обтяжливою, небезпечною, а іноді навіть нездійсненною. За наявності сертифікатів мережа, яка дає користувачу доступ до своїх ресурсів, не зберігає жодної інформації про своїх користувачів – вони надають її самостійно у своїх запитах у вигляді сертифікатів, що підтверджують особу користувача. **Сертифікати видаються спеціальними уповноваженими організаціями – центрами сертифікації** (Certificate Authority, CA). Таким чином задача зберігання секретної інформації (закритих ключів) покладається на самих користувачів, що робить такий підхід набагато кращим для масштабування, ніж варіант із централізованою базою паролів.

Сертифікат – електронна форма, що містить таку інформацію: відкритий ключ власника цього сертифіката; відомості про власника сертифіката, наприклад, ім’я, адресу електронної пошти, назву організації, у якій він працює тощо; назву сертифікаційної організації, що видала такий сертифікат; електронний підпис сертифікаційної організації, тобто зашифровані закритим ключем цієї організації дані, які містить сертифікат.

**Використання сертифікатів базується на припущенні, що сертифікаційних організацій небагато і їхні відкриті ключі широко доступні**, наприклад, завдяки публікаціям у журналах. Коли користувач хоче підтвердити свою особистість, він пред'являє свій сертифікат у двох формах: відкритій (тобто в такій, у якій він отримав його в сертифікаційній організації) і зашифрованій з використанням свого закритого ключа. Сторона, що проводить аутентифікацію, бере із незашифрованого сертифіката відкритий ключ користувача і за допомогою нього розшифровує зашифрований сертифікат. Збіг результату з відкритим сертифікатом підтверджує, що пред'явник дійсно є власником закритого ключа, який відповідає вказаному відкритому. Далі за допомогою відомого відкритого ключа вказаної у сертифікаті організації проводиться розшифрування підпису цієї організації у сертифікаті. Якщо в результаті виходить той самий сертифікат з таким самим ім'ям користувача і його відкритим ключем, а отже, він дійсно пройшов реєстрацію в сертифікаційному центрі.

**Сертифікат засвідчує не тільки особистість, а й приналежність відкритого ключа.** Цифровий сертифікат встановлює і гарантує відповідність між відкритим ключем і його власником. Це запобігає загрозі підміни відкритого ключа. Якщо деякий абонент А отримує через мережу сертифікат від абонента Б, то він може бути впевнений, що відкритий ключ, який знаходиться в сертифікаті, гарантовано належить абоненту Б, адреса й інші відомості про якого знаходяться в цьому сертифікаті. Це означає, що абонент А може без побоювань використовувати відкритий ключ абонента Б для секретних повідомлень на адресу останнього.

**Сертифікат є засобом аутентифікації користувача під час його доступу до мережевих ресурсів**, роль аутентифікуючої сторони при цьому відіграють інформаційні сервери корпоративної мережі або Інтернету. Водночас і сама процедура отримання сертифіката містить етап аутентифікації, коли аутентифікатором є сертифікаційна організація. Для отримання сертифікату клієнт має повідомити сертифікаційній організації свій відкритий ключ і ті чи інші відомості, що засвідчують його особистість. Усі ці дані клієнт може надіслати електронною поштою або принести на змінному носії особисто. Перелік необхідних даних залежить від типу одержуваного сертифікату. Сертифікаційна організація перевіряє докази аутентичності, заносить свій цифровий підпис у файл, що містить відкритий ключ, і відсилає сертифікат назад, підтверджуючи факт приналежності конкретного ключа конкретній особистості. Після цього сертифікат може бути вбудований у будь-який запит на використання інформаційних ресурсів мережі.

### 4.5 Аутентифікація інформації

Під поняттям аутентифікації інформації в комп'ютерних системах розуміють встановлення аутентичності отриманих через мережу даних винятково на основі інформації, що знаходиться в отриманому повідомленні. Виділяють два види аутентифікації інформації: аутентифікація масивів даних і програм, що зберігаються.

Для вирішення задачі аутентифікації інформації використовується **концепція цифрового, або електронного, підпису**. Відповідно до термінології, затвердженої Міжнародною організацією зі стандартизації (ISO), термін «цифровий підпис» визначає методи, що дозволяють установлювати аутентичність автора повідомлення (документа) під час виникнення суперечки щодо авторства. Для побудови схеми цифрового підпису широко застосовується алгоритм RSA. Цей алгоритм базується на концепції Діффі-Хеллмана, відповідно до якої кожен користувач мережі має свій закритий ключ, необхідний для формування підпису, а відкритий ключ, що відповідає цьому секретному ключу і призначений для перевірки підпису, відомийусім іншим користувачам мережі. На рис. 4.2 показана схема формування цифрового підпису за алгоритмом RSA. Підписане повідомлення складається з двох частин: незашифрованої частини, у якій знаходиться вихідний тест m, і зашифрованої частини, що є цифровим підписом. Цифровий підпис S обчислюється з використанням закритого ключа $K^-$ за формулою: $S= K^-(m)$. Повідомлення надсилається у вигляді пари (m,S). Кожен користувач, котрий має відповідний відкритий ключ ($K^+$), отримавши повідомлення, розшифровує цифровий підпис S. Якщо результат розшифрування цифрового підпису збігається з відкритою частиною повідомлення, то документ є справжнім, не був змінений під час передачі, а його автором є саме та людина, котра передала свій відкритий ключ отримувачу. Якщо повідомлення містить цифровий підпис, то отримувач може бути впевнений, що воно не було змінене або підроблене.

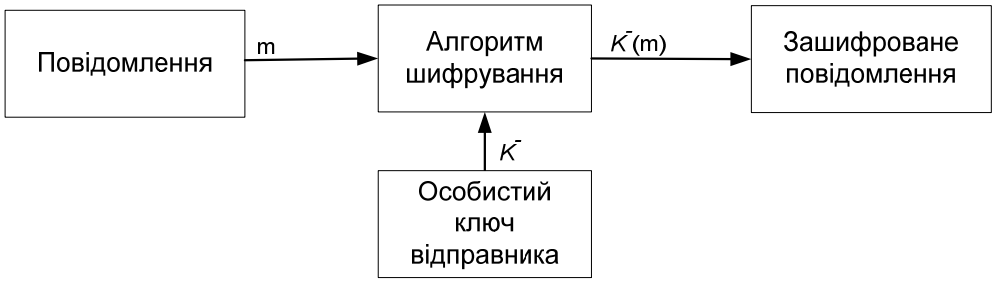


Рисунок 4.2 – Створення цифрового підпису документа

### 4.6 Авторизація та аудит

Окрім процедури аутентифікації, у комп'ютерних системах та мережах постають задачі авторизації користувачів та аудиту їхніх дій.

**Авторизація** – це процедура контролю доступу легальних користувачів до ресурсів системи і надання кожному з них саме тих прав, які були призначені адміністратором. На відміну від аутентифікації, яка дозволяє розпізнати легальних і нелегальних користувачів, авторизація має справу

тільки з легальними користувачами, котрі успішно пройшли процедуру аутентифікації. Окрім надання користувачам прав доступу до каталогів, файлів і принтерів, засоби авторизації можуть контролювати можливість виконання користувачами різних системних функцій, зокрема, локальний доступ до сервера, установлення системного часу, створення резервних копій даних, вимкнення сервера тощо.

**Процедури авторизації часто поєднуються з процедурами аутентифікації** і реалізуються одними й тими ж програмними засобами, які можуть як вбудовуватись в операційну систему або програму, так і додатково інсталюватися у вигляді окремих програмних продуктів. При цьому програмні системи аутентифікації і авторизації можуть бути побудовані на базі двох схем – централізованої та децентралізованої.

**Централізована схема** базується на використанні спеціального сервера. У цій схемі сервер керує процесом надання користувачу мережених ресурсів. Головна роль таких систем – реалізувати «принцип єдиного входу». Відповідно до централізованої схеми користувач один раз логічно входить у мережу і отримує на увесь проміжок часу роботи деякий набір дозволів на доступ до різноманітних ресурсів. Системи Kerberos, Shibboleth, TACACS і RADIUS є прикладами реалізації такого підходу.

**Децентралізована схема**, що базується на робочих станціях. При використанні цього підходу засоби авторизації працюють на кожній машині. Адміністратор має відслідковувати роботу механізмів безпеки кожної окремої програми – електронної пошти, довідкової служби, локальних баз даних тощо.

**Аудит** (auditing) – це набір процедур моніторингу й обліку всіх подій, що становлять потенційну загрозу для безпеки системи. Аудит дозволяє «шпигувати» за обраними об’єктами й видавати повідомлення тривоги, коли, наприклад, звичайний користувач спробує прочитати чи модифікувати системний файл. Якщо хтось намагається виконати дії, обрані системою безпеки для моніторингу, то система аудиту записує відповідне повідомлення в журнал реєстрації небезпечних подій. Системний менеджер може робити звіти безпеки, які містять інформацію із цього журналу. Функції аудиту вбудовуються в різні засоби безпеки: мережеві екрани, системи виявлення вторгнень, антивірусні системи, мережеві монітори.

## 4.7 Реалізація AAA сервісу в мережевому обладнанні

### 4.7.1. Огляд та характеристики AAA

Для запобігання доступу зловмисників до мережевих пристроїв та мережі загалом використовуються методи керування доступом. Ці методи дозволяють визначити, хто та якою мірою може скористатись тим чи ін-

шим ресурсом або сервісом. Використовуються багато методів аутентифікації і кожний з них гарантує певний рівень безпеки.

Найпростіша форма аутентифікації – це паролі, але цей метод є найбільш уразливим до атак. Для реалізації аутентифікації доступу до мережевого пристрою може бути використано локальну базу даних, облікові записи в яку заносяться за допомогою однієї з команд:

```
Router(config)# username username password password
Router(config)# username username secret password
```

Цей метод дозволяє створити кожному користувачу індивідуальний обліковий запис на кожному пристрої. Захищеність підвищується за рахунок того, що зловмиснику, окрім пароля, треба ще знати ім'я користувача.

Використання локальної бази даних має багато обмежень, зокрема, це дуже великий обсяг роботи у великій мережі, складна підтримка тощо. Значно кращий підхід – використання єдиної бази даних для всієї установи і розташування її на окремому сервері.

AAA сервіс (Authentication, Authorization, Accounting) забезпечує базову структуру для організації керування доступом до мережевих пристроїв. AAA – це шлях для керування доступом до мережі (Аутентифікація), визначення прав після отримання доступу (Авторизація) та відслідковування, хто що робив (Аудит). Такий підхід забезпечує значно вищий рівень масштабованості, порівняно з локальними обліковими записами.

**AAA Аутентифікація.** AAA може використовуватись для аутентифікації користувачів, що бажають отримати адміністративний доступ для керування обладнанням, так званий, символьний режим, або просто доступ у мережу з іншої мережі – пакетний режим.

У символьному режимі користувач відправляє запит на доступ до того чи іншого мережевого пристрою для адміністративних цілей, наприклад, налаштування маршрутизації або перегляду поточного стану пристрою. У пакетному режимі користувач відправляє запит для встановлення зв'язку з певною мережею для отримання дозволу на проходження трафіку в певну мережу. Існують два загальних методи реалізації AAA сервісів – локальний і серверний, до того ж серверний варіант є основним, а локальний використовується як допоміжний у разі відсутності зв'язку з сервером. При застосуванні серверного підходу використовується зовнішній сервер баз даних, який працює через протоколи RADIUS або TACACS+ і може бути реалізований як програмно на базі серверної операційної системи, так і програмно-апаратно у вигляді спеціалізованого пристрою.

**AAA Авторизація.** Після того, як користувачі були успішно аутентифіковані за допомогою тієї чи іншої бази даних (локальної або серверної), вони проходять авторизацію для певних мережевих ресурсів. Авторизація фактично визначає, що користувач може, а що не може робити в мережі, аналогічно до того, як визначаються різні рівні привілеїв у ролевому CLI. Авторизація, зазвичай, здійснюється з використанням серверного рішення.

Авторизація використовує набір атрибутів, що описує доступ користувача до мережі. Ці атрибути порівнюються з інформацією, що зберігається в базі даних, і набір обмежень для цього користувача передається на локальний маршрутизатор, до якого він підключився. Як правило, авторизація виконується автоматично і не потребує окремих дій користувача після аутентифікації, а в деяких протоколах, наприклад, RADIUS, ці процеси взагалі об'єднані.

**AAA Аудит.** Сервіс аудиту накопичує і готує звіти щодо використання ресурсів з метою обліку й аудиту. Дані, які накопичуються, можуть містити часові позначки початку й кінця з'єднання, перелік команд, що запускав користувач, кількість пакетів або байтів, що були передані тощо. Сервіс аудиту реалізується із застосуванням серверного рішення. Відповідна статистика відсилається на сервер і може бути використана для створення детальних звітів. Застосування аудиту разом з аутентифікацією дозволяє фіксувати, який користувач і що робив, коли під'єднувався до мережі, або окремих пристроїв, які команди виконував тощо. Лог-файли містять велику кількість полів, а саме: ім'я користувача, час, команду, що була введена. Ця інформація є корисною під час пошуку несправностей.

### 4.7.2 Локальна AAA аутентифікація

Використовує локальну базу даних для аутентифікації. У цьому випадку імена та паролі записуються безпосередньо в конфігураційному файлі маршрутизатора, й аутентифікація відбувається з використанням локальної інформації. Фактично, це та ж база даних, що використовується для налаштування CLI на основі ролей. Є ідеальним підходом для невеликої кількості пристроїв або як резервний варіант при неможливості здійснити серверну аутентифікацію. Локальна, або автономна AAA аутентифікація може бути використана для малих мереж. Цей метод використовує імена й паролі, що записані локально на маршрутизаторі. Конфігурування локальної AAA аутентифікації для символьного режиму передбачає кількох кроків: створення в локальній базі облікових записів користувачів, що складаються з імені й пароля; увімкнення AAA-сервісу на маршрутизаторі; конфігурування AAA параметрів на маршрутизаторі; перевірка та виправлення помилок. Увімкнення AAA-сервісу: `Router(config)#aaa new-model.`

Після активації AAA-сервісу для конфігурування AAA на консольний доступ (console) та доступ у режимі віддаленого терміналу (vty), необхідно визначити іменований список методів аутентифікації і призначити його до відповідного інтерфейсу. Для створення списку аутентифікації: `Router(config)#aaa authentication login.`

Повний перелік методів аутентифікації з відповідними параметрами наведено у табл. 4.1. Як параметри використовують ім'я списку та перелік методів аутентифікації, що будуть використані під час реєстрації користувача.

Таблиця 4.1 – Перелік методів аутентифікації

| Ключові слова методу | Опис |
|---|---|
| **enable** | Застосовує enable password для аутентифікації |
| **krb5** | Застосовує Kerberos 5 для аутентифікації |
| **krb5-telnel** | Застосовує протокол аутентифікації Kerberos 5 telnet при використанні telnet для під'єднання до роутера |
| **line** | Застосовує line password для аутентифікації |
| **local** | Використовує локальну базу даних користувачів для аутентифікації |
| **local-case** | Враховує чутливість регістра при введенні username |
| **None** | Не застосовує аутентифікацію |
| **cache *group-name*** | Застосовує cache server group для аутентифікації |
| **group radius** | Застосовує список усіх RADIUS серверів для аутентифікації |
| **group tacacs+** | Застосовує список всіх TACACS+ серверів для аутентифікації |
| **group *group-name*** | Застосовує частковий список RADIUS чи TACACS+ серверів для аутентифікації, згідно з визначеним у сервісі **aaa** параметром |

Використання кількох методів аутентифікації дозволяє забезпечити доступ до системи, якщо один з них не може бути використаний. Наприклад, **enable** метод може бути сконфігурований як запасний варіант у разі, якщо заблоковано або видалено обліковий запис користувача:

```
Router(config)#aaa authentication login TELNET-ACCESS local
enable
```

У цьому випадку сконфігуровано локальну аутентифікацію, однак якщо вона не спрацює за певних причин, користувач зможе аутентифікуватись за паролем, що встановлено на режим enable. В одному списку методів може бути визначено мінімум 1 і максимум 4 методи. Спроби реєстрації здійснюються в тому порядку, у якому вказані методи в списку. Далі використовується метод, коли на попередній не отримано відповіді або відбулась помилка сервісу. Якщо в результаті аутентифікації блокується доступ, наприклад, перебільшено задану кількість спроб ввести вірний пароль, то наступний метод не пропонується. Визначений перелік методів аутентифікації має бути асоційований з певним інтерфейсом або лінією. Різні переліки аутентифікації можуть використовуватись на різних лініях та інтерфейсах. Для асоціювання списку методів з лінією або інтерфейсом використовується команда:

```
Router(config-line)#aaa login authentication list-name
```

Є також опція, яка дозволяє конфігурувати перелік методів аутентифікації за замовчуванням. Після активації AAA-сервісу перелік методів з

іменем **default** автоматично асоціюється з усіма інтерфейсами й лініями. Для призначення переліку методів, що входять до списку за замовчуванням, використовується команда:

```
Router(config)#aaa authentication login default
method1...[method2]
```

Якщо визначено і default список, і персональний перелік методів аутентифікації, для того чи іншого інтерфейсу або лінії перевага буде надаватись персональному, якщо немає ні default, ні персонального списку, для доступу використовується локальна база даних. Це матиме той же ефект, якби було вказано команду:

```
Router(config)#aaa authentication login default local.
```

У такому разі консольний доступ відбуватиметься без перевірки паролів. Якщо було визначено персональний перелік методів, а потім виникла необхідність від них відмовитись, дається та ж команда, що й під час створення списку тільки з «no» на початку. Таким чином, актуальним буде default перелік. Приклад налаштування:

```
R1(config)# username JR-ADMIN secret Str0ngPa55w0rd
R1(config)# username ADMIN secret Str0ngPa55w0rd
R1(config)# aaa new-model
R1(config)# aaa authentication login default local-case ena-
ble
R1(config)# aaa authentication login TELNET—LOGIN local-case
R1(config)# line vty 0 4
R1(config-line)# login authentication TELNET—LOGIN
```

Додаткова безпека може бути застосована на лінії шляхом визначення максимальної кількості невдалих спроб аутентифікації:

```
Router(config)#aaa local authentication attempts max-fail
number-of-unsuccessful-attempts.
```

Для перегляду переліку заблокованих користувачів є команда:

```
Router#show aaa local user lockout.
```

Для розблокування всіх або окремих користувачів є команда:

```
Router#clear aaa local user lockout {username username |
all}.
```

На відміну від команди login delay при використанні max-fail облікові записи залишаються заблокованими доти, доки системний адміністратор їх не розблокує. Після реєстрації користувача унікальний ID асоціюється із сесією та атрибути починають накопичуватись у базі даних, а саме: IP-адреса користувача, ідентифікатор протоколу, кількість переданих байтів тощо. Для перегляду атрибутів, що накопичились для сесії, є команда:

```
Router#show aaa user {all | unique id}.
```

Ця команда не надає інформацію про всіх користувачів, хто зареєстрований на пристрої, а тільки про тих, хто аутентифікований і авторизований з використання AAA. Для того, щоб визначити ID сесії, використовується команда: `Router#show aaa sessions.`

Для пошуку проблем з AAA аутентифікацією є команда: `Router#debug aaa authentication.`

### 4.7.3 Серверний варіант AAA

Локальний варіант AAA-сервісу має обмежені можливості щодо масштабування. Тому у великих організаціях, як правило, використовується серверний варіант. У разі застосування серверної аутентифікації має місце така послідовність взаємодії клієнта та AAA сервера (рис. 4.3):

1. Користувач встановлює з'єднання з маршрутизатором.
2. Маршрутизатор запитує ім'я та пароль користувача.
3. Маршрутизатор відправляє цю інформацію на сервер.
4. Сервер аутентифікує користувача й авторизує на маршрутизаторі з правами відповідно до бази даних.

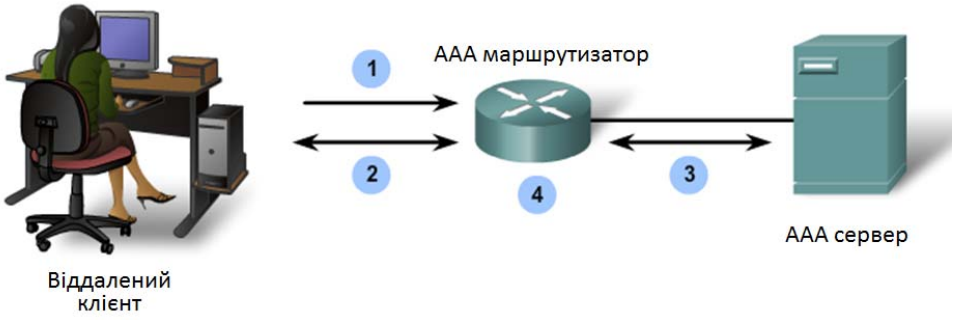


Рисунок 4.3 – Послідовність взаємодії клієнта та AAA сервера

Для забезпечення взаємодії між клієнтом та AAA-сервером застосовуються такі протоколи: Terminal Access Control Access Control Server Plus (TACACS+) та Remote Dial-in User Services (RADIUS). TACACS+ вважається більш захищеним, оскільки вся інформація шифрується. RADIUS шифрує тільки пароль користувача. Порівняльна характеристика протоколів TACACS+ та RADIUS наведена в табл. 4.2. Вибір протоколу залежить від потреб організації. Наприклад, великі ISP надають перевагу RADIUS, тому що він підтримує більш детальну інформацію по обліковому запису, що є необхідним для системи обліку (billing). Організації з різними групами користувачів можуть надавати перевагу TACACS+, оскільки він під-тримує аутентифікаційні політики окремих користувачів та груп.

Критичні параметри характерні для TACACS+ : несумісний з попередниками TACACS та XTACACS; розділяє аутентифікацію і авторизацію; шифрує увесь контент; працює через TCP порт 49. Критичні параметри для RADIUS: використовує RADIUS проксі-сервер для масштабованості; об'єднує аутентифікацію й авторизацію в єдиному процесі; шифрує тільки паролі; використовує UDP; підтримує технології віддаленого доступу 802.1X та SIP

Таблиця 4.2 – Порівняння протоколів TACACS+ та RADIUS

| Особливість | TACACS+ | RADIUS |
|---|---|---|
| Функціональність | Розділяє процедури аутентифікації, авторизації та аудиту, що підвищує гнучкість застосування | Процедури аутентифікації та авторизації об'єднані, аудит відокремлений, що зменшує гнучкість, порівняно з TACACS+ |
| Стандартизація | Здебільшого, підтримується компанією Cisco | Відкритий стандарт RFC 2865 |
| Транспортний протокол | TCP порт 49 | UDP порт 1645 або 1812 для аутентифікації, UDP порт 1646 або 1813 для аудиту |
| Конфіденційність | Увесь пакет шифрується | Тільки пароль шифрується |
| Аудит | Обмежені можливості | Розширені можливості |

### 4.7.4 Конфігурування серверного варіанта на маршрутизаторах

На відміну від локальної AAA аутентифікації, серверна передбачає ідентифікацію TACACS+ та RADIUS серверів, з якими AAA сервіс має консультуватись для аутентифікації і авторизації користувачів.

Основні етапи налаштування серверної аутентифікації:

1. Глобально ввімкнути AAA-сервіс, щоб дозволити використання всіх AAA елементів.

2. Указати адреси серверів аутентифікації.

3. Налаштувати ключ шифрування, який буде використовуватись для шифрування сеансу між AAA-сервером та клієнтом.

4. Сконфігурувати перелік методів аутентифікації. Приклад налаштування серверного варіанту аутентифікації наведено нижче:

```
R1(config)# aaa new-model
R1(config)# tacacs-server host 192.168.1.101 single-
connection
R1(config)# tacacs-server key TACACS+Pa55w0rd
R1(config)# radius-server host 192.168.1.100
R1(config)# radius-server key RADIUS-Pa55w0rd
R1(config)# aaa authentication login default group tacacs+
group radius local-case
```

### 4.7.5 Авторизація і аудит

У той час, як аутентифікація має справу з перевіркою легітимності користувача або кінцевого пристрою, авторизація призначена для надання аутентифікованим користувачам доступу до певних зон або програм у мережі. TACACS+ протокол дозволяє відділити аутентифікацію від авто-

ризації. Застосування AAA-авторизації на маршрутизаторі дозволяє регламентувати перелік дій користувача або групи користувачів після успішної реєстрації. Авторизацію можна сконфігурувати як для символьного режиму (перелік доступних дій на маршрутизаторі), так і для пакетного (обмеження на передачу трафіку). Важливим аспектом авторизації є можливість керувати доступом користувача до окремих команд. Наприклад, авторизованому користувачу може бути дозволений доступ до команди show version, але заборонений до configure terminal. Маршрутизатор буде запитувати сервер про дозвіл виконувати команду користувача, і буде, відповідно, отримувати у відповідь дозвіл або заборону. Для конфігурування авторизації використовується команда:

```
R1(config)# aaa authorization{network | exec | commands
level} {default | list-name} method1…[method4]
```

Тип сервіса може визначити типи команд або сервісів: **commands level** – для exec команд; **exec** – для запуску exec; **network** – для мережевих сервісів (PPP, SLIP, ARAP). Допоки AAA-авторизація не увімкнена, доти всі користувачі отримують повний доступ до команд інтерфейсу командного рядка. Якщо авторизацію активовано, то після завершення проходження аутентифікації, за замовчуванням доступ до всіх команд буде заборонено. Це означає, що адміністратор обов'язково має створити користувача з правами повного доступу перед увімкненням процедури авторизації. Якщо цього не зробити, систему буде відразу заблоковано після введення команди:

```
Router(config)#aaa authorization.
```

Приклад налаштування AAA-авторизації на маршрутизаторі:

```
R1(config)# aaa authorization exec default ?
group               Use server-group.
if-authenticated    Succeed if user has authenticated.
krb5-instance       Use Kerberos instance privilege maps.
local               Use local database.
None                No authorization (always succeeds).
R1(config)# aaa authorization exec default group ?
WORD                Server-group name
radius              Use list of all Radius hosts.
tacacs+             Use list of all Tacacs+ hosts.
R1# conf t
Rl(config)# username JR—ADMIN secret Str0ngPa55v0rd
R1(config)# username ADMIN secret Str0ngPa55v0rd
Rl(config)# aaa new-model
Rl(config)# aaa authentication login default group tacacs+
R1(config)# aaa authorization exec default group tacacs+
Rl(config)# aaa authorization network default group tacacs+
```

Досить часто виникає необхідність відслідковувати, які ресурси використовували окремі користувачі або групи. AAA-аудит забезпечує можливість відслідковувати використання ресурсів через віддалений доступ,

формувати log-повідомлення на основі даних, що збережені в базі даних, готувати відповідні звіти. Хоча аудит більше стосується питань керування мережею та фінансового керування, однак питань безпеки цей сервіс також стосується. При застосуванні функції AAA-аудиту сервер є центральним сховищем інформації аудиту, фактично, відслідковуючи події, що сталися в мережі. Кожна сесія, яка була встановлена через сервер, може бути повністю відслідкована, задокументована й збережена на сервері. Аналогічно до переліку методів аутентифікації і авторизації, перелік методів для аудиту визначає шлях, яким здійснюється аудит і послідовність, у якій ці методи застосовуються. Після увімкнення, перелік методів аудиту за замовчуванням автоматично застосовується до всіх інтерфейсів, за винятком тих, що мають іменований перелік. Для конфігурування AAA-аудиту використовується команда:

```
default | list-name } {start-stop | stop-only | none}
[broacast] method1…[method4]
```

Ключові слова мають таке призначення: `network` – запускає аудит для всіх сервісів доступу до мережі, наприклад, PPP; `exec` – дозволяє фіксувати хто і коли переходив в exec режим на маршрутизаторі; `connection` – запускає аудит на всіх низхідних з'єднаннях, наприклад, Telnet.

Так само, як і у випадку з AAA-аутентифікацією, для визначення переліку дій використовується або ключове слово default, або іменований перелік. Далі визначається тип запису, або тригер. Тригер визначає, які дії призводять до створення запису. Можливі варіанти: `none`, `start-stop`, або `stop-only`. Наприклад, вибір тригеру start-stop для exec режиму дає команду фіксувати всіх користувачів, що заходили в exec режим та виходили з нього. Нижче наведено приклад, у якому виконано налаштування всіх трьох AAA-сервісів:

```
R1(config)# aaa accounting exec start-stop group tacacs+
R1(config)# aaa accounting network start-stop group tacacs+
Приклад.
R1# conf t
R1(configq)# username JR-ADMIN secret Str0ngPa55w0rd
R1(configq)# username ADMIN secret Str0ngPa55w0rd
R1(config)# aaa new-model
R1(config)# aaa authentication login default group tacaca+
R1(config)# aaa authorization exec default group tacaca+
R1(ccnfig)# aaa authorization network default group tacaca+
R1(config)# aaa accounting exec default start-stop group
tacaca+
R1(config)# aaa accounting network default start-stop group
tacacs
```

# 5 РЕАЛІЗАЦІЯ БЕЗПЕЧНОГО ПЕРИМЕТРУ

**Периметром** у комп'ютерних мережах називають захищений кордон мережі, до складу якого можуть входити такі компоненти: прикордонні маршрутизатори; мережеві екрани; проксі-сервери; системи виявлення вторгнень; демілітаризована зона.

## 5.1 Прикордонні маршрутизатори та типи міжмережевих екранів

Прикордонний маршрутизатор (border router) є останнім маршрутизатором, який знаходиться під контролем адміністратора і через який проходить увесь трафік, спрямований у мережу Інтернет, або такий, що надходить зовні. У деяких випадках прикордонний маршрутизатор створює єдину лінію захисту мережі шляхом фільтрації трафіку, що через нього проходить. Сучасні прикордонні маршрутизатори можуть виконувати також і функції інших елементів захисту.

**Міжмережевий екран** (МЕ) – це комплекс програмно-апаратних засобів, що здійснює інформаційний захист однієї частини комп'ютерної мережі від іншої шляхом аналізу трафіку, що проходить між ними. У літературі досить часто ці пристрої фігурують під назвами **брандмауер** та **фаєрвол** (firewall). Для реалізації контролю доступу МЕ має підтримувати такі функції: аналізувати, контролювати і регулювати трафік (функція фільтрації); бути логічним посередником між внутрішніми клієнтами та зовнішніми серверами (функція проксі-сервера); фіксувати всі події, пов'язані з безпекою (функція аудиту). Поряд з базовими функціями МЕ може виконувати й інші функції захисту, зокрема, виявлення вторгнень та атак, шифрування трафіку, антивірусний захист, фільтрація повідомлень за вмістом, функції VPN тощо. Одним із підходів щодо класифікації МЕ є розподіл їх на типи залежно від рівня моделі OSI, на якому вони працюють.

**МЕ мережевого рівня**, або екрани з фільтрацією пакетів (packet filtering firewall), вирішують задачу фільтрації пакетів за IP-адресами і протокольними портами. Фільтрація на основі статичних правил, при якій не відслідковуються стани з'єднань, називається простою фільтрацією (stateless packet filtering). Цьому типу мережевих екранів відповідають маршрутизатори з налаштованими списками керування доступом (access control list ACL). Перевагами такого рішення є простота, невисока вартість і мінімальний вплив на продуктивність мережі.

**МЕ сеансового рівня** відслідковують стан з'єднань. Вони фіксують підозрілу активність, спрямовану на сканування портів і збирання іншої інформації про мережу. Відслідковування станів з'єднань полягає в тому, що МЕ перевіряє, наскільки послідовність обміну повідомленнями відповідає контрольованому протоколу. Наприклад, якщо клієнт відсилає TCP-повідомлення SYN – запит на з'єднання, сервер має відповісти повідомленням ACK SYN, а не відсилати у відповідь, скажімо, свій TCP-запит

SYN. Після того, як МЕ установив можливість TCP-з'єднання, він починає працювати як звичайна передавальна ланка між клієнтом і сервером, причому фіксується поточний стан з'єднання, тобто запам'ятовується, яке останнє повідомлення відправив клієнт і яке повідомлення він очікує отримати. Такий підхід, коли пропускаються тільки ті пакети, які задовольняють логіку роботи відповідного протоколу, називають фільтрацією з урахуванням контексту (stateful packet inspection). Завдяки такій можливості брандмауери сеансового рівня можуть захищати сервери внутрішньої мережі від різноманітних видів атак, що використовують уразливі місця протоколів, зокрема, від DoS-атак.

**МЕ прикладного рівня**, зазвичай, називають проксі-серверами. Вони здатні інтерпретувати, аналізувати й контролювати вміст повідомлень, якими обмінюються програми. Проксі-сервер перехоплює запити клієнтів до зовнішніх серверів для того, щоб потім відправити їх від свого імені. Такий тип МЕ забезпечує найвищий рівень захисту, але вимагає великих обчислювальних витрат. Реалізація МЕ є такою ж багатоваріантною, як і його функціональність. Апаратна складова МЕ може бути реалізована на основі маршрутизатора або сукупності маршрутизаторів, комп'ютера або сукупності комп'ютерів, сукупності маршрутизаторів і комп'ютерів, та врешті-решт це може бути спеціалізований пристрій. Аналогічно існують різноманітні варіанти реалізації програмної складової МЕ.

## 5.2 Міжмережеві екрани

Міжмережевий екран (МЕ) **(**Firewall) – це система або група систем, які реалізують політику керування доступом між мережами. Будь-який МЕ має такі властивості: здійснює опір атакам; є транзитною точкою між мережами; реалізовує політику керування доступом.

У 1988 році компанія DEC створила перший МЕ, який фактично був пакетним фільтром. Пакетні фільтри належать до, так званих, МЕ без урахування стану (stateless firewall), оскільки в цьому випадку не аналізується, чи є пакет складовою частиною певного потоку (наприклад, TCP з'єднання). Кожний пакет обробляється окремо незалежно від інших.

У 1989 році AT&T Bell Laboratories розробила перший МЕ експертного рівня (stateful firewall), який може визначати, чи належить той чи інший пакет певному потоку даних. Статичні правила, що є характерними для пакетних фільтрів, замінюються на динамічні, що утворюються в режимі реального часу. МЕ експертного рівня допомагає запобігти DoS атакам.

На першому етапі МЕ не були окремими спеціальними пристроями, а реалізовувались програмно. Сьогодні МЕ реалізовуються як програмно, так і апаратно. Виділений МЕ дозволяє вивільнити пам'ять та ресурси маршрутизатора від задач, пов'язаних із фільтрацією трафіку. Сучасні маршрутизатори, зокрема, Cisco Integrated Services Routers (ISRs), також можуть використовуватись як інтелектуальні МЕ експертного рівня в організаціях, що не мають можливості придбати окремий пристрій.

До переваг використання МЕ можна віднести:

1) Запобігання доступу до критичних вузлів та програм нелегітимними користувачами.
2) Блокування небезпечного програмного забезпечення.
3) Реалізація політики безпеки може бути простою, стійкою й масштабованою у разі вірно сконфігурованого МЕ.
4) Організація доступу до мережі через обмежену кількість входів із застосуванням МЕ може спростити реалізацію захисту мережі.

Використання МЕ може також створювати певні проблеми:

1) Помилки у конфігурації МЕ можуть створювати проблеми доступу до мережі, або «дірки» у системі безпеки.
2) Багато програм не можуть коректно працювати через МЕ.
3) Користувачі можуть шукати шляхи обходження МЕ, що погіршує захищеність мережі.
4) Зменшується продуктивність мережі.
5) Неавторизований трафік може тунелюватись або маскуватись під легітимний.

Виділяють такі різновиди МЕ:

- пакетні фільтри – аналізують інформацію 3-го та 4-го рівнів моделі OSI, загалом є статичними, реалізовуються, переважно, у вигляді списків керування доступом (ACL);
- МЕ експертного рівня (Stateful) – відслідковує стан з'єднання;
- МЕ рівня додатків, або проксі (proxy) сервер – аналізує інформацію 3-го, 4-го, 5-го, 7-го рівнів моделі OSI, переважно, реалізовується програмно;
- МЕ трансляції адрес (NAT) – додаткова можливість захисту мережі при застосуванні технологій трансляції адрес;
- МЕ на основі комп'ютера (host-based) – це серверний або персональний комп'ютер з програмною реалізацією МЕ;
- прозорий МЕ – не здійснює процедуру маршрутизації та не вносить зміни в транзитний трафік (якщо він пропускається);
- гібридний МЕ – комбінує функції різних типів.

МЕ експертного рівня (Stateful firewall) є найбільш універсальним. Він використовує інформацію про встановлене з'єднання. Як правило, працює на 3-му та 4-му рівнях моделі OSI, хоча в деяких випадках аналізується також інформація вищих рівнів. Використовує таблицю стану для відслідковування актуальних комунікаційних процесів, наприклад, у TCP-сегменті аналізує службові біти для визначення стану з'єднання.

Як тільки здійснюється доступ до зовнішнього сервісу, МЕ експертного рівня фіксує певну інформацію із запиту, зберігаючи інформацію про стан запиту в таблиці станів. Коли зовнішня система відсилає відповідь на запит, МЕ порівнює пакет зі збереженим станом і дозволяє або забороняє доступ до мережі.

Таблиця стану містить інформацію про адреси та порти відправника і отримувача, інформацію, що впорядковує TCP-сегменти та додаткову інформацію, специфічну для кожного сеансу. Ця інформація створює об'єкт зв'язку, який використовується МЕ для порівняння з пакетами, що надходять у межах сесії. Дозволяється проходження даних лише тоді, якщо вони передаються в межах існуючого з'єднання.

Більш «розумні» експертні МЕ охоплюють можливість розпізнавання команд FTP протоколу, дозволяючи останньому працювати прозоро через МЕ. Також «розумний» МЕ може аналізувати номер послідовності в TCP-сегменті, відповідність DNS-запитів та відповідей; ця опція зменшує небезпеку загрози TCP RST flood атаки та некоректне заповнення DNS-кеша.

Недоліком МЕ експертного рівня є те, що адреси внутрішньої мережі передаються в пакетах, що йдуть назовні; ці недоліки усунені при застосуванні NAT та проксі-серверів.

Cisco Systems надає кілька інструментів для спеціалістів з безпеки: МЕ на основі Cisco IOS дозволяє налаштувати фільтрацію трафіка на маршрутизаторі та у вигляді спеціалізованого пристрою – Adaptive Security Appliances (ASA). МЕ на основі Cisco IOS – це спеціальні опції в операційній системі, що застосовуються в невеликих організаціях та організаціях середнього рівня. Cisco ASA – багатофункціональний пристрій, який реалізує багаторівневий захист для різного типу трафіка і може використовуватись у мережах різного рівня. Вибір на користь того чи іншого рішення здійснюється на основі порівняння вартості й ризику.

У технологіях захисту мереж часто використовують термін «демілітаризована зона» (DMZ) – це частина мережі, доступ до якої обмежено МЕ як із внутрішньої мережі організації, так і зовні. DMZ визначає частини мережі, яким можна довіряти, а яким – ні.

Найпростіший варіант реалізації – це МЕ на кордоні приватної та публічної мережі (рис. 5.1, а). Як правило, вихідний трафік пропускається майже увесь, а вхідний, здебільшого, блокується. Пропускається тільки трафік, що передається зовні в межах сеансів, ініційованих із середини.

Часто використовується архітектура МЕ з трьома інтерфейсами – один у приватну мережу, другий – у публічну, третій – у DMZ (рис. 5.1, б). Трафік із внутрішньої мережі вільно пропускається як у зовнішню мережу, так і в DMZ. Трафік з DMZ вільно пропускається у внутрішню мережу, але блокується у зовнішню. Трафік зовні, як правило, блокується повністю, за винятком, рефлексивного, ініційованого або з приватної мережі, або з DMZ. Однак при застосуванні DMZ певні типи трафіка, ініційовані зовні, пропускаються в DMZ-зону, наприклад, DNS, HTTP або HTTPS. Під час застосування багаторівневого сценарію захисту МЕ забезпечують захист периметра всієї мережі та найважливіших сегментів внутрішньої мережі. Наприклад, захист фінансової мережі від інших сегментів внутрішньої мережі.

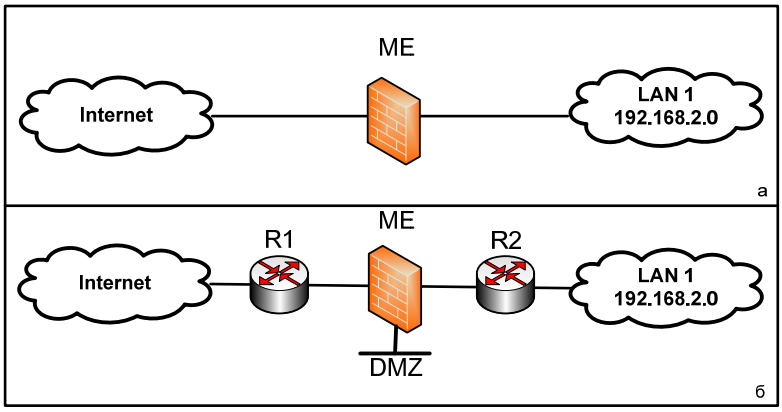


Рисунок 5.1 – Реалізації МЕ
на кордоні приватної та публічної мереж

Для забезпечення ефективного захисту варто дотримуватись базових рекомендацій:

1. МЕ мають розташовуватись на найкритичніших, з погляду безпеки, кордонах мережі.
2. МЕ є основними пристроями захисту мережі, але не є універсальним засобом захисту.
3. На МЕ слід блокувати увесь трафік за замовчуванням, а дозволяти тільки потрібні сервіси.
4. Варто жорстко контролювати фізичний доступ до МЕ.
5. Необхідно регулярно переглядати та аналізувати системні повідомлення (лог-файли), що генеруються МЕ.
6. Варто уважно ставитись до змін, що робляться в конфігурації МЕ.
7. МЕ, переважно, захищає від технічних атак зовні, внутрішні атаки можуть мати нетехнічну природу.

### 5.2.1 МЕ керування доступом на основі контексту

Керування доступом на основі контексту (Context-based access control (CBAC) – це функція, яка є доступною в МЕ на основі Cisco IOS. CBAC інтелектуально фільтрує TCP та UDP-трафік, використовуючи інформацію рівня додатків (7-й рівень моделі OSI). Він забезпечує експертну фільтрацію на програмному рівні, ураховуючи особливості того чи іншого протоколу рівня додатків. Також можуть бути коректно обслуговувані протоколи, що використовують кілька логічних з'єднань (FTP, H.323). CBAC можуть урахувати особливості, які створюють NAT та PAT, можуть блокувати P2P-з'єднання, а також трафік миттєвих повідомлень. CBAC реалізують чотири основні функції: фільтрацію трафіку, перевірку, детектування зловмисників, генерацію повідомлень аудиту та застережень.

**Фільтрація трафіку.** CBAC може бути сконфігурований, щоб дозволяти проходження тільки рефлективного трафіку аналогічно до МЕ експертного рівня. CBAC аналізує не тільки інформацію мережевого й транспортного рівнів, а й рівня додатків (наприклад, інформацію про FTP-

з’єднання) для визначення стану з’єднання. Це дозволяє обслуговувати протоколи, які використовують кілька з’єднань для своєї роботи.

**Перевірка трафіка.** CBAC перевіряє в пакетах інформацію рівня додатків, а також інформацію TCP та UDP-заголовків, що дозволяє запобігати певному типу атак, зокрема, атакам типу SYN-flooding. Для цього перевіряється поле «номер в послідовності», і якщо воно не задовольняє певний діапазон, пакет відкидається; CBAC може також закривати TCP-з’єднання, установлення яких не було завершено.

**Детектування зловмисника.** CBAC забезпечує обмежений інструментарій для захисту від певних SMTP-атак. Деякі атаки мають певні характеристики або сигнатури. Якщо CBAC визначив факт атаки, він розриває відповідне з’єднання і відправляє повідомлення на сервер системних повідомлень.

**Аудит та застереження.** CBAC дозволяє здійснювати процедури аудиту та генерування застережень (Alert) у зв’язку з небезпечними подіями. Базові можливості CBAC:

- моніторинг встановлення TCP-з’єднання;
- підтримка інформації про UDP-сесію;
- відслідковування послідовності TCP-сегментів;
- інспекція DNS-запитів та відповідей;
- інспекція типів ICMP-повідомлень;
- підтримка програм, що використовують кілька з’єднань;
- перевірка інкапсульованих адрес (при застосуванні тунелів);
- перевірка інформації рівня додатків.

Варто зауважити, що CBAC контролює тільки ті протоколи, для яких здійснені налаштування, інші ж блокуються. Контролюються лише запити, що проходять через ME.

Без застосування CBAC фільтрація трафіку обмежується ACL, який перевіряє пакети на мережевому і, максимум, транспортному рівні. CBAC базується на експертному пакетному фільтрі, специфічному для кожного протоколу рівня додатків. Це означає, що аналізується також інформація рівня додатків, підтримується і аналізується інформація в таблиці станів (або зв’язків) для відслідковування активних сесій.

Таблиця станів відслідковує сесії і перевіряє всі пакети, що проходять через ME. CBAC використовує таблицю станів для створення динамічних записів, що дозволяють проходження рефлективного трафіку через ME. У процесі роботи CBAC створює «отвір» шляхом додавання тимчасових ACL записів для специфічних сесій. Ці «отвори» утворюються, коли специфічний трафік виходить із захищеної мережі в публічну через ME. Таблиця станів динамічно змінюється та адаптується до потоків трафіка.

Припустимо, що користувач ініціює вихідне з’єднання, наприклад, Telnet із захищеної мережі у зовнішню мережу (рис. 5.2) і CBAC дозволяє інспектування Telnet трафіку. Також припустимо, що ACL, розташований на зовнішньому інтерфейсі, забороняє проходженню Telnet трафіку зовні. У такому разі з’єднання проходить багатоетапне оброблення.

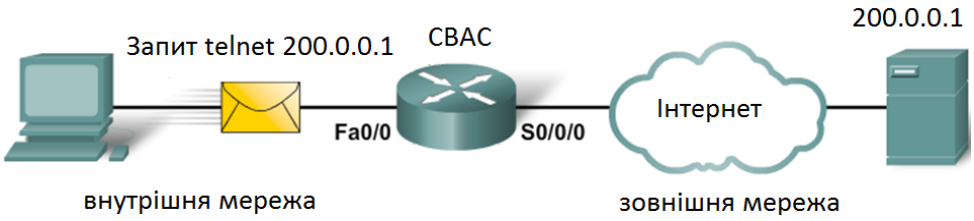


Рисунок 5.2 – З’єднання Telnet із захищеної мережі у зовнішню

1. Якщо на інтерфейс маршрутизатора, через який надходить трафік, налаштовано вхідний ACL, то пакет проходить перевірку. У разі дозволу починається перевірка правил CBAC.
2. Якщо в правилах CBAC нічого не сказано про Telnet трафік, то він передається на зовні й жодна інша інформація не зберігається. В іншому випадку переходим до наступного етапу.
3. Інформація про з’єднання порівнюється із записами в таблиці станів. Якщо такого з’єднання ще немає, воно додається. Якщо воно є, то таймер пасивності скидається.
4. Як тільки новий запис додано, динамічний запис додається на зовнішній інтерфейс для перевірки вхідного трафіку. Він дозволятиме проходження рефлективного трафіку для цього з’єднання. Цей тимчасовий «отвір» залишається відкритим тільки доти, допоки сесія відкрита. Динамічний ACL не зберігається в NVRAM.
5. Коли сесія розривається, динамічна інформація з таблиці станів та динамічний запис видаляються.

Робота CBAC схожа на роботу рефлексивного ACL.

CBAC є гнучким в конфігуруванні, особливо щодо вибору напрямку перевірки. Його можна налаштувати на роботу в обох напрямках.

**Обробка TCP CBAC.** Як тільки TCP-сегмент з бітом SYN потрапляє на маршрутизатор і пропускається вхідним ACL, створюється динамічний запис про нову сесію. Сесія описується адресами й портами кінцевих вузлів, а також номером в послідовності і службовими бітами. Усі подальші пакети, що належать до цієї сесії, перевіряються відповідно до поточного стану і відкидаються, якщо вони невірні. Перевірка здійснюється шляхом аналізу номера в послідовності, кількості байтів, що передаються, та значення службових бітів, відповідно до правил роботи TCP-з’єднання. Після встановлення з’єднання всі сегменти обов’язково містять біт ACK.

**Обробка UDP CBAC.** Обробка UDP базується на аналізі тільки адрес і портів кінцевих станцій. Якщо таймер пасивності перебільшує певне значення, інформація про з’єднання видаляється.

**Обробка тунельних протоколів.** Експертні ME, як правило, не підтримують обробку тунельних протоколів, наприклад, GRE та IPsec і обробляють їх як звичайні пакетні фільтри. Якщо експертний ME налаштовано

на обробку того чи іншого тунельного протоколу, то він робить це аналогічно до обробки UDP. Багато протоколів, зокрема, FTP та SQLnet використовують кілька логічних з'єднань під час своєї роботи. Експертний ME, знаючи про таку особливість, після відправки запиту на встановлення з'єднання «підглядає» процедуру встановлення додаткового з'єднання і вносить ці дані також у таблицю станів. CBAC використовує набір правил для перевірки. Правило перевірки асоціюється з певним інтерфейсом і певним напрямком.

CBAC ME може розпізнавати команди, специфічні для певного протоколу, наприклад, невірні SMTP-команди в каналі керування, а також виявляти та запобігати певним атакам програмного рівня. Якщо ME визначив факт атаки, він може виконати такі дії:

1. Генерувати повідомлення застереження (Alert).
2. Захистити системні ресурси, які виснажуються під час атаки.
3. Блокувати пакети, що утворюють атаку.

Значення timeout та threshold використовуються для керування інформацією про стан TCP-з'єднання. Вони дозволяють розривати незавершені з'єднання. ME CBAC використовує 3 порогових значення для запобігання DoS-атакам типу TCP-flooding: загальна кількість напіввідкритих TCP-сесій; кількість напіввідкритих TCP-сесій за одиницю часу; кількість напіввідкритих TCP-сесій на хост. Якщо перебільшено загальну кількість напіввідкритих TCP-сесій, ME може виконати такі дії: відправити повідомлення reset на кінцеву точку з найстарішою напіввідкритою сесією для звільнення ресурсів для нових з'єднань; блокувати всі SYN-пакети протягом часу, визначеного таймаутом для незавершених сесій. Протягом цього часу нові з'єднання не зможуть встановлюватись.

**Конфігурування CBAC:**

1. Вибрати інтерфейси – внутрішній і зовнішній.
2. Сконфігурувати ACL на інтерфейсі.
3. Визначити правила перевірки.
4. Призначити правила на інтерфейси.

**Вибір інтерфейсів.** У випадку двох інтерфейсів внутрішнім інтерфейсом вважається той, на якому буде ініціюватись сесія. У випадку трьох інтерфейсів ME має дозволяти зовнішній трафік до ресурсів в DMZ таких, як: DNS та веб-сервери. Той же ME може перешкоджати певному трафіку, що входить у внутрішню мережу, якщо він не є рефлективним в межах встановлених сесій. CBAC може бути сконфігурований у двох напрямках на одному або більше інтерфейсів.

**Конфігурування ACL на інтерфейсі.** Для забезпечення ефективного захисту необхідно дотримуватись таких рекомендацій:

- варто налаштувати базову конфігурацію, яка дозволяє проходження всього трафіку з внутрішньої мережі в зовнішню;

- дозвольте трафік, який МЕ буде інспектувати; він має бути дозволений на інтерфейсах, на які він буде надходити;
- використовуйте розширені ACL для фільтрації трафіку, що надходить із зовнішньої мережі. Бажано призначати ACL, що аналізують вхідний трафік (in), оскільки в такому разі менше буде завантажений маршрутизатор;
- установіть захист від підміни адрес (antispoofing) шляхом заборони будь-якого вхідного трафіку, що надходить на зовнішній інтерфейс з адрес, які збігаються з адресами внутрішньої мережі;
- забороніть широкомовні повідомлення з адресами відправника 255.255.255.255;
- хоча за замовчуванням припускається в кінці ACL наявність правила заборони всього, що не дозволено (deny any any), краще це правило задати явно, що дасть змогу контролювати кількість пакетів, відкинутих ACL.

**Визначення правил перевірки.** Адміністратор має виокремити правила для перевірки, що визначатимуть тип протоколу програмного рівня, що перевіряється. Зазвичай, достатньо одного правила. Винятком є випадок, якщо МЕ треба налаштувати для роботи в обох напрямках через один інтерфейс. У такому разі створюють два правила – по одному на кожний напрямок. Правила для перевірки мають специфікувати кожний дозволений протокол програмного рівня або просто TCP, UDP, або ICMP. У другому випадку інспекції підлягають усі активні сеанси відповідних протоколів. ICMP-інспекція дозволяє пропускати echo-відповіді на echo-запити із внутрішньої мережі.

Правило інспекції містись набір рядків, кожний з яких містить назву протоколу, ім’я правила, а також опції відправки застережень та записів в журнал аудиту:

```
Router(config)# ip inspect name inspection_name protocol
[alert {on | off}] [audit-trail {on | off}] [timeout seconds]
```

Перелік найбільш популярних протоколів, що можуть інспектуватись, наведено в табл. 5.1.

Приклад:

```
Router(config)# ip inspect name RULE1 http alert on audit-
trail on timeout 200
Router(config)# ip inspect name RULE1 ftp alert on audit-
trail on timeout 200
```

У цьому прикладі інспектуються протоколи http та ftp, попередження та аудит увімкнено, timeout становить 200 секунд. Останній крок конфігурування CBAC – призначення інспекційного правила на інтерфейс:

```
Router(config-if)# ip inspect inspection_name {in | out}
```

Таблиця 5.1 – Популярні протоколи, що можуть інспектуватись

| **Ключове Слово** | **Протокол** |
|---|---|
| icmp | Internet Control Massage Protocol |
| ftp | File Transfer Protocol |
| h323 | H.323 Protocol (for example Microsoft NetMeeting or Intel Video Phone) |
| http | HTTP Protocol |
| dns | DNS Protocol |
| realaudio | Real Audio Protocol |
| rpc | Remote Procedure Call Protocol |
| smtp | Simple Mail Transfer Protocol |
| sqlnet | SQL Net Protocol |
| streamworks | StreamWorks Protocol |
| tcp | Transmission Control Protocol |
| tftp | TFTP Protocol |
| udp | User Datagram Protocol |
| telnet | telnet Protocol |

Правила інспектування та застосування ACL:

1. На інтерфейс, куди надходить трафік, який буде інспектуватись, призначається ACL, що дозволяє бажаний трафік та призначаються інспекційні правила в тому ж напрямку.
2. На всіх інших інтерфейсах призначаються ACL на вхідний трафік, які блокують увесь трафік, за винятком трафіку, що не інспектується ME, наприклад, GRE та ICMP.

Наприклад, у мережі, що показана на рис. 5.3, адміністратору треба дозволити внутрішнім користувачам ініціювати TCP, UDP та ICMP з усіма зовнішніми хостами. Зовнішнім клієнтам дозволяється доступ до SMTP-сервера з адресою 194.146.141.3 та HTTP-сервера з адресою 194.146.141.4, які розташовані в DMZ. Також необхідно дозволити певні ICMP-повідомлення на всі інтерфейси. Увесь інший зовнішній трафік має блокуватись.

ACL, що дозволяє проходження внутрішнього трафіку назовні:

```
R1(config)# access-list 101 permit tcp 10.10.10.0 0.0.0.255
any
R1(config)# access-list 101 permit udp 10.10.10.0 0.0.0.255
any
R1(config)# access-list 101 permit icmp 10.10.10.0 0.0.0.255
any
R1(config)# access-list 101 deny ip any any
```

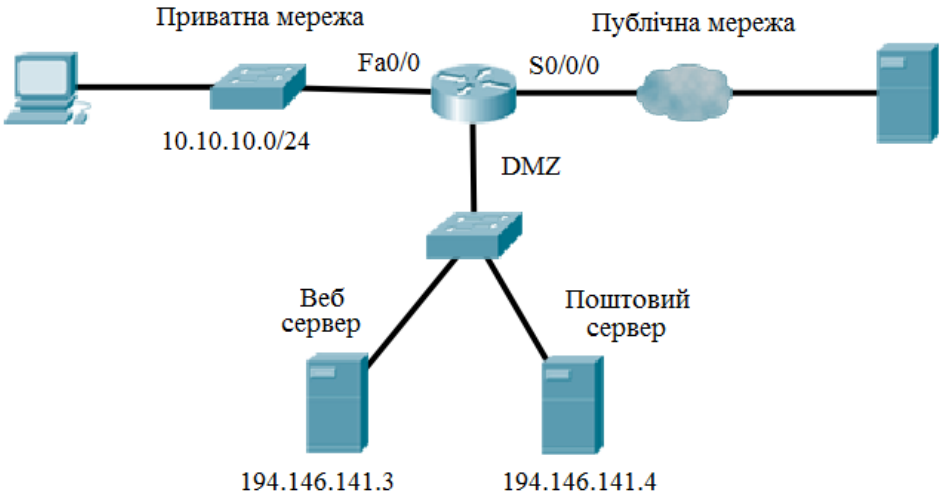


Рисунок 5.3 – Схема мережі

Призначаємо створений список на внутрішній інтерфейс:

```
R1(config)# interface Fa0/0
R1(config-if)# ip access-group 101 in
```

Створюємо розширений ACL для дозволу проходження SMTP, HTTP та службового ICMP-трафіку в DMZ зону та блокування іншого трафіку:

```
R1(config)# access-list 102 permit tcp any host 194.146.141.3
eq www
R1(config)# access-list 102 permit tcp any host 194.146.141.4
eq smtp
R1(config)# access-list 102 permit icmp any any echo-reply
R1(config)# access-list 102 permit icmp any any unreachable
R1(config)#   access-list   102   permit   icmp   any   any
administratively-prohibited
R1(config)# access-list 102 permit icmp any any packet-too-
big
R1(config)# access-list 102 permit icmp any any echo
R1(config)# access-list 102 permit icmp any any time-exceeded
R1(config)# access-list 102 deny ip any any
```

Цей ACL призначаємо на інтерфейс, що дивиться назовні:

```
R1(config)# interface S0/0/0
R1(config-if)# ip access-group 102 in
```

Створюємо правила для інспектування TCP та UDP-трафіку:

```
R1(config)# ip inspect name RULE tcp
R1(config)# ip inspect name RULE udp
```

Ці правила призначаємо на внутрішній інтерфейс:

```
R1(config)# interface Fa0/0
R1(config-if)# ip inspect RULE in
```

Інспекційний лист автоматично створить тимчасовий ACL, призначений до зовнішнього інтерфейсу для TCP та UDP-зв’язків. Це дозволить проходження рефлективного трафіку. Для видалення CBAC з маршрутизатора використовується команда: `Router(config)# no ip inspect`. Ця команда видаляє всі CBAC команди, таблицю станів, усі тимчасові ACL, скидає всі порогові значення й тайм-аути до значень за замовчуванням.

**Моніторинг роботи МЕ CBAC.** CBAC-інспектування підтримує два типи звітів: попередження та аудит.

**Попередження** (Alerts) виводять повідомлення, що стосуються функціонування CBAC, наприклад, інформацію про недостатню кількість ресурсів, DoS-атаки та інші загрози. Попередження за замовчування увімкнені й автоматично виводяться на консоль адміністратора. Їх можна вимкнути, хоча вкрай небажано: `Router(config)#ip inspect alert-off`.

Існує можливість вимкнути попередження для окремих правил, хоча це теж не рекомендують робити.

**Аудит** дозволяє реєструвати інформацію про зв'язки, що CBAC інспектує, зокрема, вдалі та невдалі спроби. Виводить повідомлення, коли CBAC додає або видаляє записи з таблиці станів. За замовчуванням аудит вимкнено. Для увімкнення команда: `Router(config)# ip inspect audit-trail`. За замовчуванням і попередження, і аудит виводять інформацію на консоль. Можна сконфігурувати виведення на Syslog-сервер:

```
Router(config)# logging on
Router(config)# logging host 10.0.0.3
Router(config)# ip inspect audit-trail
Router(config)# no ip inspect alert-off
```

Для перегляду поточної інформації про роботу МЕ використовується команда `show ip inspect` з параметрами, що вказані в табл. 5.2.

Таблиця 5.2 – Параметри команди `show ip inspect`

| Параметр | Пояснення |
|---|---|
| Name | Видає інформацію про конкретний набір правил (за іменем) |
| Config | Видає повну CBAC-інспекцію конфігурації на роутері |
| Interfaces | Видає правила інспекції, що активовані на інтерфейсі роутера |
| Sesions | Видає перелік з'єднань у таблиці стану CBAC |
| Sessions (detail) | Видає деталі з'єднань у таблиці стану CBAC |
| all | Видає дані всіх опцій, перелічених у таблиці |

### 5.2.2 МЕ, що базується на зонах

Більш гнучка модель налаштування політик безпеки реалізована в МЕ, що базується на понятті зони (**zone-based firewall**, ZBF). Відповідно до цієї моделі, інтерфейси асоціюються з певними зонами, а потім інспекційна політика прикладається до трафіку, що передається між зонами. ZBF дозволяє різні інспекційні політики призначати до комп'ютерних груп, під'єднаних до одного інтерфейсу. Це також дає можливість заборонити трафік за допомогою політик за замовчуванням (default) – політики блокування всього трафіка (deny-all) між зонами МЕ. Політики інспектування, реалізовані в ZBF, підтримують усі попередньо визначені функції МЕ, зокрема, інспекцію TCP, UDP-протоколів, інспекцію на рівні додатків, URL-фільтрацію, запобігання DoS-атакам тощо. Застосування ZBF спрощує механізм захисту й робить його структурно-орієнтованим. Використання CBAC МЕ є досить складним через відсут-

ність можливості застосування ієрархічних структур. CBAC має такі основні обмеження:

- велика кількість інспекційних політик та ACL на кількох інтерфейсах маршрутизатора досить сильно ускладнює процедуру визначення кореляції політик для аналізу трафіку між кількома інтерфейсами;
- політики не можуть бути прив'язані до груп хостів або підмереж за допомогою ACL, увесь трафік, що проходить через певний інтерфейс, потрапляє під одну політику інспектування;
- процес налаштування та функціонування CBAC фактично прив'язаний до списків керування доступом.

При застосуванні ZBF-зони встановлюють кордони безпеки мережі. Зони самостійно визначають кордони, де трафік потрапляє під обмеження політики, коли проходить в інший регіон мережі. За замовчуванням політика між зонами – усе блокувати. Якщо жодна політика не сконфігурована між зонами, увесь трафік буде блокуватись. Це суттєва відмінність від CBAC, де трафік за замовчуванням пропускається доти, допоки він не буде заблокований ACL. Основні переваги ZBF: не залежать від ACL; за замовчуванням здійснюється блокування всього трафіка; одна політика впливає на будь-який трафік, замість використання багатьох ACL й інспекційних дій.

Варто звернути увагу, що обидва підходи можуть застосовуватись спільно, однак дві моделі не можуть одночасно використовуватись на одному інтерфейсі, наприклад: інтерфейс не може бути сконфігурований як член зони і для IP-інспекції одночасно. Етапи проектування зонних політик:

1. Визначення зон. Уся мережа поділяється на зони з різними рівнями безпеки. На цьому етапі не здійснюється вибір обладнання, кількість пристроїв тощо.
2. Визначення політик для зон – для кожної пари «відправник-отримувач» визначаються сесії, які клієнти в зоні відправника можуть запитати в зоні отримувача, наприклад, для TCP та UDP або ICMP-трафіка. Цей етап не передбачає жодних фізичних дій.
3. Проектування фізичної інфраструктури з урахуванням вимог захищеності та доступності. Він визначає кількість пристроїв між найбільш захищеними і найменш захищеними зонами та визначає надлишкові пристрої.
4. Ідентифікація груп зон та об'єднання вимог до трафіку. Адміністратор має ідентифікувати підмножини зон, під'єднаних до його інтерфейсів й об'єднати вимоги до трафіку для цих зон.

ZBF ME може бути сконфігурований на виконання трьох дій:

- Інспектування (Inspect) – дія, еквівалентна до команди ip inspect в CBAC ME. У цьому випадку автоматично дозволяється проходження рефлективного трафіку. Також коректно обслуговуються протоколи, що використовують кілька TCP-з'єднань, наприклад, FTP.
- Блокування (Drop) – аналог команди deny в ACL, блокує увесь трафік.
- Дозвіл (Pass) – аналог команди permit в ACL. Ця дія не відслідковує стан з'єднання або сесії. Дозволяється проходження трафіку тільки в одному напрямку. У цьому випадку відповідна політика має бути застосована для того, щоб дозволити проходження трафіку у зворотному напрямку.

При конфігуруванні членства інтерфейсу маршрутизатора в тій чи іншій зоні необхідно враховувати такі правила:

- зона має бути створена перед тим, як адміністратор зможе асоціювати з нею інтерфейси;
- якщо в передаванні трафіку беруть участь усі інтерфейси маршрутизатора, кожний інтерфейс має бути членом однієї із зон;
- інтерфейс може бути асоційованим тільки з однією зоною;
- за замочуванням трафіку дозволяється передаватись між інтерфейсами, що є членами однієї зони;
- для того, щоб дозволити проходження трафіку в зону або із зони, членом якої є інтерфейс, має бути сконфігурована політика дозволу або інспекції трафіку між цією зоною та іншою зоною;
- трафік не може передаватись між інтерфейсом, що є членом зони й іншим інтерфейсом, що не входить до жодної із зон. Дії (інспектування, блокування та дозвіл) можуть призначатись тільки між зонами;
- інтерфейси, які не призначені в зону, можуть використовувати CBAC-інспекцію;
- якщо є необхідність, щоб інтерфейс маршрутизатора не брав участі в зонній політиці й пропускав увесь трафік, цей інтерфейс можна додати в зону та сконфігурувати політику, що дозволятиме проходження всього трафіку між цією зоною й будь-якою іншою зоною.

Загалом правила обробки трафіку під час застосування ZBF ME визначаються зідно з табл. 5.3.

Таблиця 5.3 – Правила обробки трафіку під час застосування ZBF ME

| Інтерфейс відправника є членом зони? | Інтерфейс одержувача є членом зони? | Чи існує зонна пара? | Чи існує політика між зонами? | Результат |
|---|---|---|---|---|
| Ні | Ні | - | - | Увесь трафік пропускається |
| Так (зона 1) | Так (зона 1) | - | - | Увесь трафік пропускається |
| Так | Ні | - | - | Увесь трафік блокується |
| Ні | Так | - | - | Увесь трафік блокується |
| Так (зона 1) | Так (зона 2) | Ні | - | Увесь трафік блокується |
| Так (зона 1) | Так (зона 2) | Так | Ні | Увесь трафік блокується |
| Так (зона 1) | Так (зона 2) | Так | Так | Дія визначається політикою |

Коли інтерфейс сконфігурований як член зони, хости, що під'єднані до нього, також входять у зону. Але трафік між інтерфейсом маршрутизатора і хостами не контролюється зонною політикою. Усі інтерфейси маршрутизатора автоматично є членами, так званої, власної зони маршрутизатора (self zone). Тому для обмеження трафіку, що спрямовується на IP-адреси інтерфейсів маршрутизатора з різних зон мають бути застосовані відповідні політики. Якщо політик між певною зоною й власною зоною немає, дозволяється проходження усього трафіку на інтерфейс без інспектування. Правила обробки трафіка у такому разі визначаються згідно з табл. 5.4.

Таблиця 5.4 – Правила обробки трафіку

| Інтерфейс відправника є членом зони? | Інтерфейс одержувача є членом зони? | Чи існує зонна пара? | Чи існує політика між зонами? | Результат |
|---|---|---|---|---|
| Маршрутизатор | Так | Ні | - | Увесь трафік пропускається |
| Маршрутизатор | Так | Так | Ні | Увесь трафік пропускається |
| Маршрутизатор | Так | Так | Так | Дія визначається політикою |
| Так | Маршрутизатор | Ні | - | Увесь трафік пропускається |
| Так | Маршрутизатор | Так | Ні | Увесь трафік пропускається |
| Так | Маршрутизатор | Так | Так | Дія визначається політикою |

**Конфігурування ZBF**

Для конфігурування ZBF ME необхідно:

1. Створити зони за допомогою команди zone security.
2. Визначити класи трафіка за допомогою команди class-map type inspect.
3. Визначити політики за допомогою команди policy-map type inspect.
4. Призначити політики до пар зон відправника та отримувача використовуючи команду zone-pair security.
5. Асоціювати інтерфейси маршрутизатора із зонами, використовуючи команду zone-member security interface.

Під час конфігурування ZPF потрібно враховувати кілька факторів:

- Зона має бути сконфігурована, перш ніж вона буде асоціюватись з інтерфейсом.
- Інтерфейс не може належати до кількох зон.
- ZBF може співіснувати з CBAC. Команда ip inspect може використовуватись тільки на інтерфейсах, які не є членами зон безпеки.
- Трафік ніколи не буде передаватись між інтерфейсами, асоційованими із зоною та інтерфейсами, що не асоційовані із зонами.
- За замовчуванням міжзонна політика блокує увесь трафік доти, допоки не буде явно визначена політика між зонами.
- Маршрутизатор ніколи не фільтрує трафік між інтерфейсами всередині зони.
- Членство у зонах не захищає сам маршрутизатор, оскільки трафік до та від маршрутизатора не фільтрується доти, допоки не буде створено пари, що утворюються власною зоною (self-zone).

**Створення зон:**

```
Router(config)# zone security zone-name
Router(config-sec-zone)# description line-of-description
```

Під час створення зони потрібно зрозуміти, які інтерфейси будуть до неї входити. Як правило, до однієї зони входять інтерфейси, що мають спільні вимоги, з погляду безпеки.

**Визначення класів трафіку**

Класи трафіку дозволяють визначати потоки даних, щоб далі оперувати з ними як з єдиним цілим:

```
Router(config)# class-map type inspect [match-any | match-all] class-map-name
```

Для 3-го та 4-го рівня match-any визначає поведінку за замовчуванням:

```
Router(config)# class-map type inspect protocol-name [match-any | match-all] class-map-name
```

Для 7-го рівня використовується class maps специфічна для того чи іншого протоколу рівня додатків.

Для додавання трафіку в карту може використовуватись ACL за допомогою команди:

```
Router(config-cmap)# match access-group {access-group | name
access-group-name}
```

Для додавання протоколу:

```
Router(config-cmap)# match protocol protocol-name
```

**Визначення політик.** Політики визначають, що робити з тим чи іншим класом трафіку (відкидати, пропускати або інспектувати):

```
Router(config)# policy-map type inspect policy-map-name
```

Класи трафіку, на які буде розповсюджуватись дія, визначаються за допомогою `class type inspect`

```
Router(config-pmap)# class type inspect class-name
```

На подальшому етапі вказується дія для певного класу трафіку:

```
Router(config-pmap-c)# pass | inspect | drop [log]| police
```

**Застосування політик.** Після створення політик вони призначаються для трафіку, що передається між парою зон. Для призначення політики спочатку треба створити зонну пару. Далі треба визначити зону відправника, зону отримувача та політику, що керує трафіком між ними:

```
Router(config)# zone-pair security zone-pair-name [source
source-zone-name | self] destination [self | destination-
zone-name]
```

Команда `service-policy type inspect policy-map-name` використовується для приєднання карти політики до зонної пари. Інспекція на 7-му рівні здійснюється так:

```
Router(config-pmap-c)# service-policy {h323 | http | im |
imap | p2p | pop3 | sip | smtp | sunrpc | urlfilter} policy-
map
```

**Асоціювання інтерфейсів з відповідною зоною.** Останній етап налаштування – це прив'язування інтерфейсу до певної зони:

```
Router(config-if)# zone-member security zone-name
```

Розглянемо приклад налаштування ZBF для схеми, що наведено на рис. 5.4.

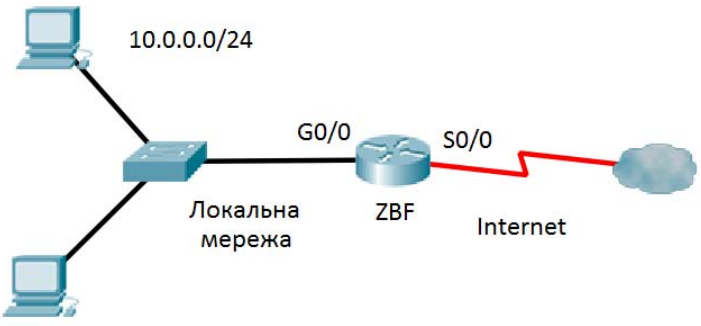


Рисунок 5.4 – Схема налаштування ZBF

1. Створюємо зони:
```
ZBF(config)# zone security Internal
ZBF(config-sec-zone)# description Internal network
ZBF(config)# zone security External
ZBF(config-sec-zone)# description External network
```
2. Визначаємо клас трафіка за допомогою розширеного ACL. У такому разі це буде увесь трафік, що передається з локальної мережі:
```
ZBF(config)# class-map type inspect USERS_TRAFFIC
ZBF(config-cmap)# match access-group 101
ZBF(config-cmap)# exit
ZBF(config)# access-list 101 permit ip 10.0.0.0 0.0.0.255 any
```
3. Визначаємо політику, що буде застосовуватись до вибраного класу трафіку. У такому разі це буде інспектування:
```
ZBF(config)# policy-map type inspect Internal-To-External
ZBF(config-pmap)# class type inspect USERS_TRAFFIC
ZBF(config-pmap-c)# inspect
```
4. Призначаємо політику до пари зон відправника та отримувача:
```
ZBF(config)# zone-pair security Internal-External source
Internal destination External
ZBF(config-sec-zone-pair)# description Internet Access
ZBF(config-sec-zone-pair)# service-policy type inspect
Internal-To-External
```
5. Асоціюємо інтерфейси маршрутизатора із зонами:
```
ZBF(config)# interface G0/0
ZBF(config-if)# zone-member security Internal
ZBF(config-if)# interface S0/0
ZBF(config-if)# zone-member security External
```

### 5.3 Проксі-сервери

Мережеві екрани прикладного рівня називають проксі-серверами. **Проксі-сервер – це програма, яка виконує функції посередника між клієнтськими й серверними частинами розподілених мережевих програм**, причому припускається, що клієнти належать до внутрішньої (захищеної) мережі, а сервери – до зовнішньої (потенційно небезпечної) мережі. Роль транзитного вузла дозволяє проксі-серверу логічно розірвати пряме з'єднання між клієнтом і сервером з метою контролю за процесом обміну повідомленнями між ними.

Очевидно, що проксі-сервер може ефективно виконувати свої функції тільки за умови відсутності інших точок доступу до мережі. Коли клієнту необхідно отримати дані, наприклад, файл або поштове повідомлення з певного сервера, він відсилає свій запит проксі-серверу, що аналізує цей запит на основі визначених адміністратором правил і вирішує, яким чином він має бути оброблений. Запит може бути відкинутий, переданий без змін відповідному серверу, модифікований перед передачею тощо.

Правила, якими керується проксі-сервер, можуть бути сформульовані у вигляді умов пакетної фільтрації, наприклад, у робочі години забо-

роняється доступ до певних ресурсів Інтернет, зокрема, соціальних мереж, при роботі з FTP-серверами дозволяється тільки запис на сервер, а зчитування із сервера заборонено тощо. Проксі-сервери також можуть фільтрувати поштові повідомлення за типами файлів, наприклад, заборонити отримання повідомлень з файлами формату *.mp3, а також аналізувати контент листів. До різних користувачів можуть застосовуватись різні правила фільтрації, тому часто на проксі-сервери покладається завдання аутентифікації користувачів. Якщо проксі-сервер після перевірки правил робить висновок, що запит задовольняє умовам проходження у зовнішню мережу, то він виконує за дорученням клієнтської програми, але від свого імені, процедуру з'єднання з відповідним сервером. У деяких випадках проксі-сервер може змінювати запит клієнта. Наприклад, якщо в нього вбудована функція трансляції мережевих адрес, він може підмінювати в пакеті запиту IP-адреси і/або номери TCP і UDP-портів відправника. За допомогою такого підходу проксі-сервер позбавляє зловмисника можливості сканувати внутрішню мережу для отримання інформації про адреси вузлів і структуру мережі. Єдина адреса в такому разі, що може стати відомою зловмиснику, – це адреса комп'ютера, на якому працює програма проксі-сервера. Тому багато атак, що базуються на знанні зловмисником адрес вузлів внутрішньої мережі, стають неможливими. Проксі-сервер, що є посередником між клієнтом і сервером, не може не враховувати специфіку протоколу їхньої взаємодії. Для кожного з протоколів SMTP/POP, HTTP, HTTPS, telnet, FTP існує власний проксі-сервер, орієнтований на використання відповідними програмами: електронною поштою, веб-браузером, клієнтом telnet або FTP-клієнтом. Кожен посередник приймає і обробляє пакети тільки того типу програм, для обслуговування якого він був створений.

За принципом функціонування виділяють проксі-сервери прикладного рівня та рівня з'єднань.

**Проксі-сервер прикладного рівня** «вклинюється» у процедуру взаємодії клієнта й сервера за одним з протоколів верхнього рівня, наприклад, HTTP. Щоб виконувати функцію посередника на програмному рівні, проксі-сервер має «розуміти» сенс команд, «знати» формати й послідовність повідомлень, якими обмінюються клієнт і сервер відповідної служби. Це дає можливість проксі-серверу проводити аналіз вмісту повідомлень, робити висновки про підозрілий характер того чи іншого сеансу.

**Проксі-сервер рівня з'єднань** працює на транспортному рівні, контролюючи TCP-з'єднання є менш «інтелектуальним» і, як наслідок, має менше можливостей для виявлення і запобігання атакам. Проте він є більш універсальним, тобто його можна використовувати у будь-яких програмах, що працюють за протоколом TCP або UDP.

## 5.4 Системи виявлення та запобігання вторгнень

**Система виявлення вторгнень**, СВВ, (Intrusion Detection System (IDS)) – програмний або апаратний засіб, який призначено для виявлення фактів неавторизованого доступу (вторгнення або мережевої атаки) у комп'ютерну систему або мережу. СВВ є одним з перших засобів мережевої безпеки; система вперше була розроблена фірмою SRI International у 1984 році.

СВВ дозволяє в режимі реального часу виявляти певні типи атак й інформувати про це адміністратора. Використання СВВ дозволяє досягти таких цілей:

- виявити факт вторгнення або мережевої атаки;
- спрогнозувати можливі майбутні атаки й виявити вразливості для запобігання їхнього подальшого розвитку;
- виконати документування існуючих загроз;
- забезпечити контроль якості адміністрування, з погляду безпеки;
- отримати корисну інформацію щодо проникненнь, які були, з метою унеможливлення їх у майбутньому;
- визначити розташування джерела атаки відносно локальної мережі.

Архітектура типової СВВ містить такі компоненти:

- сенсорну підсистему, призначену для збирання подій, пов'язаних з безпекою мережі, що захищається;
- підсистему аналізу, призначену для виявлення мережевих атак та підозрілих дій;
- підсистему, де накопичується інформація про первинні події та результати аналізу;
- підсистему керування, яка дозволяє конфігурувати СВВ, спостерігати за станом системи, що захищається, переглядати виявлені системою інциденти.

Наприкінці 90-х IDS стали заміняти на системи запобігання вторгнень, СЗВ, (Intrusion Prevention System (IPS). Ці системи будуються на основі СВВ і дають можливість у режимі реального часу не тільки виявляти факт атаки, а й автоматично її блокувати.

## 5.5 Демілітаризована зона

Найпростішою архітектурою захищеної мережі (рис. 5.5, а) є варіант, коли всі функції захисту реалізуються одним програмно-апаратним пристроєм, наприклад, маршрутизатором або універсальним комп'ютером. Такий спосіб побудови захисту логічно найпростіший, але він має суттєвий недолік, а саме: повна залежність системи захисту від працездатності однієї ланки – прикордонного маршрутизатора.

Надійніші схеми захисту містять кілька елементів. У мережі (рис. 5.5, б) на кордоні захисту встановлено два маршрутизатори, між якими розташована мережа **демілітаризованої зони** (ДМЗ) – це мережа, яку розташовують між внутрішньою та зовнішньою мережами і яка виконує функцію буфера.

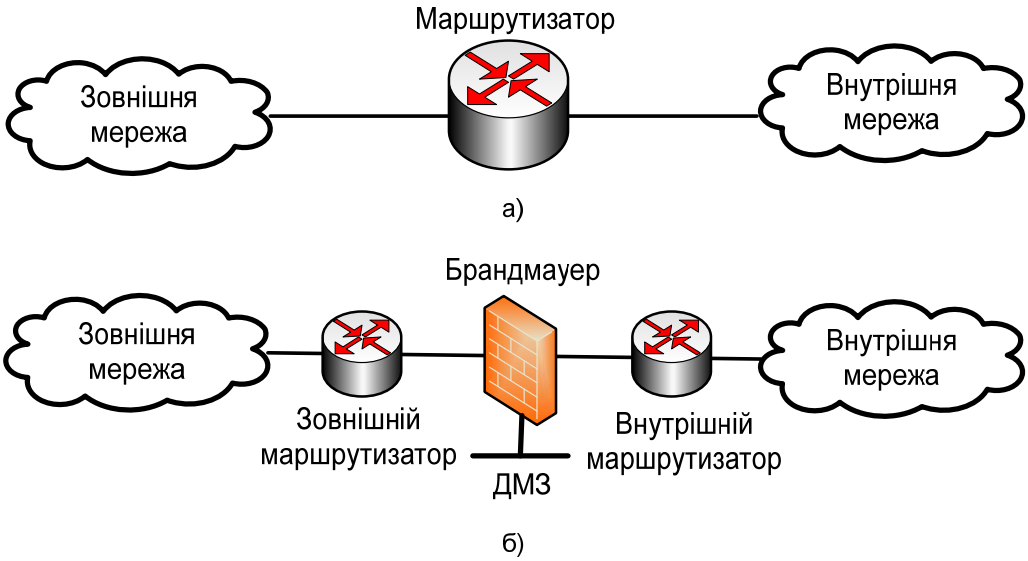


Рисунок 5.5 – Архітектура захищеної мережі на основі:
а) прикордонного маршрутизатора; б) демілітаризованої зони

У ДМЗ, зазвичай, знаходяться комп'ютери, які надають загальнодоступні сервіси, наприклад, поштовий сервер, зовнішній сервер DNS або зовнішній веб-сервер підприємства. У цій зоні також можуть бути розташовані проксі-сервери. Ураховуючи те, що сама назва цих комп'ютерів передбачає практично необмежений доступ до них з боку зовнішніх користувачів, їх необхідно захищати особливо ретельно. Головними задачами під час захисту цих комп'ютерів є забезпечення цілісності й доступності розміщених на них даних для користувачів зовнішньої мережі. Цю задачу вирішують «індивідуальні» засоби захисту, наприклад, антивірусні програми або фільтри спаму, або брандмауер.

Зовнішній маршрутизатор призначений для фільтрації трафіку з метою захисту мережі периметра й внутрішньої мережі. Проте сувора фільтрація у такому разі є непотрібною. Загальнодоступні сервери за своєю суттю призначені для практично необмеженого доступу. Що стосується захисту внутрішньої мережі, правила фільтрації для доступу до її вузлів і сервісів є аналогічними для обох маршрутизаторів, тому цю функцію доцільно покласти на внутрішній маршрутизатор.

Основна робота щодо забезпечення безпеки локальної мережі покладається на внутрішній маршрутизатор, що захищає її як від зовнішньої мережі, так і від мережі ДМЗ. Правила, визначені для вузлів мережі ДМЗ щодо доступу до ресурсів внутрішньої мережі, часто бувають суворішими, ніж правила, що регламентують доступ до даних ресурсів зовнішніх користувачів. Це робиться для того, щоб у разі зламу комп'ютера в мережі ДМЗ зменшити кількість вузлів і сервісів, які згодом можуть бути атаковані з цього комп'ютера.

# 6 ОСНОВИ ФІЛЬТРАЦІЇ ТРАФІКУ В КОМП'ЮТЕРНИХ МЕРЕЖАХ

## 6.1 Списки керування доступом

### 6.1.1. Базові поняття та принципи роботи ACL

**ACL** (Access Control List) – це набір текстових виразів, які щось дозволяють або щось забороняють. Зазвичай, ACL дозволяє або забороняє IP-пакети, але крім усього іншого він може аналізувати вміст IP-пакета, переглядати тип пакета, TCP і UDP-порти. Також ACL існує для різних мережевих протоколів (IPv4, IPv6, IPX тощо).

Списки доступу (access-lists) використовуються загалом в низці випадків і є загальним механізмом задання умов, які маршрутизатор перевіряє перед виконанням будь-яких дій. Деякі приклади використання списків доступу:

- управління передачею пакетів на інтерфейсах;
- управління доступом до віртуальних терміналів маршрутизатора й управління через SNMP;
- обмеження інформації, переданої динамічними протоколами маршрутизації.

Ідентифікатором списку доступу може бути або число (нумерований список), або алфавітно-цифрова послідовність (іменований список). Використання нумерованих або іменованих списків доступу визначається їхнім застосуванням (деякі протоколи вимагають використання тільки нумерованих списків, а деякі допускають як іменовані, так і нумеровані списки).

Якщо використовуються нумеровані списки, то значення номера визначає протокол і тип списку (табл. 6.1).

Таблиця 6.1 – Діапазони номерів типів списків доступу

| Протокол | Діапазон номерів |
|---|---|
| Стандартний список IPv4 | 1 to 99 |
| Розширений список IPv4 | 100 to 199 |
| MAC Ethernet address | 700 to 799 |
| IPX | 800 to 899 |
| Extended IPX | 900 to 999 |
| IPX SAP | 1000 to 1099 |

Правила побудови списків доступу для різних протоколів є різними, але загалом можна виділити два етапи роботи з будь-якими списками доступу. Спочатку необхідно створити список доступу, потім застосувати його до відповідного інтерфейсу, лінії або логічної операції, що виконується маршрутизатором.

Списки доступу визначають критерії, на відповідність яким перевіряється кожен пакет, що обробляється маршрутизатором. Типовими критеріями є адреси відправника й одержувача пакета, тип протоколу. Однак для кожного конкретного протоколу існує свій власний набір критеріїв, які можна задавати в списках доступу. Кожен критерій у списку доступу записується окремим рядком. Список доступу загалом є набором рядків з критеріями, що мають один і той же номер (або ім'я). При застосуванні нумерованих списків доповнення списку новими критеріями проводиться в кінці списку, що ускладнює процедуру його редагування.

Важливим є порядок задання критеріїв у списку доступу. Перевірка пакета на відповідність списку проводиться послідовним застосуванням критеріїв з цього списку (у тому порядку, у якому вони були введені). Якщо пакет задовольняє певному критерію, то подальші перевірки його на відповідність іншим критеріям у списку не здійснюються. У кінці кожного списку системою додається неявне правило – пакет, який не відповідає жодному із введених критеріїв, буде відкинутий.

Для кожного протоколу на інтерфейсі маршрутизатора може бути призначений тільки один список доступу (рис. 6.1). Для більшості протоколів можна задати окремі списки для різних напрямків трафіку. Якщо список доступу орієнтований на аналіз вхідного трафіку, то при отриманні пакета через відповідний інтерфейс маршрутизатор перевіряє критерії, задані в списку. Якщо пакет дозволений цим списком, то він передається для подальшої обробки, в іншому випадку – відкидається. Якщо список доступу призначений для аналізу вихідного трафіку, то після прийняття рішення про передачу пакета через відповідний інтерфейс маршрутизатор перевіряє критерії, задані в списку. Якщо пакет дозволений списком, то він передається для подальшої обробки. Якщо пакет заборонений, то він відкидається.

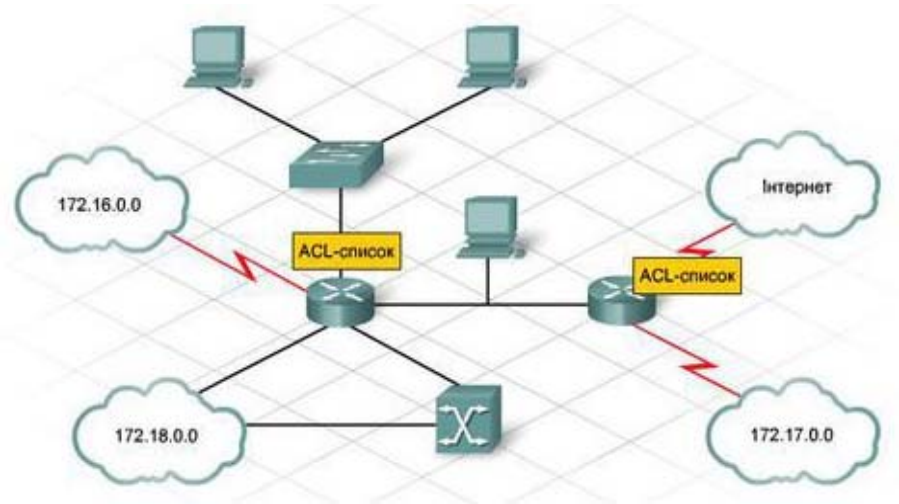


Рисунок 6.1 – Застосування вхідних та вихідних списків керування доступом

Необхідно пам’ятати, що в кінці кожного списку стоїть неявне правило «блокувати все (deny all)», тому під час призначення списків на інтерфейс потрібно явно дозволити всі види необхідного трафіку (не тільки користувальницького, а й службового, наприклад, обмін інформацією за протоколами динамічної маршрутизації). ACL досить широко застосовуються як для фільтрації трафіку, так і для інших задач, наприклад: пакетна фільтрація на інтерфейсі; обмеження доступу до маршрутизатора через віртуальні лінії; визначення трафіку, який потрібно шифрувати, при налаштуванні VPN; визначення трафіку для пріоритетного обслуговування (QoS); визначення адрес для трансляції (NAT).

### 6.1.2 Стандартні ACL

У результаті застосування списків керування доступом маршрутизатор відкидає деякі пакети, враховуючи критерії, які визначені мережевим інженером у списках. Призначення цих фільтрів полягає в блокуванні небажаного трафіку в мережі, що дозволяє не тільки створювати перешкоди перед зловмисниками, що намагаються проникнути в мережу, а й не дозволити службовцям самої компанії звертатися до тих систем, які для них мають бути закриті. Таким чином, списки управління доступом (Cisco ACL) варто розглядати як одну з частин загальної політики забезпечення безпеки організації.

Інженерам при створенні будь-якого списку керування доступом, який має застосовуватися для фільтрації IP-пакетів, доводиться приймати рішення щодо вибору одного з двох основних варіантів: які пакети підлягають фільтрації і де в мережі необхідно розмістити список управління доступом. У програмному забезпеченні Cisco IOS передбачена можливість застосовувати засоби фільтрації списку управління доступом або при вході пакета в інтерфейс, або при його виході з інтерфейсу. Іншими словами, у системі IOS список керування доступом асоціюється з інтерфейсом і, зокрема, з трафіком, що входять або виходять з інтерфейсу. Після вибору маршрутизатора, у якому планується використання списків управління доступом, необхідно вибрати інтерфейс, призначений для реалізації засобів фільтрації, а також вказати, чи поширюється дія цих засобів на вхідні та вихідні пакети (рис. 6.2).

Нижче перераховані деякі важливі особливості списків управління доступом Cisco ACL:

- Фільтрація пакетів може здійснюватися відповідно до їхнього надходження в інтерфейс, ще до прийняття рішення про маршрутизації.
- Фільтрація пакетів може проводитися перед виходом з інтерфейсу, після прийняття рішень щодо маршрутизації.
- У програмному забезпеченні Cisco IOS для вказівки на те, що пакет має бути відфільтрований, використовується термін заборона (deny).

- Якщо йдеться про те, що пакет не має бути відфільтрований, то в програмному забезпеченні Cisco IOS застосовується термін дозвіл (permit).
- Налаштування засобів фільтрації здійснюється за допомогою списків управління доступом.
- У кінці кожного списку управління доступом є неявна інструкція, яка забороняє весь трафік (deny all traffic). Тому, якщо пакет не відповідає жодній з інструкцій у списку управління доступом, він відкидається.

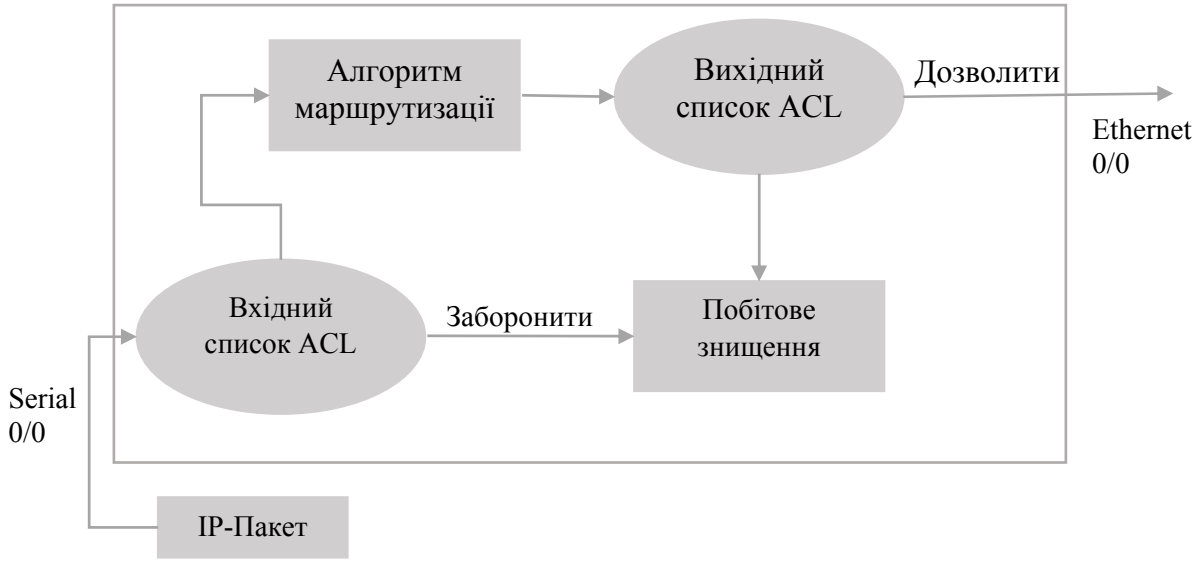


Рисунок 6.2 – Внутрішній алгоритм обробки пакетів у маршрутизаторі

Списки керування доступом складаються з двох основних компонентів: перевірка відповідності характеристик пакета правилом і дії над пакетом. Перший компонент використовується для перевірки всіх пакетів і дозволяє визначити, чи відповідає він інструкції access-list. Якщо список управління доступом складається з декількох записів, то операційна система IOS здійснює пошук в цьому списку послідовно, до виявлення першої інструкції, з якої збігаються характеристики розглянутого пакета. Принципи обробки списку керування доступом, що складається з декількох записів, можна коротко охарактеризувати так:

1) Правила списку керування доступом інструкцій access-list порівнюються із вмістом полів пакета.

2) Якщо перевірка відповідності правилам виконана успішно, здійснюється дія, яку визначено в такій інструкції access-list (permit або deny).

3) Якщо на етапі 2 не вдалося знайти збіг, етапи 1 і 2 повторюються з використанням кожної подальшої інструкції в списку управління доступом до тих пір, поки не виявиться відповідність правилу.

4) Якщо не вдається знайти збіг із жодним записом у списку керування доступом, то здійснюється дію deny (заборона пакета).

Варто пам’ятати, що стандартні списки керування доступом можуть перевіряти тільки IP-адреси відправника пакета. Для нумерування стандартних списків керування доступом Cisco ACL дозволено використовувати номери в діапазоні 1–99. Так само пам’ятайте – на один інтерфейс можна призначити тільки 1 список управління доступом. Загальний синтаксис команди настройки конструкції стандартного списку управління доступом:

**access-list** *номер-списка* **{deny | permit}** *відправник* [*інвертована маска-відправника*]

У стандартному списку керування доступом використовується низка команд **access-list**, що мають однакові номери. Команди **access-list** з однаковими номерами розглядаються як такі, що належать до одного і того ж списку, причому застосування команд у списку відбувається в тому ж порядку, у якому вони були введені в конфігурацію. Якщо процес налаштування розбити на етапи, то він матиме такий алгоритм:

**Етап 1.** Визначити, де буде розміщуватись список керування доступом (маршрутизатор та інтерфейс) і напрямок трафіку (вхідний або вихідний):

- стандартні списки керування доступом варто розміщувати з боку одержувача пакетів, щоб їхнє застосування не призводило до навмисного знищення пакетів, які не повинні бути відкинуті;
- стандартні списки керування доступом, дозволяють проводити перевірку тільки IP-адреси відправника в пакеті, тому необхідно визначити, якими будуть IP-адреси відправників у пакетах, для перевірки яких застосовується список керування доступом.

**Етап 2.** Налаштування однієї або декількох глобальних команд конфігурації **access-list** для створення списку керування доступом, керуючись зазначеними нижче правилами:

- пошук у списку відбувається послідовно, причому пакет обробляється за першого ж збігу правила з його характеристиками. Іншими словами, якщо параметри пакета збіглися з однією з інструкцій **access-list**, пошук закінчується, незважаючи на те, що пакет міг би збігтися і з однією з подальших інструкцій;
- стандартна дія у випадку, якщо пакет не співставляється з жодною з команд **access-list**, полягає в забороні проходження пакета (його знищення).

**Етап 3.** Застосування списку керування доступом в обраному інтерфейсі маршрутизатора з урахуванням потрібної спрямованості з використанням команди режиму конфігурації інтерфейсу **ip access-group number {in | out}.**

**Приклад конфігурації стандартних списків управління доступом.** Є налаштований Cisco маршрутизатор R1 з двома інтерфейсами Ethernet0/0 і Serial0/0. В інтерфейсі Ethernet0/0 увімкнена локальна мережа, в інтерфейсі Serial0/0 – зовнішня мережа. Створити Cisco ACL з номером 10 таким чином, щоб пакети проходили з локальної мережі, від комп’ютера з IP-адресою «172.16.3.10» не проходили в зовнішню мережу. Додати текс-

товий опис для ACL, що спростить розуміння, для чого використовується такий список управління доступом. Процес інсталяції:

```
R1(config)#access-list 10 remark stop traffic «172.168.3.10»
```

Створення ACL з номером 10, додання опису:

```
R1(config)#access-list 10 deny 172.16.3.10 0.0.0.0
```

Конструкція, що забороняє трафік з ip 172.16.3.10

```
R1(config-if)#ip access-group 10 out
```

Увімкнення ACL на інтерфейсі (вихідні пакети).

### 6.1.3 Розширені ACL

Розширені списки керування доступом (extended cisco access list) мають певну схожість і відмінність, порівняно зі стандартними списками керування доступом (cisco acl). Як і стандартні, розширені списки керування доступом призначаються на інтерфейсах маршрутизаторів. Система Cisco IOS проводить пошук у цьому списку послідовно. При виявленні першого ж співпадіння параметра пакета з інструкцією пошук у списку припиняється і визначається команда, яка підлягає виконанню.

Однією з основних відмінностей між списками двох типів є те, що в розширених списках керування доступом (extended cisco access list) для виявлення збігу доводиться перевіряти більшу кількість полів у пакеті. Достатньо налаштувати одну інструкцію розширеного списку керування доступом, щоб перевірити відразу кілька фрагментів інформації в заголовку пакета, причому обов'язковою є вимога, що всі параметри мають збігатися з правилами, щоб пакет відповідав інструкції списку керування доступом. Саме завдяки вказаному принципу перевірки пакетів, розширені списки керування доступом стають, водночас, і набагато кориснішими, і набагато складнішими, порівняно зі стандартними списками керування доступом.

Розширені списки керування доступом дозволяють реалізувати потужні механізми перевірки полів пакета на збіг із заданими критеріями, оскільки дозволяють перевіряти різні частини пакета. Нижче наведено список параметрів, які можна перевірити за допомогою розширених списків керування доступом:

- IP-адреса одержувача.
- Частини IP-адрес одержувача, указані за допомогою інвертованої маски підмережі.
- Тип протоколу (TCP, UDP, ICMP, IGRP, IGMP та ін.).
- Порт відправника.
- Порт одержувача.
- TCP-потоки.
- Байти TOS по протоколу IP.
- Пріоритет пакета IP.

Найбільш поширені TCP і UDP-додатки, а також передбачені для них стандартні номери портів наведені в табл. 6.2.

Таблиця 6.2 – Найбільш поширені TCP і UDP-додатки та їхні порти

| Номер порту | Протокол | Додаток | Ключове слово з визначенням назви додатку в синтаксисі команди access-list |
|---|---|---|---|
| 20 | TCP | FTP | data ftp-data |
| 21 | TCP | Протокол керування FTP | ftp |
| 22 | TCP | SSH | відсутній |
| 23 | TCP | Telnet | telnet |
| 25 | TCP | SMTP | smtp |
| 53 | UDP, TCP | DNS | domain |
| 67, 68 | UDP | DHCP | nameserver |
| 69 | UDP | TFTP | tftp |
| 80 | TCP | HTTP(WWW) | www |
| 110 | TCP | POP3 | pop3 |
| 161 | UDP | SNMP | snmp |
| 443 | TCP | SSL | відсутній |
| 16384 | UDP | VoIP) та відео | відсутній |

Розширені списки керування доступом забезпечують перевірку поля протоколу в заголовку пакета, а також номери портів TCP або UDP відправника і одержувача. Варто пам'ятати такі речі:

- У команді **access-list** має використовуватися ключове слово **tcp**, щоб з його допомогою можна було перевіряти номери TCP-портів (транспортний рівень), і ключове слово **udp** – для перевірки номерів портів протоколу UDP. Ключове слово ip не забезпечує перевірку номерів портів.
- Параметри із зазначенням порту відправника й порту одержувача в команді **access-list** є позиційними. Тобто їхнє місцезнаходження у команді визначає, для якого з портів застосовується параметр – відправника чи одержувача.
- Списки керування доступом дозволяють проводити перевірку пакетів, відправлених хосту, шляхом порівняння номера порту одержувача із зарезервованим номером порту. У списках керування доступом необхідно перевіряти номер порту відправника пакетів, відправлених хосту.

**Конфігурація розширених списків керування доступом:**

1)**access-list** *номер* **{deny | permit}** протокол адреса-відправника інвертована-маска-відправника адреса-отримувача інвертована-маска-отримувача **{log | log-input}** – глобальна команда для розширених нумерованих списків керування доступом. Використовуються номера **100-199**.

2)**access-list** *номер* **{deny | permit} {tcp | udp}** *адреса-відправника інвертована-маска-відправника* **[***оператор* **[***порт***]]** *адреса-отримувача інвертована-маска-отримувача* **[***оператор* **[***порт***]]** [**established**] **[log]** – версія команди **access-list** з параметрами, які належать до протоколу TCP або UDP.

Процес налаштування розширених списків керування доступом (**extended cisco access list**), загалом, збігається з аналогічним процесом стандартних списків керування доступом. Насамперед, необхідно вибрати місце розташування і напрямок, щоб можна було планувати застосування параметрів списку керування доступом з урахуванням інформації в пакетах. Налаштування конфігурації списку керування доступом необхідно виконувати за допомогою команд **access-list**. Після цього список керування доступом потрібно увімкнути за допомогою команди **ip access group** (на потрібному інтерфейсі), яка застосовується для роботи зі стандартними списками керування доступом. Особливістю налаштування розширених списків керування доступом є те, що в самій конфігурації є певні відмінності, які можна коротко описати так:

- Розширені списки керування доступом розміщувати якомога ближче до відправника пакетів, що підлягають фільтрації, оскільки конфігурація розширених ACL може бути налаштована так, щоб ці списки не блокували проходження пакетів, які не підлягають блокуванню. Іншими словами, застосування фільтрації в точці, розташованій ближче всього до відправника пакетів, сприяє оптимізації використання пропускної спроможності.
- Пакет розглядається як такий, що збігається з інструкцією **access-list**, тільки в тому випадку, якщо всі параметри в одній з команд **access-list** збігаються з відповідними полями пакета.
- Для розширеної команди **access-list** можуть використовуватися номери **100-199**, причому жоден номер не розглядається як пріоритетніший відносно до іншого.

Розширена версія команди **access-list** дозволяє перевіряти номери портів з використанням декількох простих операторів порівняння таких, як «дорівнює» (**equal-to**) або «менше» (**less-than**).

Приклади команд access-list розширеного ACL:

- **access-list 101 deny ip any host 10.1.1.1** – будь-який IP-пакет (мережевий рівень), будь-яка IP-адреса відправника, з IP-адресою отримувача 10.1.1.1;
- **access-list 101 deny tcp any gt 1023 host 10.1.1.1 eq 23** – пакети із заголовком TCP, будь-якою IP-адресою відправника, з номером порту відправника, який більше (**gt**) 1023, з IP-адресою отримувача, яка дорівнює 10.1.1.1, і номером порта отримувача, який дорівнює (**eq**) 23;
- **access-list 101 deny tcp any host 10.1.1.1 eq 23** – ідентично попередньому прикладу, але тут не виконується порівняння з будь-яким портом відправника, оскільки вказаний параметр в команді відсутній;
- **access-list 101 deny tcp any host 10.1.1.1 eq telnet** – ідентично попередньому прикладу. Замість зазначення порту 23, використовується ключове слово **telnet**;

– **access-list 101 deny udp 1.0.0.0 0.255.255.255 lt 1023 any** – пакет з відправником у мережі 1.0.0.0, у якій використовується протокол UDP з портом відправника, який менше (**lt**) 1023, із будь-якою IP-адресою отримувача.

Налаштування cisco access list. Маємо маршрутизатор R1, з двома активними інтерфейсами ethernet0/0 і ethernet0/1. До інтерфейсу ethernet0/0 підключено локальну мережу 172.16.1.0/24, у якій працюють хости користувачів, до інтерфейсу ethernet0/1 – мережу з серверами 172.168.2.0/24. Мета – комп'ютеру 172.16.1.10 заборонити доступ до FTP-серверів, комп'ютеру 172.16.1.20 – доступ до веб-служб сервера 172.16.2.100.

```
R1(config)#access-list 111 deny tcp host 172.16.1.10
172.16.2.0 0.0.0.255 eq ftp
R1(config)#access-list 111 deny tcp host 172.16.1.20 host
172.16.2.100 eq www
R1(config)#interface ethernet0/0
R1(config-if)#ip access group 111 in
```

### 6.1.4 Іменовані ACL

Іменовані списки керування доступом і порядкові номери списків керування доступом дозволяють простіше запам'ятовувати їхні імена і редагувати існуючі списки управління доступом, якщо буде потрібно їх змінити. Іменовані ACL вирізняються зручністю застосування, оскільки їм можна вказати їхнє призначення. Вимоги до імен: можуть містити літери і цифри; передбачається, що імена будуть писатися великими літерами; імена не можуть містити пробіли і знаки пунктуації. В іменовані ACL можна додавати і видаляти записи. Іменовані списки ACL IP мають багато схожого з нумерованими списками ACL IP. Вони застосовуються для фільтрації пакетів, а також для багатьох інших цілей. Подібно стандартним і розширеним нумерованим списками ACL, які відрізняються можливостями розпізнавання пакетів, іменовані списки ACL можуть бути стандартними і розширеними. Спочатку іменовані списки ACL мали три істотні відмінності від нумерованих списків ACL:

– Замість номерів для ідентифікації списків ACL використовуються імена, які полегшують запам'ятовування причин їхнього застосування.
– Для визначення дій і параметрів розпізнавання використовуються команди підсистеми ACL, а не глобальні команди.
– Кращі інструменти редагування списків ACL.

Єдина нова частина конфігурації іменованих списків ACL це глобальна команда конфігурації **ip access-list**, яка визначає, чи є список ACL стандартним або розширеним, а також визначає ім'я. Вона також переводить користувача в режим конфігурації ACL.

**Іменовані стандартні ACL.**

**R1(config)# ip access-list standard NAME** – оголошення іменованого стандартного ACL;

**R1(config-std-nacl)# remark Deny for host 192.168.0.13** – опис ACL;

**R1(config-std-nacl)# deny 192.168.0.13** – створення правила;

**R1(config-std-nacl)# permit 192.168.0.0 0.0.0.255**

**R1(config-std-nacl)# interface Fa0/0**

**R1(config-if)# ip access-group NAME out** – прив'язка ACL до інтерфейсу.

Називати ACL великими літерами не обов'язково. Це робиться для зручності. Перегляд і перевірка ACL:**Router # show access-lists {access-list-mumber | name}.** Наприклад:

**R1 # show access-lists** – виводить усі ACL;

**R1 # show access-lists 10** – виводить ACL з номером 10;

**R1 # show access-lists NAM** – виводить ACL з іменем NAM.

В іменованих ACL є перевага перед нумерованими – їх простіше редагувати. Усі записи правил в іменованих ACL мають порядковий номер з кроком 10. Тобто перше правило має номер 10, друге – 20 і т. д. Тому в іменованих ACL можна видаляти вказані записи, а також додавати записи між наявними правилами з присвоєнням їм номера між номерами правил, між якими додається нове правило. Наприклад, є ACL з такими записами:

```
R1 # show access-lists
Standard IP access-list WEBSERVER
10 permit 192.168.10.10
20 deny 192.168.10.0, wildcard bits 0.0.0.255
30 deny 192.168.12.0, wildcard bits 0.0.0.255
```

Додамо ще одне правило:

```
R1 (config) # access-list standard WEBSERVER
R1 (config-std-nacl) # 15 permit 192.168.10.13
```

Отримаємо такий результат:

```
R1 # show access-lists
Standard IP access-list WEBSERVER
10 permit 192.168.10.10
15 permit 192.168.10.13
20 deny 192.168.10.0, wildcard bits 0.0.0.255
30 deny 192.168.12.0, wildcard bits 0.0.0.255
```

**Створення іменованого розширеного ACL.**

**R1(config)# ip access-list extended SURFING** – оголошення іменованого розширеного ACL для вихідного трафіку;

**R1(config-ext-nacl)# pernit tcp 192.168.10.0 0.0.0.255 any eq 80** – створення правила;

**R1(config-ext-nacl)# pernit tcp 192.168.10.0 0.0.0.255 any eq 443**

**R1(config)# access-list extended BROWSING** – оголошення іменованого розширеного ACL для вхідного трафіку;

**R1(config-ext-nacl)# pernit tcp any 192.168.10.0 0.0.0.255 established** – створення правила.

### 6.1.5 Конфігурування TCP-established та reflexive ACL

У сучасних мережах firewall має бути розташований між внутрішньою і зовнішньою мережами. Основна ідея – це блокування трафіку, який приходить зовні, за винятком, трафіку, що пропускається ACL та трафіку, що є відповіддю на ініційований зсередини.

Багато сучасних програм використовують TCP, що створює віртуальне з'єднання між кінцевими точками. Перше покоління фільтрів для врахування двоспрямованої структури TCP з'єднань використовувало ключове слово **established** для розширеного фільтра. Такі фільтри блокували увесь трафік, що надходив з Інтернету, за винятком, трафіку – відповіді для з'єднань, що ініційовані зсередини мережі.

Ще одне покоління – рефлексивні ACL. Ці фільтри базуються на адресах відправника, отримувача, на номерах портів і відслідковують сесію. Вони використовують тимчасові фільтри, які видаляються після завершення сесії. Команда конфігурування TCP **established** фільтра:

```
Router(config)# access-list {100-199} {permit | deny}
protocolsource-addr [source-wildcard] [operator operand] des-
tination-addr [destination-wildcard] [operator operand] [es-
tablished]
```

Ключове слово **established** примушує маршрутизатор перевіряти, чи встановлені біти ACK або RST. Якщо ні – то ці сегменти вважаються новим з'єднанням і відкидаються. У цьому випадку жодна сеансова інформація не використовується для відслідковування трафіку. Такий фільтр досить легко обійти шляхом установлення відповідних бітів у хакерські сегменти. Крім того, цей фільтр не дозволяє фільтрувати UDP або ICMP-трафік. Не дивлячись на недоліки, використання **established** дозволяє покращити безпеку. Приклад:

```
R1(config)# access-list 100 permit tcp any eq 443 192.168.1.0 0.0.0.255 estab-
lished
R1(config)# access-list 100 permit tcp any 192.168.1.3 0.0.0.0 eq 22
R1(config)# access-list 100 deny ip any any
R1(config)# interface s0/0/0
R1(config-if)# ip access-group 100 in
```

Рефлексивні ACLs забезпечують більш коректну форму фільтрації трафіку TCP-з'єднання, тому що перевіряють значно більше параметрів. Наприклад, адреси й порти відправника та отримувача. Крім того, сесійні фільтри використовують тимчасові фільтри, які видаляються після завершення зв'язку, що також обмежує хакерів у часі.

Для реалізації цієї функції іменований розширений ACL перевіряє трафік, що виходить з мережі. Фільтр може бути призначений або як вхідний для внутрішнього інтерфейсу, або як вихідний – для зовнішнього. ACL перевіряє трафік, пов'язаний з новою сесією, використовуючи параметр reflect. На основі чого динамічно будується ACL, який дозволяє проходження зворотного трафіку. Як тільки відповідний трафік виходить з мережі, тимчасовий запис додається до ACL. Для кожного запису з опцією reflect маршрутизатор будує окремий ACL.

Рефлексивний ACE містить інвертований варіант адрес і портів відправника й отримувача, команду permit.

Етапи конфігурування рефлексивних ACL:

1) Створити внутрішній ACL, який буде спостерігати за новими вихідними сесіями та створюватиме рефлективні ACEs.

2) Створити зовнішній ACL, який використовує рефлексивні ACL для перевірки зворотного трафіку.

3) Активувати іменований ACL на відповідних інтерфейсах.

Синтаксис для внутрішнього ACL:

```
Router(config)# ip access-list extended internal_ACL_name
Router(config-ext-nacl)# permit protocol source-addr [source-mask] [operator operand] destination-addr [destination-mask] [operator operand] [established] reflect reflexive_ACL_name [timeout seconds]
```

Наприклад, ці команди для обслуговування внутрішніх користувачів, що працюють з web-browser та DNS.

```
R1(config)# ip access-list extended internal_ACL
R1(config-ext-nacl)# permit tcp any any eq 80 reflect web-only-reflexive-ACL
R1(config-ext-nacl)# permit udp any any eq 53 reflect dns-only-reflexive-ACL timeout 10
```

Cisco IOS створить 2 рефлексивних записи, що підтримують сесійну інформацію для вихідних веб-з'єднань та DNS-запитів.

Далі на тимчасові записи має бути посилання, щоб вони здійснювали фільтрацію зворотного трафіку. Це робиться шляхом створення другого ACL. У ньому використовується слово evaluate для посилання на рефлексивні ACEs, що були створені за допомогою внутрішнього ACL.

```
Router(config)# ip access-list extended external_ACL_name
Router(config-ext-nacl)# evaluate reflexive_ACL_name
Для нашого прикладу:
R1(config)# ip access-list extended external_ACL
R1(config-ext-nacl)# evaluate web-only-reflexive-ACL
R1(config-ext-nacl)# evaluate dns-only-reflexive-ACL
R1(config-ext-nacl)# deny ip any any
Останній крок – асоціювання з інтерфейсом:
R1(config)# interface s0/0/0
R1(config-if)# description connection to the ISP.
R1(config-if)# ip access-group internal_ACL out
R1(config-if)# ip access-group external_ACL in
```

### 6.1.6 Динамічні ACLs

Динамічні ACLs відомі як lock-and-key ACLs. Є доступними тільки для IP-трафіку. Залежать від Telnet-з'єднання, аутентифікації (локальної або віддаленої) та розширених ACLs. Конфігурування починається з того, що створюється ACL, який блокує увесь трафік, що проходить через маршрутизатор. Користувачам, яким треба пройти через маршрутизатор, потрібно утворити Telnet-з'єднання і пройти процедуру аутентифікації. Далі Telnet-з'єднання розривається, а однорядковий ACL додається до існуючого ACL. Він дозволяє проходження трафіку протягом певного періоду. Можливі

налаштування періоду як за часом пасивності, так і фіксоване значення часу.Динамічні ACL, з одного боку, забезпечують доступ до мережі користувачам, які цього потребують, а з іншого – мережа захищена брандмауером. Динамічні ACL мають низку переваг, зокрема, дозволяють зробити таке: наприклад у вас є маршрутизатор, який підключений до сервера і потрібно закрити доступ до нього із зовнішнього світу, але в той же час є декілька людей, які можуть підключатися до сервера.

Можна налаштувати динамічний список доступу, прикріпити його на вхідному напрямку, а далі людям, яким потрібно підключитися до віддаленого пристрою, роблять це засобами Telnet до такого пристрою і в результаті динамічний ACL відкриває вхід на сервер. Тоді користувач може зайти, скажімо, через HTTP та потрапити на сервер. За замовчуванням через 10 хвилин цей вхід закривається і користувач змушений ще раз виконати Telnet, щоб підключитися до пристрою.

**Функціонування динамічних ACL**

1) Віддалений користувач має відкрити Telnet або SSH з'єднання з маршрутизатором. Зовнішній ACL має це дозволити. Маршрутизатор запитує ім'я і пароль користувача.

2) Маршрутизатор аутентифікує з'єднання використовуючи або локальну базу даних, або AAA-сервер. Якщо аутентифікація успішна, Telnet або SSH-з'єднання розривається.

3) Cisco IOS додає динамічний ACL запис, що гарантує доступ до внутрішніх ресурсів. У такому разі немає потреби створювати окремі політики для кожного користувача. Адміністратор створює одну політику для всіх користувачів, і ця політика застосовується до всіх аутентифікованих користувачів.

4) Користувач отримує доступ до внутрішніх ресурсів, до яких він не мав доступу без динамічного ACL.

**Етапи конфігурування динамічних ACL**

1) Створення зовнішнього ACL, який дозволяє Telnet або SSH-з'єднання з маршрутизатором.

2) Визначення аутентифікації. Динамічні ACL підтримують такі методи аутентифікації: локальну, зовнішній AAA-сервер та звичайний пароль. Останній варіант, як правило, не використовується, оскільки всі користувачі повинні мати однаковий пароль.

3) Активація методу динамічної аутентифікації. Це відбувається на vty-лініях маршрутизатора. Коли активацію зроблено, маршрутизатор може створювати динамічні записи в ACL інтерфейса, на який посилається динамічний ACL.

**Команда створення динамічного ACL**

```
Router(config)# access-list {100-199} dynamic dynamic_ACL_name [timeout minutes] {permit | deny} protocol source-addr [source-wildcard] [operator operand] destination-addr [destination-wildcard] [operator operand] [established]
```

Ключове слово **dynamic** дозволяє адміністратору визначити ім'я для динамічного ACL. Це ім'я має бути унікальним на цьому маршрутизаторі. Параметр **timeout** (1–9999 хв) визначає абсолютний час роботи ACL.

З динамічними ACL асоціюються два тайм-аути: абсолютний та пасивний. Останній визначається командою **autocommand**, яка активує **lock-and-key** аутентифікацію на **vty** лінії. Якщо тайм-аут не заданий, то за замовчуванням він нескінченний. Доцільно задавати або один з тайм-аутів або обидва. За тайм-аутом задаються адреси користувачів, яким дозволено доступ, і як правило, це можуть бути будь-які адреси.

Після створення зовнішнього ACL, що дозволяє Telnet або SSH та містить параметр dynamic, ACL призначається на відповідний інтерфейс маршрутизатора командою **ip access-group.** Останній етап – увімкнути **lock-and-key** аутентифікацію на **vty**.

```
Router(config)# line vty 0 4
Router(config-line)# autocommand access-enable host [timeout
minutes]
```

Команда **autocommand access-enable** визначає **lock-and-key** аутентифікацію. Після успішної аутентифікації тимчасовий ACL-запис вставляється в розширений ACL в те місце, де знаходиться параметр dynamic. Запис додається тільки на одному інтерфейсі, до якого користувач під'єднується. Без команди **autocommand access-enable** маршрутизатор не створюватиме динамічний ACL. Параметр **host** необов'язковий. Якщо він заданий, Cisco IOS замінить слово **any** в динамічному ACL на IP-адресу користувача. Якщо ACL призначений **inbound**, то буде замінено **any** в адресі відправника, якщо **outbound** – в адресі отримувача. Призначення параметра **timeout** було описано вище.

Таблиця 6.3 – Етапи надаштування динамічного списку доступу

| | |
|---|---|
| Етап 1 | R3(config)# **username Student password 0 cisco** |
| Етап 2 | R3(config)# **acces-1ist 101 permit tcp any host 10.2.2.2 eq telnet**<br>R3(config)# **access-list 101 dynamic testlist timeout 15 permit ip 192.168.10.0 0.0.0.255 192.168.30.0 0.0.0.255** |
| Етап 3 | R3(config)# **interface serial 0/0/1**<br>R3(config-if)# **ip access—group 101 in** |
| Етап 4 | R3(config)# **line vty 0 4**<br>R3(config-line)# **login local**<br>R3(config—line)# **autocommand access-enable host timeout 5** |

### 6.1.7 ACL, що базуються на часі

У цьому випадку визначається часовий інтервал, коли ACL буде працювати:

```
Router(config)# time-range time_range_name
Router(config-time-range)# absolute [start_time start_date]
[end_time end_date]
Router(config-time-range)# periodic day_of_the_week hh:mm to
[day_of_the_week] hh:mm
```

```
Router(config)# access-list {100-199} {permit | deny} proto-
col source-addr [source-mask] [operator operand] destination-
addr [destination-mask] [operator operand] [established] [log
| log-input] [established] [time-range name_of_time_range]
```

## 6.2 Firewall

Firewall – це система або група систем, які реалізують політику керування доступом між мережами. Будь-який Firewall має такі властивості: здійснює опір атакам; є транзитною точкою між мережами; реалізує політику керування доступом.

1988 р. DEC створила перший firewall, який фактично був пакетним фільтром. Пакетні фільтри належать до, так званих, stateless (ті, що не враховують стан) firewall. Оскільки у такому випадку не аналізується, чи є пакет складовою частиною певного потоку (наприклад, TCP-з'єднання). Кожний пакет обробляється окремо незалежно від інших пакетів. 1989 р. AT&T Bell Laboratories розробила перший stateful (експертний firewall, цей firewall може визначати, чи належить той чи інший пакет певному потоку даних. Статичні правила, що є характерними для пакетних фільтрів, замінюються на динамічні, які утворюються в режимі реального часу. Експертний firewall допомагає запобігти DoS-атакам. На першому етапі firewalls не були окремими спеціальними пристроями, а реалізовувались програмно. Сьогодні з'явились апаратні реалізації. Виділені firewall дозволяють вивільнити пам'ять та ресурси маршрутизатора від задач, пов'язаних з фільтрацією трафіку. Сучасні маршрутизатори, зокрема Cisco Integrated Services Routers (ISRs), також можуть використовуватись як інтелектуальні експертні firewalls в організаціях, що не мають можливості придбати окремий пристрій. Переваги використання firewall.

1) Запобігання доступу до критичних вузлів та програм нелегітимних користувачів.

2) Блокування небезпечного програмного забезпечення.

3) Реалізація політики безпеки може бути простою, стійкою та масштабованою у разі вірно сконфігурованого firewall.

4) Переведення великої кількості точок доступу до мережі до обмеженої кількості може спростити реалізацію захисту мережі.

Обмеження, пов'язані з використанням firewall.

1) У випадку невірної конфігурації використання firewall може давати дуже погані наслідки.

2) Багато програм не можуть працювати коректно через firewall.

3) Користувачі можуть шукати шляхи обходження firewall, що погіршує захищеність мережі.

4) Продуктивність мережі зменшується.

5) Неавторизований трафік може тунелюватись або маскуватись під легітимний.

Виділяють такі різновиди МЕ:

- Пакетні фільтри. 3-й, 4-й рівні моделі OSI, статичний firewall, ACL.
- Експертний (Stateful) firewall. Відслідковує стан з'єднання.
- Firewall рівня додатків (proxy firewall) 3-й, 4-й, 5-й, 7-й рівні, переважно, програмна реалізація.
- Firewall трансляції адрес (NAT).
- Host-based firewall (серверний та персональний) – комп'ютер з програмним firewall.
- Прозорий firewall фільтрує трафік між парою інтерфейсів.
- Гібридний firewall комбінує функції різних типів.

МЕ експертного рівня (Stateful firewall) є найбільш універсальним. Він використовує інформацію про встановлене з'єднання. Як правило, працює на 3-му та 4-му рівнях моделі OSI, хоча в деяких випадках аналізується також інформація вищих рівнів. Використовує таблицю стану для відслідковування актуальних комунікаційних процесів, наприклад, у TCP-сегменті аналізує службові біти для визначення стану з'єднання.

Як тільки здійснюється доступ до зовнішнього сервісу, експертний firewall зберігає певну інформацію щодо запиту, зберігаючи інформацію про стан запиту в таблиці станів. Коли зовнішня система відсилає відповідь на запит, firewall порівнює пакет із збереженим станом і дозволяє або забороняє доступ до мережі. Таблиця стану містить інформацію про адреси та порти відправника й отримувача, інформацію, що впорядковує TCP-сегменти та додаткову інформацію, специфічну для кожного сеансу. Ця інформація створює об'єкт зв'язку, який використовується firewall для порівняння з пакетами, що надходять у межах сесії. Firewall дозволяє проходження даних тільки у випадку, якщо вони передаються в межах існуючого з'єднання. Більш розумні експертні firewall охоплюють можливість розпізнавання команд FTP-порту і оновлювати таблицю станів, дозволяючи FTP працювати прозоро через firewall. Також розумний firewall може аналізувати номер послідовності в TCP-сегменті, відповідність DNS-запитів та відповідей, ця опція зменшує небезпеку загрози TCP RST flood атаки та некоректне заповнення DNS-кеша.

Недоліком експертного firewall є те, що адреси внутрішньої мережі передаються в пакетах, що йдуть назовні, ці недоліки усунені при застосуванні NAT та проксі-серверів. Cisco Systems надає кілька інструментів для спеціалістів з безпеки: Cisco IOS Firewall, PIX Security Appliances (вважається застарілим) Adaptive Security Appliances. МЕ на основі Cisco IOS – це спеціальні опції в операційній системі, застосовуються в невеликих та середнього рівня організаціях.

ASA – багатофункціональний пристрій, який реалізує багаторівневий захист для різного типу трафіку і може використовуватись в мережах різного рівня. Вибір на користь того чи іншого рішення здійснюється на основі порівняння вартості й ризику. У технологіях захисту мереж часто використовують термін ДМЗ – це частина мережі, обмежена firewall ДМЗ визначає частини мережі, яким можна довіряти і яким неможна довіряти.

Найпростіший варіант – це firewall на кордоні приватної і публічної мережі (рис. 6.3). Причому, як правило, вихідний трафік пропускається майже увесь, а вхідний переважно блокується. Пропускається тільки трафік, що передається зовні в межах сеансів, ініційованих зсередини.

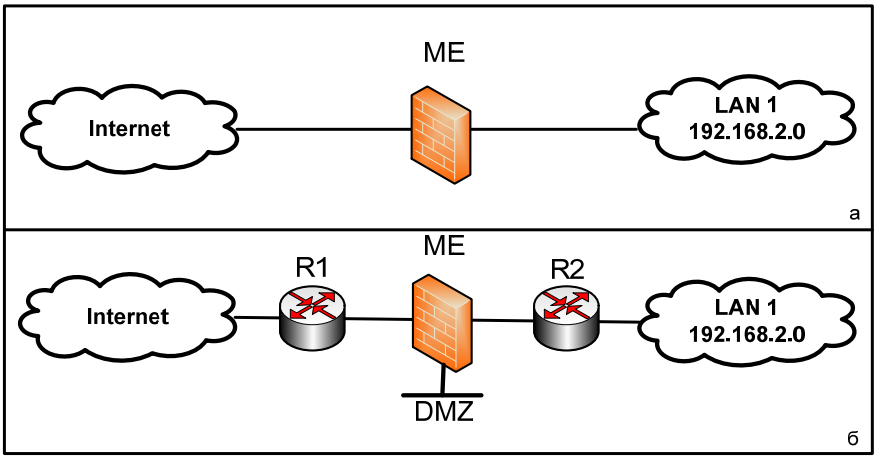


Рисунок 6.3 – Реалізація firewall на кордоні приватної й публічної мережі

Часто використовується архітектура firewall з трьома інтерфейсами – один в приватну мережу, один – у публічну, один – у ДМЗ. Трафік з внутрішньої мережі вільно пропускається як у зовнішню мережу, так і в ДМЗ. Трафік з ДМЗ вільно пропускається у внутрішню мережу, але блокується у зовнішню. Трафік зовні, як правило, блокується повністю, за винятком рефлексивного, ініційованого або з приватної мережі, або з ДМЗ. Однак при застосуванні ДМЗ певні типи трафіку, ініційовані зовні, пропускаються в ДМЗ зону, наприклад, DNS, HTTP, HTTPS.

При застосуванні багаторівневого сценарію захисту firewalls забезпечують захист периметра всієї мережі та найважливіших сегментів внутрішньої мережі. Наприклад, захист фінансової мережі від інших сегментів внутрішньої мережі.

Наприклад, трафік, що надходить з публічної мережі спочатку обробляється пакетним фільтром зовнішнього маршрутизатора, далі потрапляє на екранований firewall або хост-бастіон, на якому налаштовано значно більше правил. Хост-бастіон – це добре захищений комп'ютер, розташований у ДМЗ-зоні. Далі трафік передається на внутрішній екранований маршрутизатор. Трафік потрапляє у внутрішню мережу тільки успішно пройшовши всі фільтри. Такий тип ДМЗ називається конфігурацією екранованих підмереж. На перший погляд складається думка, що багаторівнева топологія захисту на основі МЕ дозволяє надійно захистити внутрішню мережу. Однак МЕ не завжди може захистити мережу від вірусів, що надходять електронною поштою, не допоможуть проти несанкціонованих модемів, не замінять резервного копіювання. Для надійного захисту потрібні зовнішні місця збереження даних та використання надлишкових пристроїв.

Для забезпечення ефективного захисту варто дотримуватись базових рекомендацій для реалізації політики захисту на основі МЕ:

1) МЕ мають розташовуватись на найкритичніших, з погляду безпеки на межах мережі.
2) МЕ є основними пристроями захисту мережі, але вони не є панацеєю.
3) На МЕ варто блокувати увесь трафік за замовчуванням, а дозволяти тільки потрібні сервіси.
4) Потрібно жорстко контролювати фізичний доступ до МЕ.
5) Необхідно регулярно переглядати та аналізувати системні повідомлення (лог-файли), що генеруються МЕ.
6) Необхідно дуже уважно ставитись до змін, що робляться в конфігурації МЕ.

МЕ здебільше захищає від технічних атак зовні, внутрішні атаки можуть мати нетехнічну природу.

### 6.2.1 МЕ керування доступом на основі контексту

Керування доступом на основі контексту (Context-based access control (CBAC) – це функція яка є доступною в МЕ на основі Cisco IOS. CBAC інтелектуально фільтрує TCP та UDP-трафік використовуючи інформацію рівня додатків (7). Він забезпечує експертну фільтрацію на програмному рівні, враховуючи особливості того чи іншого протоколу рівня додатків. Також можуть бути коректно обслуговані протоколи, що використовують кілька логічних з'єднань (FTP, H.323). CBAC можуть врахувати особливості, які створюють NAT та PAT, можуть блокувати P2P-з'єднання, а також трафік миттєвих повідомлень. CBAC реалізують 4 основні функції: фільтрацію трафіку, перевірку, детектування зловмисників, генерацію повідомлень аудиту та застережень.

**Фільтрація трафіку.** CBAC може бути сконфігурований, щоб дозволяти тільки рефлексивний трафік, аналогічно до firewall експертного рівня. CBAC аналізує не тільки інформацію мережевого і транспортного рівня, але і рівня додатків (наприклад, інформацію про FTP-з'єднання) для визначення стану з'єднання. Це дозволяє обслуговувати протоколи, які використовують кілька з'єднань для своєї роботи.

**Перевірка трафіка.** CBAC перевіряє в пакетах інформацію рівня додатків, а також інформацію TCP та UDP-заголовків, що дозволяє запобігати певному типу атак, зокрема, атакам типу SYN-flooding. Для цього перевіряється поле «номер в послідовності», і якщо воно не задовольняє певному діапазону, пакет відкидається, CBAC може також закривати TCP-з'єднання, установлення яких не було завершено.

**Детектування зловмисника.** CBAC перевіряє в пакетах інформацію рівня додатків, а також інформацію TCP та UDP-заголовків, що дозволяє запобігати певному типу атак, зокрема, атакам типу SYN-flooding. Для

цього перевіряється поле «номер в послідовності», і якщо воно не задовольняє певному діапазону, пакет відкидається, CBAC може також закривати TCP-з'єднання, установлення яких не було завершено.

**Аудит та застереження.** CBAC дозволяє здійснювати процедури аудиту та генерування застережень (Alert) у зв'язку з подіями, що несуть небезпеку. Базові можливості CBAC:

- Моніторинг установлення TCP-з'єднання.
- Підтримка інформації про UDP-сесію.
- Відслідковування послідовності TCP-сегментів.
- Інспекція DNS-запитів та відповідей.
- Інспекція типів ICMP-повідомлень.
- Підтримка програм, що використовують кілька з'єднань.
- Перевірка інкапсульованих адрес (при застосуванні тунелів).
- Перевірка інформації рівня додатків.

Варто зауважити, що CBAC контролює тільки ті протоколи, для яких здійснені налаштування, а інші блокуються. Контролюються тільки запити, що проходять через ME. Без застосування CBAC фільтрація трафіку обмежується ACL, який перевіряє пакети на мережевому і, максимум, транспортному рівні. CBAC базується на експертному пакетному фільтрі, специфічному для кожного протоколу рівня додатків. Це означає, що також аналізується інформація рівня додатків, підтримується і аналізується інформація в таблиці станів (або зв'язків) для відслідковування активних сесій. Таблиця станів відслідковує сесії і перевіряє всі пакети, що проходять через ME. CBAC використовує таблицю станів для створення динамічних записів, що дозволяють проходження рефлективного трафіку через ME. Під час своєї роботи CBAC створює «отвір» шляхом додавання тимчасових ACL-записів для специфічних сесій. Ці «отвори» утворюються, коли специфічний трафік виходить із захищеної мережі в публічну через ME. Таблиця станів динамічно змінюється і адаптується до потоків трафіку.

Припустимо, що користувач ініціює вихідне з'єднання, наприклад, Telnet із захищеної мережі у зовнішню мережу (рис. 6.4) і CBAC дозволяє інспектування Telnet-трафіку. Також припустимо, що ACL, розташований на зовнішньому інтерфейсі, забороняє проходженню Telnet-трафіку. У такому разі з'єднання проходить багатоетапну обробку.

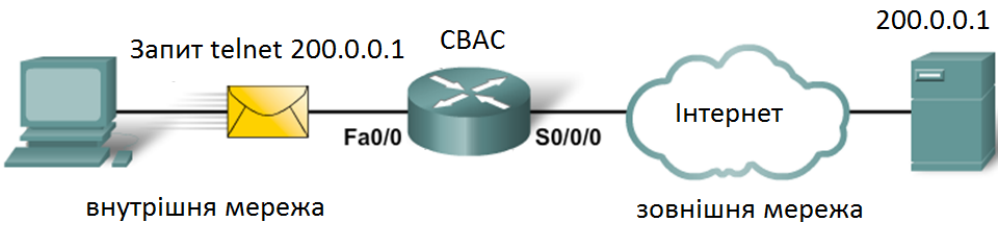


Рисунок 6.4 – Реалізація інспектування Telnet-трафіку

1) Якщо на інтерфейс маршрутизатора, через який надходить трафік, налаштовано вхідний ACL, то пакет проходить перевірку. У разі дозволу починається перевірка правил CBAC.
2) Якщо в правилах CBAC нічого не сказано про Telnet-трафік, то він передається на зовні й жодна інша інформація не зберігається. В іншому випадку переходимо до подальшого етапу.
3) Інформація про з'єднання порівнюється із записами в таблиці станів. Якщо такого з'єднання ще немає, воно додається. Якщо воно є, то таймер пасивності скидається.
4) Як тільки новий запис додано, динамічний запис додається на зовнішній інтерфейс для перевірки вхідного трафіку. Він дозволятиме проходження рефлективного трафіку для цього з'єднання. Такий тимчасовий «отвір» залишається відкритим тільки доти, допоки сесія відкрита. Динамічний ACL не зберігається в NVRAM.
5) Коли сесія розривається, динамічна інформація з таблиці станів та динамічний запис видаляються.

Робота CBAC схожа на роботу рефлексивного ACL.

CBAC є гнучким в конфігуруванні, особливо щодо вибору напрямку перевірки. Його можна налаштувати на роботу в обох напрямках.

**Обробка TCP CBAC.** Як тільки TCP-сегмент з бітом SYN потрапляє на маршрутизатор і пропускається вхідним ACL, створюється динамічний запис про нову сесію. Сесія описується адресами й портами кінцевих вузлів, а також номером в послідовності і службовими бітами. Усі подальші пакети, що належать до цієї сесії, перевіряються відповідно до поточного стану і відкидаються, якщо вони невірні. Перевірка здійснюється шляхом аналізу номера в послідовності, кількості байтів, що передаються, та значення службових бітів відповідно до правил роботи TCP-з'єднання. Після встановлення з'єднання всі сегменти обов'язково містять біт ACK.

**Обробка UDP CBAC.** Обробка UDP базується на аналізі тільки адрес і портів кінцевих станцій. Якщо таймер пасивності перебільшує певне значення, інформація про з'єднання видаляється.

**Обробка інших протоколів.** Експертні firewalls, як правило, не підтримують обробку інших протоколів, наприклад, GRE та IPsec, і обробляють їх як звичайні пакетні фільтри. Якщо експертний firewall налаштовано на обробку того чи іншого протоколу, то він робить це аналогічно до обробки UDP. Багато протоколів, зокрема FTP, SQLnet використовують декілька логічних з'єднань під час своєї роботи. Експертний firewall, знаючи цю особливість, після відправки запиту на встановлення з'єднання «підглядає» процедурі встановлення додаткового з'єднання і вносить їх також в таблицю станів. CBAC використовує набір правил для перевірки. Правило перевірки асаціюється з певним інтерфейсом і певним напрямком. Cisco IOS Firewall може розпізнавати команди специфічні для певного протоколу, наприклад, невірні SMTP команди в каналі керування, а також детектувати

та запобігати певним атакам програмного рівня. Якщо firewall визначив факт атаки, він може виконати такі дії:

1) Генерувати повідомлення застереження (Alert).
2) Захистити системні ресурси, які виснажуються під час атаки
3) Блокувати пакети, що утворюють атаку.

Значення timeout та threshold використовуються для керування інформацією про стан з'єднання. Вони дозволяють розривати незавершені з'єднання.

ME CBAC використовує 3 порогових значення для запобігання DoS-атакам типу TCP-flooding:

- Загальна кількість напіввідкритих TCP-сесій.
- Кількість напіввідкритих TCP-сесій на одиницю часу.
- Кількість напіввідкритих TCP-сесій на хост.

Якщо перебільшено загальну кількість напіввідкритих TCP-сесій, ME може виконати такі дії:

- Відправити повідомлення reset на кінцеву точку з найстарішою напіввідкритою сесією для звільнення ресурсів для нових з'єднань.
- Блокувати всі SYN-пакети протягом часу, визначеного тайм-аутом для незавершених сесій. Протягом цього часу нові з'єднання не зможуть встановлюватись.

**Конфігурування CBAC**

1) Відібрати інтерфейси – внутрішній і зовнішній.
2) Сконфігурувати ACLs на інтерфейсі.
3) Визначити правила перевірки.
4) Призначити правила на інтерфейси.

**Відбір інтерфейсів.** У випадку двох інтерфейсів внутрішнім інтерфейсом вважається той, на якому буде ініціюватись сесія.

У випадку трьох інтерфейсів firewall має дозволяти зовнішній трафік до ресурсів в DMZ таких, як DNS та веб-сервіси. Той же firewall може перешкоджати певному трафіку, що входить у внутрішню мережу, якщо він не є рефлексійним у межах установлених сесій. CBAC може бути сконфігурований у двох напрямках на одному або більше інтерфейсів.

**Конфігурування ACL на інтерфейсі.** Для забезпечення ефективного захисту варто дотримуватись таких рекомендацій:

- Починайте з основної конфігурації. Базова початкова конфігурація дозволяє проходження всього трафіку із захищеної мережі в незахищену.
- Насамперед, варто налаштувати базову конфігурацію, яка дозволяє проходження всього трафіку з внутрішньої мережі в зовнішню;
- Дозвольте трафік, який ME буде інспектувати. Він має бути дозволений на інтерфейсах, на які він буде надходити.
- Використовуйте розширені ACL для фільтрації трафіку, що надходить із зовнішньої мережі. Бажано призначати ACL, що аналізують

вхідний трафік (in), оскільки в такому разі менше буде завантажений маршрутизатор.

- Установіть захист від підміни адрес (antispoofing) шляхом заборони будь-якого вхідного трафіку, що надходить на зовнішній інтерфейс з адрес, що збігаються з адресами внутрішньої мережі.
- Забороніть широкомовні повідомлення з адресами відправника 255.255.255.255.
- Хоча за замовчуванням припускається в кінці ACL наявність правила заборони всього, що не дозволено (deny any any), краще це правило задати явно. Це дасть змогу контролювати кількість пакетів, відкинутих ACL.

**Визначення правил перевірки.** Адміністратор має визначити правила для перевірки, що визначатимуть тип протоколу програмного рівня, який перевіряється. Загалом достатньо визначити одне правило. Винятком є випадок, якщо firewall треба налаштувати для роботи в обох напрямках через один інтерфейс. У такому разі створюють два правила – по одному на кожний напрямок. Правила для перевірки мають специфікувати кожний бажаний протокол програмного рівня, або просто TCP, UDP, або ICMP. У другому випадку інспекції підлягають усі активні сеанси відповідних протоколів. ICMP-інспекція дозволяє пропускати echo-відповіді на echo-запити із внутрішньої мережі. Правило інспекції містись набір рядків, кожний з яких містить назву протоколу, ім'я правила, а також опції відправки застережень та записів у журнал аудиту.

```
Router(config)# ip inspect name inspection_name protocol
[alert {on | off}] [audit-trail {on | off}] [timeout seconds]
```

Перелік найбільш популярних протоколів, що можуть інспектуватись наведено в табл. 6.4.

Приклад:

```
Router(config)# ip inspect name RULE1 http alert on audit-
trail on timeout 200
Router(config)# ip inspect name RULE1 ftp alert on audit-
trail on timeout 200
```

У цьому прикладі інспектуються протоколи http та ftp, попередження та аудит увімкнено, timeout становить 200 секунд.

Останній крок конфігурування CBAC – призначення інспекційного правила на інтерфейс.

```
Router(config-if)# ip inspect inspection_name {in | out}
```

Правила інспектування та застосування ACL:

1) На інтерфейс, куди надходить трафік, який буде інспектуватись, призначається ACL, що дозволяє бажаний трафік, та призначаються інспекційні правила в тому ж напрямку.
2) На всіх інших інтерфейсах призначаються ACL на вхідний трафік, які блокують увесь трафік, за винятком трафіку, що не інспектується ME, наприклад, GRE та ICMP.

Таблиця 6.4 – Перелік популярних протоколів для інспектування

| Ключове слово | Протокол |
|---|---|
| icmp | Internet Control Massage Protocol |
| ftp | File Transfer Protocol |
| h323 | H.323 Protocol (for example Microsoft NetMeeting or Intel Video Phone) |
| http | HTTP Protocol |
| dns | DNS Protocol |
| realaudio | Real Audio Protocol |
| rpc | Remote Procedure Call Protocol |
| smtp | Simple Mail Transfer Protocol |
| sqlnet | SQL Net Protocol |
| streamworks | StreamWorks Protocol |
| tcp | Transmission Control Protocol |
| tftp | TFTP Protocol |
| udp | User Datagram Protocol |
| telnet | telnet Protocol |

Наприклад, у мережі, що показана на рис. 6.5, адміністратору треба дозволити внутрішнім користувачам ініціювати TCP, UDP та ICMP з усіма зовнішніми хостами. Зовнішнім клієнтам дозволяється доступ до SMTP-сервера з адресою 194.146.141.3 та HTTP сервера з адресою 194.146.141.4, які розташовані в DMZ. Також необхідно дозволити певні ICMP-повідомлення на всі інтерфейси. Увесь інший зовнішній трафік має блокуватись.

ACL, що дозволяє проходження внутрішнього трафіка назовні:

```
R1(config)# access-list 101 permit tcp 10.10.10.0 0.0.0.255
any
R1(config)# access-list 101 permit udp 10.10.10.0 0.0.0.255
any
R1(config)# access-list 101 permit icmp 10.10.10.0 0.0.0.255
any
R1(config)# access-list 101 deny ip any any
```

Призначаємо створений список на внутрішній інтерфейс:

```
R1(config)# interface Fa0/0
R1(config-if)# ip access-group 101 in
```

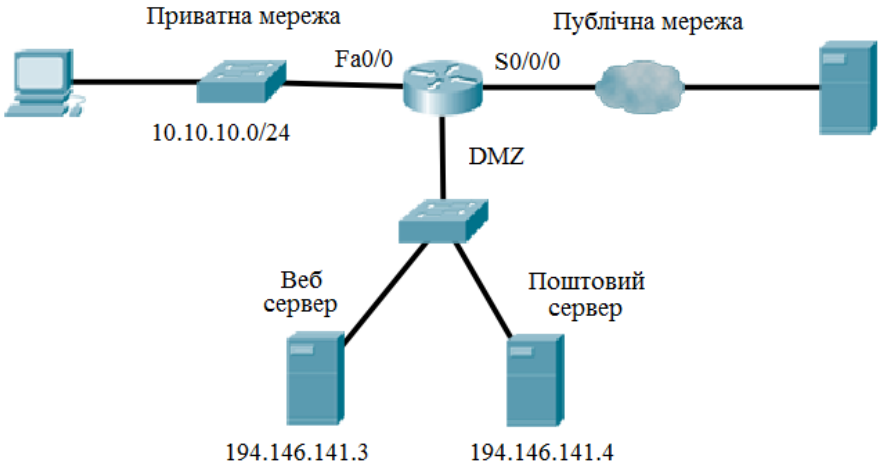


Рисунок 6.5 – Схема мережі

Створюємо розширений ACL для дозволу проходження SMTP, HTTP та службового ICMP-трафіку в DMZ-зону та блокування іншого трафіку:

```
R1(config)# access-list 102 permit tcp any host 209.165.201.1 eq www
R1(config)# access-list 102 permit tcp any host 209.165.201.2 eq smtp
R1(config)# access-list 102 permit icmp any any echo-reply
R1(config)# access-list 102 permit icmp any any unreachable
R1(config)# access-list 102 permit icmp any any administratively-prohibited
R1(config)# access-list 102 permit icmp any any packet-too-big
R1(config)# access-list 102 permit icmp any any echo
R1(config)# access-list 102 permit icmp any any time-exceeded
R1(config)# access-list 102 deny ip any any
```

Цей ACL призначаємо на інтерфейс, що дивиться у зовнішню мережу:

```
R1(config)# interface S0/0/0
R1(config-if)# ip access-group 102 in
```

Створюємо правила для інспектування TCP та UDP-трафіку:

```
R1(config)# ip inspect name RULE tcp
R1(config)# ip inspect name RULE udp
```

Ці правила призначаємо на на внутрішній інтерфейс:

```
R1(config)# interface Fa0/0
R1(config-if)# ip inspect RULE in
```

Інспекційний лист автоматично створить тимчасовий ACL, призначений до зовнішнього інтерфейсу для TCP та UDP-зв'язків. Це дозволить проходження рефлексивного трафіку. Для видалення CBAC з маршрутизатора: `Router(config)# no ip inspect`. Ця команда видаляє всі CBAC команди, таблицю станів, усі тимчасові ACL, скидає всі порогові значення та тайм-аути до значень за замовчуванням.

**Моніторинг роботи ME CBAC**. CBAC-інспектування підтримує два типи лог-функцій: попередження та аудит.

**Попередження** (Alerts) виводять повідомлення, що стосуються функціонування CBAC, наприклад, інформацію про недостатню кількість ресурсів, DoS-атаки та інші загрози. Попередження за замовчування увімкнені й автоматично виводяться на консоль адміністратора. Їх можна вимкнути, хоча вкрай небажано: `Router(config)#ip inspect alert-of`.

Існує можливість вимкнути попередження для окремих правил, хоча це теж не рекомендують робити.

**Аудит.** Аудит реєструє інформацію про зв'язки, що CBAC інспектує, водночас вдалі та невдалі спроби. Виводить повідомлення, коли CBAC додає або видаляє записи з таблиці станів (табл. 6.5). За замовчуванням аудит вимкнено. Для увімкнення команда `Router(config)# ip inspect audit-trail`. За замовчуванням і попередження, і аудит виводять інформацію на консоль. Можна сконфігурувати виведення на Syslog-сервер:

```
R1(config)# logging on
R1(config)# logging host 10.0.0.3
R1(config)# ip inspect audit-trail
R1(config)# no ip inspect alert-off
```

Таблиця 6.5 – Перегляд поточної інформації **show ip inspect parameters**

| Параметр | Пояснення |
|---|---|
| **name** inspection_name | Обмежує виведення на екран інспекційний набір правил, який був зазначений |
| **config** | Виводить повну CBAC конфігурацію інспекції на роутері |
| **interfaces** | Виводить правила інспекції, що активовані на інтерфейсі роутера |
| **sessions** | Виводить всі з'єднання в таблиці станів CBAC |
| **sessions** [detail] | Виводить всі деталі з'єднання в таблиці станів CBAC |
| **all** | Displays all the information from the options listed in this table |

### 6.2.2 ME, що базується на зонах

Більш гнучка модель налаштування політик безпеки реалізована в ME, що базується на понятті зони (zone-based firewall, ZBF). Відповідно до цієї моделі, інтерфейси асоціюються з певними зонами, а потім інспекційна політика прикладається до трафіку, що передається між зонами. ZBF дозволяє різні інспекційні політики призначати до комп'ютерних груп, під'єднаних до одного інтерфейсу. Це також дає можливість заборонити трафік за допомогою політик за замовчуванням (default) – політики блокування всього трафіку (deny-all) між зонами ME.

Політики інспектування, реалізовані в ZBF, підтримують усі попередньо визначені функції ME, у тому числі інспекцію TCP, UDP-протоколів, інспекцію на рівні додатків, URL-фільтрацію, запобігання DoS-атакам тощо. Застосування ZBF спрощує механізм захисту і робить його структурно-орієнтованим.

Використання CBAC ME є досить складним унаслідок відсутності можливості застосування ієрархічних структур. Зокрема, CBAC має такі основні обмеження:

- велика кількість інспекційних політик та ACL на кількох інтерфейсах маршрутизатора досить сильно ускладнює процедуру визначення кореляції політик для аналізу трафіку між кількома інтерфейсами;
- політики не можуть бути прив'язані до груп хостів або підмереж за допомогою ACL, увесь трафік, що проходить через певний інтерфейс, підпадає під одну політику інспектування;
- процес налаштування та функціонування CBAC фактично жорстко прив'язаний до списків керування доступом.

При застосуванні ZBF-зони встановлюють кордони безпеки мережі. Зони самі по собі визначають межі, де трафік підпадає під обмеження політики коли проходить в інший регіон мережі. За замовчуванням політика між зонами – усе блокувати. Якщо жодна політика не сконфігурована між зонами, увесь трафік буде блокуватись. Це суттєва відмінність від CBAC, де трафік за замовчуванням пропускається доти, допоки він не буде заблокований ACL. Основні переваги ZBF: не залежать від ACL; за замовчуванням здійснюється блокування всього трафіку; одна політика впливає на будь-який трафік, замість використання багатьох ACL та інспекційних дій.

Варто звернути увагу, що обидва підходи можуть застосовуватись спільно, однак дві моделі не можуть одночасно використовуватись на одному інтерфейсі, наприклад: інтерфейс не може бути сконфігурований як член зони і для IP-інспекції одночасно.

Проектування зонних політик передбачає кілька етапів:

1) Визначення зон. Уся мережа поділяється на зони з різними рівнями безпеки. На цьому етапі не здійснюється вибір обладнання, кількість пристроїв тощо.
2) Визначення політик для зон – для кожної пари «відправник-отримувач» визначаються сесії, які клієнти в зоні відправника можуть запитати в зоні отримувача, наприклад, для TCP, UDP або ICMP-трафіку. Цей етап, як і попередній, не передбачає ніяких фізичних дій.
3) Проектування фізичної інфраструктури з урахуванням вимог захищеності й доступності. Він охоплює визначення кількості пристроїв між найбільш захищеними і найменш захищеними зонами та визначення надлишкових пристроїв.
4) Ідентифікація підмножин зон та об'єднання вимог до трафіку. Для кожного ME адміністратор має ідентифікувати підмножини зон, під'єднаних до його інтерфейсів і об'єднати вимоги до трафіку для цих зон.

ZBF ME може бути сконфігурований на виконання трьох можливих дій. **Інспектування** (Inspect) – дія, еквівалентна до команди ip inspect в CBAC ME. У цьому випадку автоматично дозволяється проходження рефлективного трафіку. Також коректно обслуговуються протоколи, що використовують кілька TCP-з'єднань, наприклад, FTP. **Блокування** (Drop) – аналог команди deny в ACL, блокує увесь трафік. **Дозвіл** (Pass) – аналог команди permit в ACL. Ця дія не відслідковує стан з'єднання або сесії. Дозволяється проходження трафіку тільки в одному напрямку. У такому разі відповідна політика має бути застосована для того, щоб дозволити проходження трафіку у зворотному напрямку. При конфігуруванні членства інтерфейсу маршрутизатора в тій чи іншій зоні варто враховувати такі правила:

- зона має бути створена перед тим, як адміністратор зможе асоціювати з нею інтерфейси;
- якщо в передаванні трафіку беруть участь усі інтерфейси маршрутизатора, кожний інтерфейс має бути членом однієї із зон;
- інтерфейс може бути асоційованим тільки з однією зоною;
- за замочуванням трафіку дозволяється передаватись між інтерфейсами, що є членами однієї зони;
- для того, щоб дозволити проходження трафіку в зону або із зони, членом якої є інтерфейс, має бути сконфігурована політика дозволу або інспекції трафіку між цією зоною та іншою зоною;

- трафік не може передаватись між інтерфейсом, що є членом зони й іншим інтерфейсом, який не входить до жодної із зон. Дії (інспектування, блокування та дозвіл) можуть призначатись тільки між зонами;
- інтерфейси, які не призначені в зону, можуть використовувати CBAC-інспекцію;
- якщо є необхідність, щоб інтерфейс маршрутизатора не брав участі в зонній політиці і пропускав увесь трафік, цей інтерфейс можна додати в зону й сконфігурувати політику, що дозволятиме проходження всього трафіку між цією зоною та будь-якою іншою зоною.

Правила обробки трафіку при застосуванні ZBF ME визначається табл. 6.6. Коли інтерфейс сконфігурований як член зони, хости, що під'єднані до нього, також входять у зону. Але трафік між інтерфейсом маршрутизатора й хостами не контролюється зонною політикою. Усі інтерфейси маршрутизатора автоматично є членами, так званої, власної зони маршрутизатора (self zone). Тому для обмеження трафіку, що спрямовується на IP-адреси інтерфейсів маршрутизатора, з різних зон мають бути застосовані відповідні політики. Політики можуть бути налаштовані на блокування, дозвіл та інспектування трафіку між певною зоною та власною зоною маршрутизатора. Якщо політик між певною зоною й власною зоною немає, дозволяється проходження усього трафіку на інтерфейс без інспектування (табл. 6.7).

Таблиця 6.6 – Правила обробки трафіка при застосуванні ZBF ME

| Чи є інтерфейс відправника членом зони? | Чи є інтерфейс одержувача членом зони? | Чи існує зонна пара? | Чи існує політика між зонами? | Результат |
|---|---|---|---|---|
| Ні | Ні | – | – | Увесь трафік пропускається |
| Так (зона 1) | Так (зона 1) | – | – | Увесь трафік пропускається |
| Так | Ні | – | – | Увесь трафік блокується |
| Ні | Так | – | – | Увесь трафік блокується |
| Так (зона 1) | Так (зона 2) | Ні | – | Увесь трафік блокується |
| Так (зона 1) | Так (зона 2) | Так | Ні | Увесь трафік блокується |
| Так (зона 1) | Так (зона 2) | Так | Так | Дія визначається політикою |

Таблиця 6.7 – Правила обробки трафіку політиками

| Чи є інтерфейс відправника членом зони? | Чи є інтерфейс отримувача членом зони? | Чи існує зонна пара? | Чи існує політика між зонами? | Результат |
|---|---|---|---|---|
| Маршрутизатор | Так | Ні | – | Увесь трафік пропускається |
| Маршрутизатор | Так | Так | Ні | Увесь трафік пропускається |
| Маршрутизатор | Так | Так | Так | Дія визначається політикою |
| Так | Маршрутизатор | Ні | – | Увесь трафік пропускається |
| Так | Маршрутизатор | Так | Ні | Увесь трафік пропускається |

**Конфігурування ZBF**

Для конфігурування ZBF ME необхідно:

1) Створити зони за допомогою команди zone security.
2) Визначити класи трафіку за допомогою команди class-map type inspect.
3) Визначити політики за допомогою команди policy-map type inspect.
4) Призначити політики до пар зон відправника та отримувача використовуючи команду zone-pair security.
5) Асоціювати інтерфейси маршрутизатора із зонами використовуючи команду zone-member security interface.

При конфігуруванні ZPF треба враховувати кілька факторів:

- Зона має бути сконфігурована перш ніж вона буде асоціюватись з інтерфейсом.
- Інтерфейс не може належати до кількох зон.
- ZBF може співіснувати з CBAC. Команда ip inspect може використовуватись тільки на інтерфейсах, які не є членами зон безпеки.
- Трафік ніколи не буде передаватись між інтерфейсами, асоційованими із зоною та інтерфейсами, що не асоційовані із зонами.
- За замовчуванням міжзонна політика блокує увесь трафік доти, допоки не буде явно визначена політика між зонами.
- Маршрутизатор ніколи не фільтрує трафік між інтерфейсами всередині зони.
- Членство у зонах не захищає сам маршрутизатор, оскільки трафік до та від маршрутизатора не фільтрується до тих пір, доки не буде створено пари, що утворюються власною зоною (self-zone).

**Створення зон**

```
Router(config)# zone security zone-name
Router(config-sec-zone)# description line-of-description
```

При створенні зони треба думати, які інтерфейси будуть до неї входити. Як правило до однієї зони входять інтерфейси, що мають спільні вимоги, з погляду безпеки.

**Визначення класів трафіку**

Класи трафіку дозволяють визначати потоки даних, щоб далі оперувати з ними як з єдиним цілим.

```
Router(config)# class-map type inspect [match-any | match-all] class-map-name
```

Для 3-го та 4-го рівнів match-any визначає поведінку за замовчуванням.

```
Router(config)# class-map type inspect protocol-name [match-any | match-all] class-map-name
```

Для 7-го рівня, використовується class maps специфічна для того чи іншого протоколу рівня додатків. Для додавання трафіку в карту може використовуватись ACL за допомогою команди:

```
Router(config-cmap)# match access-group {access-group | name access-group-name}
```

Для додавання протоколу:

```
Router(config-cmap)# match protocol protocol-name
```

**Визначення політик.** Політики визначають, що робити з тим чи іншим класом трафіку (відкидати, пропускати або інспектувати):

```
Router(config)# policy-map type inspect policy-map-name
```

Класи трафіку, на які дія буде розповсюджуватись визначаються за допомогою class type inspect:

```
Router(config-pmap)# class type inspect class-name
```

На подальшому етапі вказується дія для певного класу трафіку:

```
Router(config-pmap-c)# pass | inspect | drop [log]| police
```

**Застосування політик.** Після створення політик вони призначаються для трафіку, що передається між парою зон. Для призначення політики спочатку треба створити зонну пару. Далі треба визначити зону відправника, зону отримувач та політику, що керує трафіком між ними:

```
Router(config)# zone-pair security zone-pair-name [source source-zone-name | self] destination [self | destination-zone-name]
```

Команда `service-policy type inspect policy-map-name` використовується для приєднання карти політики до зонної пари. Інспекція на 7-му рівні здійснюється за допомогою команди:

```
Router(config-pmap-c)# service-policy {h323 | http | im | imap | p2p | pop3 | sip | smtp | sunrpc | urlfilter} policy-map
```

**Асоціювання інтерфейсів з відповідною зоною.** Останній етап налаштування – це прив’язка інтерфейсу до певної зони:

```
Router(config-if)# zone-member security zone-name
```

Розглянемо приклад налаштування ZBF для схеми, що наведено на рис. 6.6.

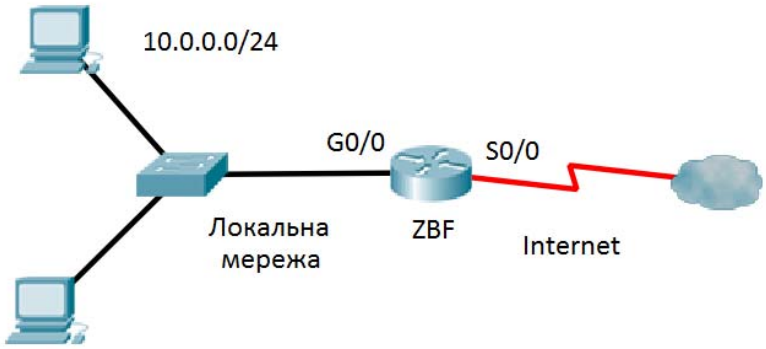


Рисунок 6.6 – Схема мережі

1. Створюємо зони:

```
ZBF(config)# zone security Internal
ZBF(config-sec-zone)# description Internal network
ZBF(config)# zone security External
ZBF(config-sec-zone)# description External network
```

2. Визначаємо клас трафіку за допомогою розширеного ACL. У такому разі це буде весь трафік, що передається з локальної мережі:

```
ZBF(config)# class-map type inspect USERS_TRAFFIC
ZBF(config-cmap)# match access-group 101
ZBF(config-cmap)# exit
ZBF(config)# access-list 101 permit ip 10.0.0.0 0.0.0.255 any
```

3. Визначаємо політику, що буде застосовуватись до вибраного класу трафіку. У такому разі це буде інспектування:

```
ZBF(config)# policy-map type inspect Internal-To-External
ZBF(config-pmap)# class type inspect USERS_TRAFFIC
ZBF(config-pmap-c)# inspect
```

4. Призначаємо політику до пари зон відправника та отримувача:

```
ZBF(config)# zone-pair security Internal-External source In-
ternal destination External
ZBF(config-sec-zone-pair)# description Internet Access
ZBF(config-sec-zone-pair)# service-policy type inspect Inter-
nal-To-External
```

5. Асоціюємо інтерфейси маршрутизатора із зонами:

```
ZBF(config)# interface G0/0
ZBF(config-if)# zone-member security Internal
ZBF(config-if)# interface S0/0
ZBF(config-if)# zone-member security External
```

# 7 ТЕХНОЛОГІЇ ЗАХИСТУ МЕРЕЖЕВИХ ПРИСТРОЇВ

## 7.1 Захист доступу до пристроїв

### 7.1.1. Захист прикордонних маршрутизаторів

Захист мережевої інфраструктури є критичним для захисту мережі у цілому. До складу мережевої інфраструктури входять: маршрутизатори, комутатори, сервери та інші пристрої. Якщо зловмисник отримав доступ до маршрутизатора, то безпека і керування всією мережею буде під загрозою. Дуже важливим є те, щоб відповідна політика безпеки та контрольні заходи постійно застосовувались для запобігання неавторизованого доступу до всіх пристроїв інфраструктури. Хоча небезпека загрожує всім пристроям, маршрутизатори є найпривабливішими для атакуючих, оскільки через них проходить увесь мережевий трафік. У багатьох організаціях використовують прикордонний маршрутизатор – останній маршрутизатор між внутрішньою та зовнішньою мережею, наприклад Internet. Виділяють три аспекти захисту маршрутизаторів. **Фізична безпека:** пристрої мають бути розташовані в кімнаті з обмеженим доступом, має бути забезпечена відсутність електромагнітних завад, наявність джерела безперебійного живлення, контроль температурного режиму та вологості, протипожежний захист, наявність запасних частин тощо. **Безпека на рівні операційної системи** передбачає встановлення максимальної кількості оперативної пам’яті, що дозволить захиститись від деяких DoS-атак та встановити різноманітні сервіси захисту. Варто використовувати останню стабільну версію операційної системи, періодично здійснювати резервне копіювання файлів ОС та конфігураційних файлів. **Захист на рівні самого маршрутизатора**; вимкнути сервіси, що є потенційно небезпечними або не використовуються; організувати захищений адміністративний доступ з використанням облікових записів та наданням необхідного рівня привілей; вимкнути порти й інтерфейси, що не використовуються.

Обмеження адміністративного доступу передбачає такі дії:

- Визначити перелік адрес, з яких дозволено адміністративний доступ та інтерфейсів (портів), через які це можна зробити.
- Забезпечити 100 % аудит для всіх випадків адміністративного доступу з фіксацією подій і часу.
- Забезпечити 100 % аутентифікацію та авторизацію адміністративного доступу з обмеженням дозволеної кількості невірних спроб.
- Використовувати офіційні попередження при доступі до пристрою відповідно до політики безпеки компанії.
- Забезпечити конфіденційність даних, що зберігаються локально та передаються через канали зв’язку.

Існують два способи адміністративного доступу: локальний та віддалений. Локальний доступ здійснюється через, так званий, консольний порт і передбачає безпосередню присутність поряд з обладнанням. Віддалений

доступ здійснюється через мережу і потребує додаткових заходів безпеки. Для захисту віддаленого доступу необхідно дотримуватись таких правил:

- Шифрувати весь трафік адміністративного сеансу, використовуючи SSH або HTTPS.
- Установити окремий виділений канал для адміністративного доступу, через який за допомогою фільтрів буде дозволено під'єднуватись тільки окремим хостам за певними протоколами.

### 7.1.2 Конфігурування захищеного адміністративного доступу

Для захисту адміністративного доступу до мережевого обладнання використовуються паролі. Паролі можуть бути встановлені на консольний доступ (console), доступ у режимі віддаленого терміналу (vty), доступ через допоміжний порт через dial-up з'єднання (aux), доступ до привілейованого режиму (enable). Одним із найважливіших елементів захисту є використання паролів, які важко підібрати зловмиснику. Для цього варто дотримуватись таких рекомендацій:

- Обмежити мінімальну довжину пароля.
- Уникати використання звичайних слів, послідовностей однакових літер або цифр, імені користувача, імен родичів та домашніх тварин, біографічної інформації: день народження, імена батьків та іншої персональної інформації.
- Можна робити свідомі помилки та замінники, наприклад: Smith записати у вигляді Smyth або 5mYth, Security у вигляді 5ekur1ti.
- Періодично змінювати пароль.
- У жодному разі не записувати пароль, тим більше залишати цей запис у загальнодоступному місці.

У корпоративній мережі може бути багато пристроїв, що вимагають захищеного доступу. У такому випадку випадку для керування паролями доцільно використовувати спеціальні централізовані TACACS+ або RADIUS-сервери. Якщо мережу адмініструють кілька адміністраторів, то для кожного з них варто налаштувати окремий пароль доступу до пристрою з відповідним рівнем привілеїв та окремо налаштувати пароль на привілейований режим. Локальну базу облікових записів доцільно використовувати тільки у разі проблем з доступом до серверу аутентифікації.

**Мінімальна довжина пароля.** Адміністратор може встановити мінімальну довжину для всіх паролів від 0 до 16 символів (рекомендують 10) за допомогою команди:

```
Router(config)#security passwords min-length lengt
```

Команда впливає на всі паролі, які будуть вводитись після неї, на паролі, що вже встановлені вона не вплине.

**Розрив пасивного з'єднання.** Може виникнути ситуація, коли адміністратор залишив комп'ютер з не закритою адміністративною сесію, що створює серйозну небезпеку. За замовчуванням адміністративний інтерфейс залишається активним протягом 10 хвилин після останньої активнос-

ті, далі розривається. Доцільно зменшити це значення до 2–3 хвилин. Для налаштування варто перейти на відповідну лінію (console або vty) і встановити відповідне обмеження за допомогою команди:

```
Router(config-line)#exec-timeout  timeout
```

На тих лініях, що не використовуються, наприклад aux, можна взагалі вимкнути exec процес за допомогою команди:

```
Router(config-line)#no exec
```

**Шифрування всіх паролів.** Команда `service password-encryption` у режимі глобального конфігурування здійснює шифрування всіх паролів, що зберігаються в конфігураційному файлі у відкритому вигляді. Однак алгоритм шифрування досить простий, тому є сенс її застосовувати, коли нема інших варіантів це зробити.

Для шифрування пароля на привілейований доступ доцільно використовувати команду `enable secret` замість `enable password`, а для шифрування особистого паролю адміністратора команду `username name secret` замість `username name password`. Варто звернути увагу, що в такому випадку йдеться про локальну базу користувачів, що зберігається безпосередньо на пристрої. Для використання локальної бази даних необхідно дати команду:

```
Router(config-line)#login local
```

### 7.1.3 Конфігурування покращеного захисту для VTY

Призначення паролів і локальна аутентифікація не перешкоджає проведенню DoS-атак у вигляді великої кількості запитів на реєстрацію. Аналогічна ситуація можлива у разі словникової атаки. При ввімкненні профіля детектування пристрій може бути сконфігуровано на певну реакцію на невірні спроби шляхом ігнорування подальших запитів (`login blocking`). Це блокування може бути сконфігуровано на певний час, який називається «періодом тиші» (`quiet period`). Легітимні спроби можуть бути дозволені протягом цього періоду за допомогою спеціального ACL, де вказуються легітимні адреси.

Для налаштування цього механізму необхідно вказати такі параметри:

1. Затримка між успішними спробами реєстрації:

```
Router(config)# login delay seconds
```

2. Блокування процесу реєстрації за підозри DoS-атаки – указується тривалість блокування при заданій кількості невдалих спроб протягом визначеного періоду:

```
Router(config)# login block-for seconds attempts tries within seconds
```

3. Генерація системних повідомлень:

```
Router(config)# login on-failure log [every login]
Router(config)# login on-success log [every login]
```

4. Адреси, з яких дозволено реєстрацію протягом «періоду тиші»:

```
Router(config)# login quiet-mode access-class {acl-name | acl-number}
```

Варто звернути увагу, що ці налаштування стосуються тільки віддалених з'єднань і не розповсюджуються на консольні підключення. Крім того, ці опції працюють тільки у випадку, коли налаштовано віддалену реєстрацію з використанням імені й пароля, якщо реєстрація відбувається тільки на основі пароля, то зазначені механізми захисту не працюють. За замовчуванням усі опції, пов'язані з детектуванням атаки, заблоковані. Для їхньої активації використовується команда *login block-for*:

```
R1(config)# login block-for 120 attempts 5 within 60
```

Ця опція керує процесом реєстрації та працює в двох режимах:

1) Нормальний режим (режим спостереження) – маршрутизатор підраховує кількість невдалих спроб реєстрації (у наведеному прикладі 5) протягом визначеного інтервалу часу (60 секунд).

2) Режим спокою (період спокою) – якщо кількість невдалих спроб перебільшує вказаний поріг, усі спроби реєстрації з використанням Telnet, SSH, та HTTP ігноруються протягом визначеного часу (у наведеному прикладі 120 секунд). У цей період усі спроби реєстрації, у тому числі легітимні не дозволяються. Однак для забезпечення доступу адміністратора це правило може бути порушено для пристроїв, визначених спеціальним ACL. ACL має бути створений та ідентифікований за допомогою відповідної команди:

```
R1(config)# ip access-list standard PERMIT-ADMIN
R1(config-std-nacl)# remark Permit only Administrative hosts
R1(config-std-nacl)# permit 192.168.10.10
R1(config-std-nac1)# permit 192.168.11.10
R1(config-std-nacl)# exit
R1(config)# login quiet-mode accent-class PERMIT-ADMIN
```

За замовчуванням маршрутизатор може обробляти запити на під'єднання так швидко, як швидко вони надходять. Це робить пристрої незахищеними до словникових атак, які можуть здійснювати до 1000 спроб за 1 секунду. Команда **`login block-for`** вводить автоматичну затримку 1 секунду між спробами, що суттєво сповільнює процедуру підбору пароля. Значення затримки може бути змінено командою **`login delay`**:

```
R1(config)# login delay 3
```

Указані команди допомагають блокувати невдалі спроби протягом певного періоду, але не заважають атакуючому робити це знову. Команда **`auto secure`** вмикає механізм реєстрації невдалих спроб під'єднання до маршрутизатора. Вдалі спроби за замовчуванням не реєструються. Існує можливість увімкнути реєстрацію вдалих спроб також:

```
R1(config)# login on-success log [every login]
R1(config)# login on-failure log [every login]
R1(config)# exit
```

або

```
R1(config)# security authentication failure rate 10 thresh-
old—rare log
```

Кількість спроб реєстрації перед тим, як лог-повідомлення буде генеруватись може бути визначено параметром **`every login.`** За замовчуванням 1 спроба, можливий діапазон: 1 – 65535. Як альтернатива, команда **`security authentication failure rate threshold-rate`**, що генерує відпо-

відне повідомлення, коли кількість невдалих спроб за одиницю часу перебільшує задане порогове значення. Приклад налаштування механізмів захисту віддаленого доступу наведено нижче:

```
R1(config)# username  ADMIN secret cisco54321
R1(config)# line vty 0 4
R1(config-line)# login local
R1(config-line)# exit
R1(config)# login block-for 15 attempts 5 within 60
R1(config)# ip access-list standard PERMIT-ADMIN
R1(config-std-nacl)# remark Permit only Administrative hosts
R1(config-std-nacl)# permit 192.168.10.10
R1(config-std-nacl)# permit 192.168.11.10
R1(config-std-nacl)# exit
R1(config)# login quiet-mode access-class PERMIT-ADMIN
R1(config)# login delay 10
R1(config)# login on-success log
R1(config)# login on-failure log
R1(config)# exit
```

Для перевірки конфігурації команди **`login block-for`** та поточного стану маршрутизатора використовується команда **`show login [failures]`**, що виводить, зокрема IP-адресу, з якої робились невдалі спроби:

```
R1# show login failures
Total failed logins: 22
Detailed information about last 50 failures
Username SourceIPAddr  1Port  Count       Timestamp
Admin  1.1.2.1         23    5     15:38:54 UTC Wed Dec 10 2008
Admin  10.10.10.10     23    13    15:58:43 UTC Med Dec 10 2008
Admin  10.10.10.111    23    3     15:57:14 UTC Wed Dec 10 2008
cisco  10.210.10.10    23    1     15:57:21 UTC fled Dec 10 2008
```

Потрібно також використовувати повідомлення, що будуть виводитись на екрані потенційного зловмисника і попереджувати його, це потрібно для вдалого проведення судового процесу у випадку взлому мережі. Треба дуже обережно вибирати текст для баннера, і в жодному разі не використовувати різноманітні привітання:

```
R1(config)# banner {exec | incoming | login | motd | slip-
ppp} d message
```

### 7.1.4 Конфігурування SSH

Для віддаленого адміністративного з'єднання традиційно використовувалась утиліта Telnet, що працює через 23-й TCP-порт. Її основною перевагою є загальна поширеність та підтримка практично всім обладнанням. Однак при використанні Telnet-трафік передається у відкритому вигляді і зловмисник може перехопити пароль. Захищеною альтернативою Telnet є утиліта SSH, що працює через 22-й TCP-порт і шифрує трафік. Перш ніж конфігурувати SSH необхідно зробити такі речі:

1) Впевнитись, що на маршрутизаторі встановлена версія IOS не менше ніж 12.1(1)T. Саме з цієї версії підтримується IPsec DES та Triple Data Encryption Standard (3DES). Ознакою того, що образ ОС підтримує відповідне шифрування є k8 або k9 в імені файлу IOS, наприклад, c1841-advipservicesk9-mz.124-10b.bin

2) Упевнитись, що кожний маршрутизатор, до якого планується SSH-під'єднання, має унікальне ім'я.

3) Упевнитись, що кожний маршрутизатор, до якого планується SSH-під'єднання, використовує коректне DNS-ім'я.

4) Упевнитись, що на маршрутизаторі налаштовано локальну аутентифікацію або AAA-сервіс з використанням імені користувача й пароля.

Етапи конфігурування SSH: 1) Якщо маршрутизатор має ім'я *Router*, змінити його на інше, яке буде унікальним та інформативним, тобто міститиме інформацію про маршрутизатор. 2) Сконфігурувати доменне ім'я за допомогою команди: `R1(config)#ip domain-name <domain-name>`. 3) Згенерувати пару ключів для шифрування SSH-трафіку. Cisco IOS використовує RSA-алгоритм. Для генерації ключів використовується команда:

```
R1(config)#crypto key generate rsa general-keys modulus <modulus-size>
```

Розмір модуля може бути від 360 до 2048 бітів. Що довший модуль, більшою є захищеність, однак за великої довжини модуля довго генерується ключ і багато часу витрачається на саму процедуру шифрування. Мінімальна довжина, яка рекомендується – 1024 біти. Для перевірки SSH та перегляду сконфігурованих ключів використовується команда: **`show crypto key mypubkey rsa`**. Якщо пара ключів уже існує, рекомендують перегенерувати за допомогою команди **`crypto key zeroize rsa`**.

1) Упевнитись, що в локальній базі даних є обліковий запис для користувача, що буде під'єднуватись через SSH-сервіс. За необхідності створити його за допомогою команди:

```
R1(config)#username <name> secret <password>
```

2) Налаштувати SSH-сесію, використовуючи команди:

```
R1(config-line)#login local
R1(config-line)#transport input ssh
```

SSH-сервіс автоматично стартує після генерації RSA-ключів.

**Додаткові команди налаштування SSH.** Додатково на маршрутизаторі можна налаштувати версію SSH, тривалість очікування відповіді клієнта (timeout) та кількість повторних спроб аутентифікації. Cisco маршрутизатори підтримують 2 версії протоколу SSH: SSHv1 та SSHv2. SSHv2 є більш захищеною за рахунок використання обміну ключами за допомогою алгоритму Diffie-Hellman та контролю цілісності за допомогою message authentication code (MAC). Для переходу з однієї версії на іншу є команда `R1(config)#ip ssh version {1 | 2}`. Тривалість часу, протягом якого чекатиме маршрутизатор на відповідь клієнта під час фази реєстрації за замовчуванням становить 120 секунд. Для змінення цього значення варто скористатись командою `R1(config)#ip ssh time-out <seconds>`. За замовчуванням користувач має 3 спроби реєстрації перш ніж він буде позбавлений доступу. Для конфігурування іншого значення команда `R1(config)# ip ssh authentication-retries <integer>`. Для перевірки SSH-конфігурації є команда **`show ip ssh`**.

## 7.2 Гранулювання прав адміністратора в Cisco IOS

Важливим елементом підвищення захищеності маршрутизатора є визначення різних рівнів доступу та адміністративних можливостей для різних користувачів. Cisco IOS дозволяє реалізувати це через механізм привілей (Privilege level) та механізм адміністративних ролей (Role-Based).

### 7.2.1 Конфігурування рівнів привілеїв

За замовчуванням Cisco IOS забезпечує два рівня доступу до команд:

- Користувацький режим (user EXEC mode), що відповідає найнижчому рівню привілеїв (privilege level 1) та забезпечує мінімальний набір прав. Фактично в цьому режимі є можливим тільки перегляд стану пристрою. Ознакою доступу до цього режиму є позначка «>» після імені маршрутизатора в рядку запрошення.
- Привілейований режим (Privileged EXEC mode) відповідає найвищому рівню привілеїв (privilege level 15) та забезпечує виконання всіх команд без винятку. Ознакою доступу до цьому режиму є позначка «#» після імені маршрутизатора в рядку запрошення.

### 7.2.2 Призначення рівнів привілеїв

Адміністратор має можливість конфігурувати кілька рівнів привілеїв. Загалом існує 16 рівнів привілеїв, причому рівні 0, 1 та 15 мають визначенні налаштування. Адміністратор може додатково визначити кілька рівнів привілеїв і призначити різні команди на кожний рівень. Базовим правилом функціонування механізму привілеїв є те, що команда, розташована на k-му рівні, доступна на всіх рівнях, починаючи з k і, відповідно, відсутня на всіх рівнях до k. Тобто вищий рівень привілей підтримує більше команд, наприклад, усі команди рівня 7 підтримуються на рівнях 8, 9 тощо. Механізм функціонування привілей продемонстровано на рис. 7.1. За замовчуванням рівні з 2-го по 14-й є порожніми.

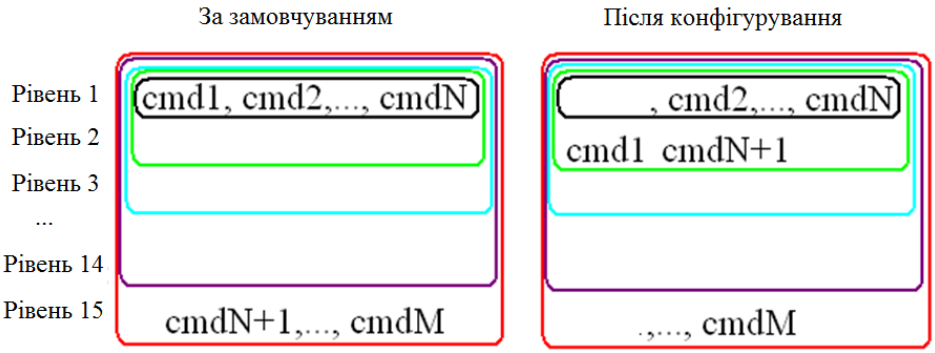


Рисунок 7.1 – Механізм функціонування рівнів привілеїв

При переміщенні команди cmd1 з першого рівня на другий вона зникає з переліку команд рівня 1. При переміщенні команди cmdN+1 з 15-го рівня на другий –вона стає доступною для всіх рівнів, починаючи з другого.

Для призначення команд на певний рівень використовується команда:

```
Router(config)# privilege <mode> {level <level command> | reset}
```

Варто зазначити, що, наприклад, призначення команди *show ip route* автоматично призначає всі команди *show* та *show ip*. Доступ до певного рівня привілеїв має забезпечуватись через механізм аутентифікації. Існують два методи керування доступом до рівня привілеїв:

1. За допомогою пароля на доступ до рівня, який задається командою:
   ```
   Router(config)# enable secret <level> <level password>
   ```
2. Шляхом асоціювання користувача з тим чи іншим рівнем привілеїв:
   ```
   Router(config)# username <name> privilege <level> secret <password>
   ```

Наприклад, адміністратор хоче призначити чотири рівня доступу до пристроїв:

- Користувач USER повинен мати можливість ініціювати всі команди рівня 1, за винятком команди ping.
- Користувач SUPPORT повинен мати можливість ініціювати всі команди рівня 1, а також команду ping.
- Користувач JR-ADMIN повинен мати можливість ініціювати всі команди попередніх рівнів, а також команду reload.
- Користувач ADMIN повинен мати можливість ініціювати всі команди.

Привілеї USER збігаються з привілеями 1-го рівня, тому налаштовувати додатково нічого не треба. Користувачу SUPPORT буде призначений 5-й рівень доступу. Він автоматично успадковує всі команди з рівнів 1 до 4, а також можуть бути призначені додаткові команди. Як було зазначено, якщо команду призначено на певному рівні, доступ до цієї команди втрачається на всіх нижніх рівнях. Команду *ping* можна перевести на 5-й рівень таким чином: `Router(config)# privilege exec level 5 ping`.

Після цього USER втрачає можливість її використовувати, оскільки тепер потрібні привілеї 5-го рівня для запуску цієї команди.

Для призначення пароля на 5-й рівень використовується команда:

```
Router(config)# enable secret level 5 cisco5.
```

Для призначення спеціального імені SUPPORT для 5-го рівня привілеїв використовується команда:

```
Router(config)# username support privilege 5 secret cisco5
```

Користувач, що зареєструвався під іменем SUPPORT, може отримати доступ тільки до 5-го рівня привілеїв й успадковує всі привілеї 1-го рівня. JR-ADMIN потребує доступу до команд всіх рівнів з 1 по 5 та команду *reload*. Тому він повинен мати рівень більший за 5-й, наприклад 10. Для

призначення команди *reload* на 10-й рівень, розташування JR-ADMIN на 10-му рівні та призначення відповідних паролів варто виконати такі команди:

```
Router(config)# privilege exec level 10 reload
Router(config)# username jr-admin privilege 10 secret cisco10
Router(config)# enable secret level 10 cisco10.
```

Для призначення прав адміністратору достатньо задати пароль на 15-й рівень та асоціювати користувача *admin* з цим рівнем.

```
Router(config)# enable secret level 15 cisco123
Router(config)# username admin privilege 15 secret cisco15
```

Для переходу з рівня на рівень, якщо це дозволяє параметр, указаний під час створення облікового запису користувача, достатньо ввести команду *enable* <*level*> і вказати вірний пароль. Для перегляду поточного рівня пріоритету використовується команда *show privilege*. Хоча призначення рівнів привілеїв дозволяє досить гнучко керувати правами, однак у деяких випадках функціональності такого підходу не вистачає, зокрема:

- Немає можливості керувати доступом до окремих інтерфейсних портів, логічних інтерфейсів та окремих слотів на маршрутизаторі.
- Команди нижніх рівнів завжди доступні на верхніх.
- Команди, специфіковані на верхніх рівнях, не доступні на нижніх.
- Призначення команди з кількох слів на певному рівні автоматично призначає всі команди, що починаються з першого слова.

Основний недолік полягає в тому, що за потреби створити користувача, який повинен мати доступ майже до всіх команд, потрібно всі ці команди уводити власноруч.

### 7.2.3 Конфігурування доступу на основі ролей

Для забезпечення більшої гнучкості Cisco розробила механізм керування доступом, що базується на ролях (Role-Based CLI). Це дозволяє визначити набір команд, доступний для тієї чи іншої ролі. Адміністратор може сконфігурувати різні «вигляди» (view) конфігурації маршрутизатора для різних ролей.

**Безпека.** Role-Based CLI надає більший рівень захисту пристрою шляхом визначення набору команд, що доступні окремому користувачу. Додатково адміністратор може контролювати доступ користувача до окремих портів, логічних інтерфейсів і слотів маршрутизатора. **Доступність.** Role-Based CLI запобігає випадковому запуску команд неавторизованим користувачам і зменшує ймовірність негативних наслідків. **Покращення ефективності роботи.** Користувач бачить тільки ті команди, які йому дозволено, що зменшує їхню загальну кількість і спрощує пошук необхідної команди. Role-based CLI підтримує 3 типи «виглядів» (рисунок): Кореневий (Root); CLI; Супервигляд (Superview). Кожний «вигляд» визначає перелік доступних команд. **Кореневий «вигляд».** Для конфігурування будь-якого «вигляду» для системи адміністратор має знаходитись в кореневому «вигляді». Кореневий «вигляд» має той же рівень привілеїв, як і користу-

вач, що має 15-й рівень привілеїв. Однак кореневий вигляд це не теж саме, що користувач 15-го рівня. Тільки користувач з кореневого «вигляду» може конфігурувати нові «вигляди» та додавати або видаляти команди з існуючих. **CLI вигляд.** Специфічний набір команд може бути об'єднаний у CLI вигляд. На відміну від рівнів привілей, CLI вигляд не має ієрархії команд і, таким чином, не існує «вищих» та «нижчих» «виглядів». До кожного вигляду мають бути прив'язані певні команди і «вигляд» не спадкує команд з інших «виглядів». Однакові команди можуть бути в різних «виглядах» (рис. 7.2).

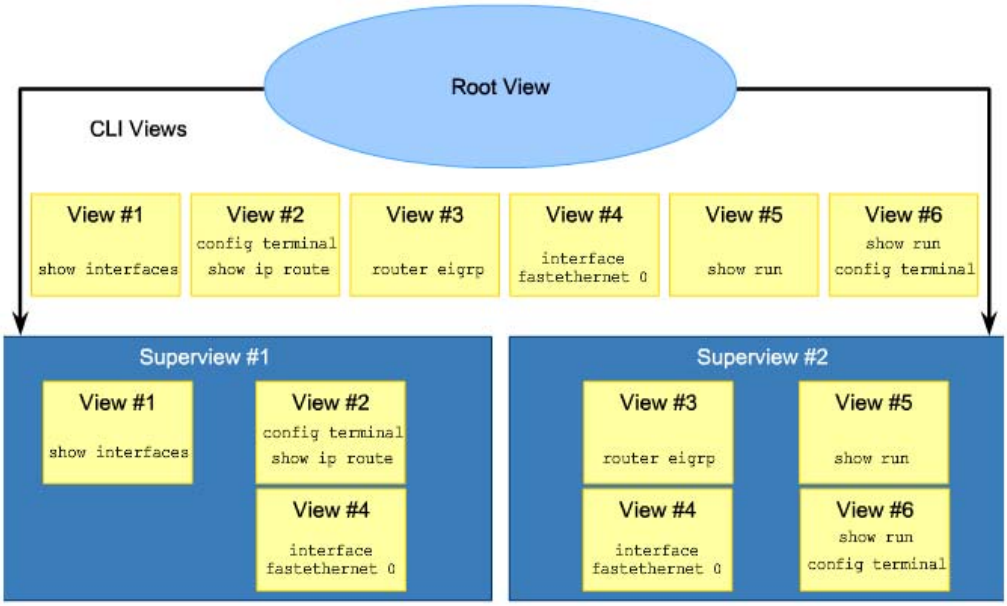


Рисунок 7.2 – Структура Role-based CLI

**Супервигляд.** Супервигляд складається з одного або кількох «виглядів». Супервигляди дозволяють мережевому адміністратору призначати користувачам та групам користувачів кілька CLI-виглядів одночасно. Супервигляди мають такі характеристики:

- Один CLI-вигляд може входити одночасно до N-супервиглядів.
- До супервигляду не можна безпосередньо додавати команди, адміністратор може додавати команди тільки до CLI-вигляду і вводити його до складу супервигляду.
- Користувач, що зареєструвався у «супервигляд» отримує доступ до всіх команд, що сконфігуровані для будь-якого CLI-вигляду, що входить до складу супервигляду.
- Кожний супервигляд має пароль, який використовується для переходу між супервиглядами або від CLI-вигляду до супервигляду.
- Видалення супервигляду не видаляє асоційовані з ним CLI-вигляди.
- Перед тим, як адміністратор може створити вигляд, має бути активований AAA-сервіс з використанням команди `aaa new-model`.

Для конфігурування і переключення між виглядами адміністратор має зареєструватись у кореневий «вигляд», використовуючи команду `Router#enable view <root>`.

У рядку запрошення необхідно ввести пароль, заданий у режимі enable secret. Для створення і налаштування специфічного вигляду варто зробити такі налаштування:

1. Активувати AAA-сервіс та увійти в кореневий вигляд:
```
Router(config)#aaa new-model
Router(config)#exit
Router#enable view <root>
```
2. Створити вигляд за допомогою команди:
```
Router(config)#parser view <view-name>
```

Ця команда активує режим конфігурування вигляду. Загальна кількість виглядів, яку можна створити, – 15, не враховуючи кореневий.

3. Призначити секретний пароль на вигляд, використовуючи команду:
```
Router(config-view)# secret <encrypted-password>
```
4. Призначити команди для цього вигляду таким чином:
```
Router(config-view)# commands parser-mode {include | include-exclusive | exclude} [all] [interface interface-name | command]
```

Приклад створення і налаштування трьох «виглядів» наведено нижче.

```
R1(config)# parser view SHOWVIEW
*Mar 1 09:54:54.873: %PARSER-6-VIEW_CREATED: view 'SHOWVIEW' successfully created.
R1(config-view)# secret cisco
R1(config—view)# commands exec include show
R1(config-view)# exit
R1(config]# parser view VERIFYVIEW
*Mar 1 09:55:24.813: %PARSER-6-VIEW_CREATED: View 'VERIFYVIEW' successfully created.
R1(config—view)# commands exec include ping
% Password not set for the view VERIFYVIEW
R1(config-view)# secret cisco5
R1(config-view)# commands exec include ping
R1(config—view)# exit
R1(config)# parser view REBOOTVIEW
R1(config-view)#
*Mar 1 09:55:52.297: %PARSER-6-VIEW_CREATED: View 'REBOOTVIEW' successfully created.
R1(config—view)# secret cisco10
R1(config—view)# commands exec include reload
R1(config-view)# exit
```

Кроки конфігурування супервигляду аналогічні конфігуруванню CLI-вигляду, за винятком того, що замість команди *commands*, яка призначає команди, використовується `view <view-name>` команда, що призначає вигляди. Адміністратор має знаходитись у кореневому вигляді для конфігурування супервигляду. Для переходу до кореневого вигляду використовується команда `enable view` або `enable view root`. Для створення і керування супервиглядами необхідно:

– Створити супервигляд, використовуючи команду:
```
Router(config)#parser view <view-name> superview
```

– Призначити секретний пароль для цього супервигляду з використанням команди:
```
secret <encrypted-password>
```
– Ввести до складу супервигляду вигляд, який існує, використовуючи команду:
```
Router(config-view)# view <view-name>
```

Більше ніж один вигляд може бути введений до складу супервигляду. Для доступу до виглядів, що існують, необхідно ввести команду `Router>enable view <view-name>`. Після входу до певного вигляду за допомогою команди «?» можна переглянути перелік доступних команд. Користувач, що зареєструвався в кореневий вигляд, має можливість за допомогою команди `show parser view all` переглянути перелік усіх виглядів.

## 7.3 Моніторинг та керування пристроями

### 7.3.1 Захист образу IOS та конфігураційного файлу

Захист процесу керування та відслідковування змін конфігураційних файлів у великій мережі є досить складною задачею. Адміністратору важливо знати стан критичних пристроїв та коли було здійснено останню модифікацію конфігурації. Також актуальним є питання гарантій, що саме легітимні користувачі мали доступ до пристроїв у момент змін конфігурації. Створення плану керування змінами налаштувань обладнання має бути частиною загальної політики безпеки. Це передбачає: збереження резервних копій конфігураційних файлів до внесення змін на FTP або TFTP сервері у разі необхідності повернення до початкових налаштувань; фіксація змін з використанням системи аутентифікації на Syslog-сервері.

Автоматичне передавання звітів та реєстраційних файлів з визначених пристроїв на комп'ютер адміністратора або спеціалізований сервер є дуже важливою задачею. Ці звіти можуть містити інформацію про тип контенту, зміни в конфігурації тощо. Перелік інформації, про яку варто звітувати вирішують спеціалісти з керування, адміністрування та захисту мережі відповідно до політики безпеки установи. Фактично, пристрої мають відправляти, так звані, лог-файли, що відіграватимуть важливу роль у разі проблем з пристроєм або під час атаки. Крім того, доцільно мати інструмент, який дозволить виділяти найактуальнішу на цей час інформацію. Для реалізації згаданої задачі можуть бути використані різноманітні протоколи й програми.

Незалежно від протоколу існують два шляхи передавання керуючої інформації між пристроями та хостами адміністраторів. Через спеціалізований канал (Out-of-band) – інформація передається через спеціально виділені мережі, через які жодний інший трафік не передається. Через загальну мережу (In-band) – інформація передається через ту ж саму мережу, що і звичайний трафік користувача. У другому випадку необхідно вживати різноманітні заходи для захисту даних, що передаються, зокрема, за допомо-

гою VPN-тунелю або шляхом шифрування. У першому випадку фактично використовується спеціальна керуюча мережа, через яку передається тільки керуючий трафік. Вона є дуже привабливою для хакерів, зокрема, хакер може зламати один з керуючих хостів і через нього спробувати отримати доступ до інших пристроїв. Для запобігання цього керуючі хости мають знаходитись у різних сегментах, VLAN тощо. Вибір конкретного способу моніторингу та передавання службової інформації залежить від конкретної ситуації. Варіант out-of-band є більш захищеним, але й дорожчим. Його доцільно використовувати у великих корпоративних мережах. In-band більше підходить для невеликих мереж, для захисту трафіку можуть використовуватись механізми тунелювання або шифрування. Найбільш поширеними сервісами, що забезпечують генерування та передавання системних повідомлень, є Syslog та SNMP.

### 7.3.2 Використання сервісу системних повідомлень (Syslog) для безпеки мережі

Коли відбуваються якісь важливі події на мережевому пристрої, наприклад, відключився інтерфейс або через протокол OSPF установились сусідські відносини з іншим маршрутизатором, генеруються так звані системні повідомлення (syslog messages). За замовчуванням ці повідомлення спрямовуються на консоль відповідного пристрою. Існує також можливість їх спрямувати на лінії віртуального термінала або в буферну пам’ять. Однак усі ці варіанти не забезпечують тривалого збереження інформації про системні події. Одним з найбільш популярних методів фіксації, збереження та аналізу системних подій є застосування виділеного Syslog-сервера, на який відправляють системні повідомлення всі пристрої мережі – Syslog-клієнти, як показано на рис. 7.3.

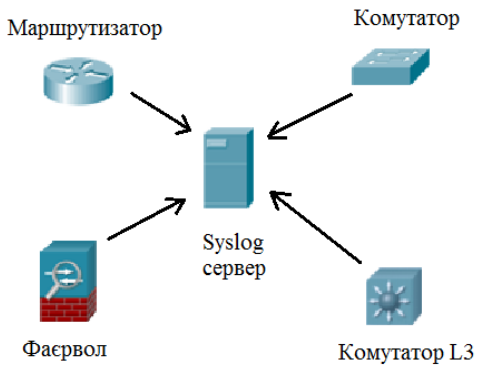


Рисунок 7.3 – Застосування виділеного Syslog-сервера

Протокол Syslog працює через UDP протокол і використовує 514 порт. Оскільки системні події мають різний ступінь важливості, їх поділено на вісім груп важливості (severity level) з номерами від 0 до 7. Що менший номер групи, то вищий ступінь важливості події. Перелік груп наведено в табл. 7.1.

Таблиця 7.1 – Перелік груп

| Номер групи | Назва групи | Опис |
|---|---|---|
| 0 | Аварія (Emergency) | Система непрацездатна |
| 1 | Тривога (Alert) | Потрібне термінове втручання |
| 2 | Критична (Critical) | Відбулись критичні події |
| 3 | Помилка (Error) | Виникла помилка |
| 4 | Попередження (Warning) | Попередження про некоректну роботу |
| 5 | Повідомлення (Notification) | Система працює в нормальному стані, але відбулась важлива подія |
| 6 | Інформація (Informational) | Звичайне інформаційне повідомлення |
| 7 | Налагодження (Debugging) | Повідомлення налагодження |

За замовчуванням структура Syslog-повідомлень має таку структуру:

```
timestamp: %facility-severity-MNEMONIC: description
```

`timestamp` – часовий штамп події;

`%facility` – компонент пристрою, з яким пов'язана подія;

`Severity` – номер групи важливості;

`MNEMONIC` – мнемонічний ідентифікатор події;

`Description` – опис події.

Наприклад, Syslog згенерував повідомлення:

```
12:45:46: %LINK-3-UPDOWN: Interface Port-channel1, changed
state to down
```

Повідомлення може бути інтерпретовано таким чином: о 12.45 відбулась подія 3-го рівня важливості, зокрема, інтерфейс Port-channel1 перейшов у вимкнений стан.

**Етапи конфігурування сервісу системних повідомлень**

– Указати адресу Syslog-сервера, куди будуть надсилатися повідомлення.

```
Router(config)# logging host [name|IP-address]
```

– Указати мінімальний рівень важливості повідомлень, які будуть надсилатись на сервер. За замовчуванням це число дорівнює 7, тобто всі повідомлення відсилаються на сервер.

```
Router(config)# logging trap level
```

– Указати інтерфейс, адреса якого буде вказуватись як адреса відправника в повідомленнях.

```
Router(config)#  logging source-interface
```

– Указати про необхідність генерування часового штампу (за замовчуванням він не генерується).

```
Router(config)#service timestamps log datetime msec
```

– Увімкнути Syslog-сервіс на пристрої.

```
Router(config)# logging on
```

Приклад конфігурування Syslog-сервіса наведено нижче.

```
Rt(config)# logging 10.2.2.6
Rt(config)# logging trap informational
Rt(config)# logging source-interface loopback 0
Rt(config)#service timestamps log datetime msec
Rt(config)# logging on
```

### 7.3.3 Використання SNMP для забезпечення мережевої безпеки

SNMP – протокол прикладного рівня, призначений для обміну керуючою інформацією і використовується для керування серверами, маршрутизаторами й іншими пристроями, дозволяє адміністратору контролювати продуктивність мережі та знаходити проблеми.

Основними компонентами SNMP-сервісу є: SNMP-менеджер – пристрій, на якому запущено систему керування мережею; SNMP-агент – пристрій, яким власне керують; база даних керуючої інформації (MIB). У будь-якій конфігурації як мінімум на одному вузлі має бути запущено програмне забезпечення менеджера, пристрої, що потребують керування, мають бути оснащені SNMP-агентами. Агент відповідає за доступ до локальної MIB об'єкта, у якій відображається активність пристрою, а також зберігаються інші параметри. Протокол SNMP використовує три типи команд: get, set та trap, як показано на рис. 7.4.

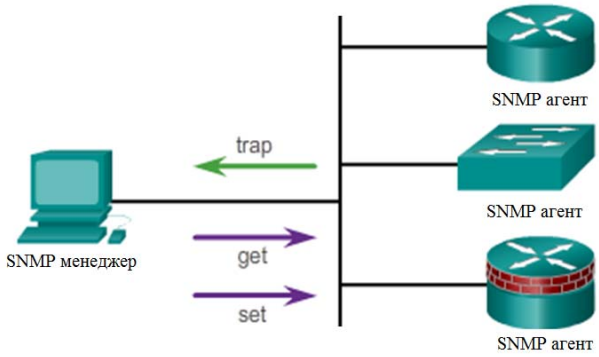


Рисунок 7.4 – Команди, що використовує протокол SNMP

За допомого команди get SNMP менеджер запитує інформацію у агента, команда set дозволяє змінювати або встановлювати інформацію на агенті. Наприклад, set може примусити маршрутизатор перезавантажитись, відправити конфігураційний файл або прийняти конфігураційний файл. Команда trap змушує агента надсилати повідомлення менеджеру у випадку певних подій, наприклад, завантаження центрального процесора перебільшило порогове значення.

SNMP-агенти приймають команди й запити від менеджера тільки за умови, що останні мають коректну community string (CS). CS – це текстовий рядок, який використовується для аутентифікації менеджера агентами й забезпечення доступу до інформації в MIBs. Існують два типи CS: Read-only CS – надає доступ у режимі «тільки зчитування» до всіх об'єктів MIB, за винятком CS; Read-write CS – надає повний доступ до всіх об'єктів MIB, за винятком CS; Read-write CS фактично є еквівалентом пароля на привілейований режим.

Варто звернути увагу, що SNMPv1 та SNMPv2 пересилають CS у відкритому вигляді, що суттєво погіршує безпеку, тому за таких умов для покращення безпеки рекомендують використовувати тільки Read-only CS. При

використанні out-of-band адміністративного доступу можна використовувати і CS з правами запису, однак треба пам'ятати про погіршення рівня безпеки. SNMPv3 забезпечує захищеність сервісу за допомогою контролю цілісності, аутентифікації та шифрування повідомлень.

База даних керуючої інформації використовується для систематизації доступу до різних параметрів та налаштувань мережевих пристроїв. Об'єкти MIB та її структура визначаються RFC 2578, а також RFC 1155, RFC 1213 та RFC 1157. База даних має ієрархічну структуру, що забезпечує уніфікований доступ до MIB змінних з боку програмного забезпечення управління та моніторингу мережевим пристроєм. Формально в MIB кожна змінна характеризується, так званим, ідентифікатором об'єкту (OID). OID дозволяє однозначно ідентифікувати керовані об'єкти в ієрархії MIB.

MIB будь-якого пристрою, як правило, відображають у вигляді дерева, що містить гілки зі змінними, загальними для багатьох мережевих пристроїв і гілки зі змінними, специфічними для такого пристрою або виробника.

Процес конфігурування SNMP-сервісу передбачає виконання таких дій:

- Конфігурування CS із вказанням рівня доступу (read-only або read-write). Для підвищення захищеності також за допомогою ACL може бути визначено перелік хостів, з яких дозволено відправляти SNMP-запити:
  ```
  Rt(config)#snmp-server community string ro | rw ACL_name
  ```
- За необхідності застосування SNMP trap варто вказати адресу менеджера, якому їх спрямовувати:
  ```
  Rt(config)#snmp-server host host-id [version{1| 2c}] community-string
  ```
- Увімкнути на агенті дозвіл на генерування SNMP trap:
  ```
  Rt(config)#snmp-server enable traps
  ```

### 7.3.4 NTP

При аналізі подій, що відбувались під час атаки, дуже важливою є їхня послідовність: для цього треба мати коректні часові позначки, Seslog-повідомлення мають бути синхронізовані в часі. Для встановлення часу та дати на маршрутизаторі є 2 підходи: ручне конфігурування та конфігурування за допомогою Network Time Protocol (NTP). Головним недоліком ручного конфігурування є погане масштабування та відсутність синхронізації між годинниками різних пристроїв. Використання NTP-сервера дозволяє маршрутизаторам синхронізувати свій час з NTP-сервером і робить годинники всіх маршрутизаторів синхронними. При застосуванні NTP може бути використано приватне джерело часу (private master clock) або публічне (NTP-сервер з Інтернет). NTP використовує 123 порт UDP і описаний в RFC 1305. При вирішенні питання щодо вибору приватного чи публічного джерела необхідно зважити ризики та переваги обох. При ви-

користанні приватного джерела, воно має синхронізуватись з Coordinated Universal Time (UTC) через супутниковий або радіоканал. Адміністратору немає потреби хвилюватись щодо коректності отриманого часу. При використанні інтернет-джерел виникає питання довіри, оскільки більшість NTP-серверів не здійснюють жодної аутентифікації під час передавання часових позначок. Взаємодія (асоціація) між пристроями, що використовують NTP, як правило, конфігурується статично. На пристрої, що виконує функцію NTP-сервера, дають команду:

```
NTP_server(config)#ntp master stratum
```

Параметр `stratum` вказує відстань у хопах від оригінального джерела часу, за замовчуванням дорівнює 1.

На кожному пристрої-клієнті вказують IP-адресу NTP-майстра. У мережі, як правило, один або кілька маршрутизаторів виконують функцію NTP master. Для конфігурування зв'язку з майстром використовується команда:

```
NTP_client(config)#ntp server <ntp-server-address>.
```

Інший варіант налаштування NTP-клієнта – не вказувати в явному вигляді адресу NTP-сервера, а на інтерфейсі, через який має надходити інформація про час, дати команду:

```
NTP_client (config-if)#ntp broadcast client.
```

У такому разі часові позначки приймаються через відповідний інтерфейс. Якщо надходять повідомлення від кількох NTP-серверів перевага надається тому, у якого менше значення параметра *stratum.*

Оскільки значення поточного часу є досить критичним параметром, необхідно використовувати опції безпеки NTP, щоб уникнути помилкового встановлення часу. Існують два механізми для вирішення цієї задачі:

- установити обмеженнях у вигляді ACL;
- скористатись механізмом захищеної аутентифікації, що реалізований в NTPv3 та вищих версіях цього протоколу.

Для конфігурування захищеного сервісу на сторонах NTP-майстра і клієнта необхідно виконати такі дії:

- Активувати функцію аутентифікації:

```
Router(config)#ntp authenticate
```

- Налаштувати перелік ключів аутентифікації:

```
Router(config)#ntp authentication-key key-number md5 key-value
```

Ідентифікувати актуальний ключ для системи часової синхронізації:

```
Router(config)#ntp trusted-key key-number
```

Для перевірки стану NTP-сервісу на клієнті використовується команда:

```
NTP_client #show ntp associations detail.
```

# 8 РЕАЛІЗАЦІЯ ЗАХИЩЕНОГО ПЕРЕДАВАННЯ ДАНИХ

## 8.1 Принципи утворення захищеного каналу

Завдання із захисту даних можна розділити на дві частини: захист даних усередині комп'ютера і захист даних під час їхнього передавання з одного комп'ютера на інший. Для забезпечення безпеки даних під час їхнього передавання через публічні мережі використовуються різноманітні технології захищеного каналу.

**Технологія захищеного каналу забезпечує захист трафіку між двома точками у відкритій транспортній мережі**, наприклад в Інтернеті. Захищений канал передбачає виконання трьох основних функцій: взаємна аутентифікація абонентів при встановленні з'єднання; захист повідомлень, що передаються через канал, від несанкціонованого доступу; контроль цілісності повідомлень, що надходять через канал.

Залежно від місця розташування програмного або апаратного забезпечення захищеного каналу існують дві схеми його утворення:

- схема з кінцевими вузлами, що взаємодіють через публічну мережу;
- схема із обладнанням постачальника послуг публічної мережі, розташованим на кордоні між приватною і публічною мережами.

У першому випадку захищений канал утворюється програмними засобами, установленими на двох віддалених комп'ютерах, які належать двом різним локальним мережам одного підприємства і з'єднані між собою через публічну мережу. Перевагою такого підходу є повна захищеність каналу вздовж всього шляху проходження даних, а також можливість використання будь-яких протоколів створення захищених каналів за умови, що в кінцевих точках каналу підтримується однаковий протокол. Недоліки полягають у надлишковості й децентралізованості рішення. Надлишковість полягає в недоцільності використання захисту на всьому шляху, як правило, небезпечними є тільки окремі ланки, що проходять через публічні мережі. Тому захист каналів доступу до публічної мережі можна вважати надлишковим. Децентралізація полягає в тому, що для кожного комп'ютера, якому необхідно надати послуги захищеного каналу, потрібно окремо встановлювати, конфігурувати й адмініструвати програмні засоби захисту даних.

У другому випадку клієнти та сервери не беруть участі у створенні захищеного каналу – він прокладається тільки всередині публічної мережі з комутацією пакетів, наприклад, усередині мережі Інтернет. Захищений канал, наприклад, може бути прокладений між сервером віддаленого доступу постачальника послуг публічної мережі й прикордонним маршрутизатором корпоративної мережі. Таке рішення є добре масштабованим і зручним, з погляду адміністрування. Для комп'ютерів корпоративної мережі канал прозорий – програмне забезпечення цих кінцевих вузлів залишається без змін. Такий гнучкий підхід дозволяє легко створювати

нові канали захищеної взаємодії між комп'ютерами незалежно від місця їхнього розташування. Реалізація такого підходу є складнішою – потрібен стандартний протокол утворення захищеного каналу, необхідне встановлення у всіх постачальників послуг програмного забезпечення, що підтримує такий протокол, необхідна підтримка протоколу виробниками прикордонного комунікаційного обладнання. Проте варіант, коли всі турботи щодо підтримки захищеного каналу бере на себе постачальник послуг публічної мережі, залишає сумніви в надійності захисту: по-перше, незахищеними виявляються канали доступу до публічної мережі; по-друге, споживач послуг відчуває себе повністю залежним від надійності постачальника послуг.

## 8.2 Протокол IPsec

Безпеку на мережевому рівні забезпечує протокол IPsec (IP security – безпечний протокол IP), який фактично є набором протоколів, основні аспекти якого описані в RFC 2401 та RFC 2411. До складу ядра IPsec входять три протоколи: AH (Authentication Header – заголовок аутентифікації) – гарантує цілісність і аутентичність даних; ESP (Encapsulating Security Payload – інкапсуляція зашифрованих даних) – шифрує дані, що передаються, забезпечуючи конфіденційність, може також підтримувати аутентифікацію і цілісність даних; IKE (Internet Key Exchange – обмін ключами Інтернету) – вирішує допоміжну задачу автоматичного надання кінцевим точкам захищеного каналу секретних ключів, необхідних для роботи протоколів аутентифікації і шифрування даних.

Для того, щоб протоколи AH і ESP могли виконувати свої функції щодо захисту даних, протокол IKE встановлює між двома кінцевими точками логічне з'єднання, яке в стандартах має назву **безпечної асоціації** (БА) (Security Association, SA). Безпечна асоціація в протоколі IPsec є однонаправленим (симплексне) логічним з'єднанням, тому у разі потреби безпечного двобічного обміну даними необхідно встановлювати дві безпечні асоціації. Ці асоціації загалом можуть мати різні характеристики, наприклад, в один бік під час передачі запитів до бази даних достатньо тільки аутентифікації, а для відповідей, що містять цінну інформацію, додатково необхідно забезпечити конфіденційність.

Установлення безпечної асоціації починається із взаємної аутентифікації сторін, оскільки всі заходи безпеки втрачають сенс, якщо дані передаються або приймаються не тією особою або не від тієї особи. Параметри БА, що вибираються далі, визначають, який із двох протоколів, AH або ESP, буде застосовуватись для захисту даних, які функції виконуватиме протокол (тільки аутентифікацію і перевірку цілісності або, крім того, також забезпечувати конфіденційність). Дуже важливими параметрами безпечної асоціації є також секретні ключі, що використовуються в роботі протоколів AH і ESP. Протокол IPsec припускає як автоматичне, так і руч-

не встановлення безпечної асоціації. Для кожної задачі, яку вирішують протоколи AH і ESP, пропонується кілька схем аутентифікації й шифрування. Це робить протокол IPsec досить гнучким засобом.

### 8.3 Режими роботи IPsec

Протоколи AH і ESP можуть захищати дані у двох режимах: **транспортному** і **тунельному**. У транспортному режимі передача IP-пакета через мережу виконується за допомогою оригінального заголовка цього пакета, а в тунельному режимі вихідний пакет розміщується в новому IP-пакеті, і передача даних через мережу здійснюється на основі заголовка нового IP-пакета. Застосування того чи іншого режиму залежить від вимог, що висуваються до захисту даних, а також від ролі, яку відіграє в мережі вузол, що завершує захищений канал. Вузол може бути кінцевим (хостом) або проміжним (шлюзом), відповідно, існує три схеми застосування протоколу IPsec: шлюз–шлюз (рис. 8.1, а), хост–шлюз (рис. 8.1, б) та хост–хост.

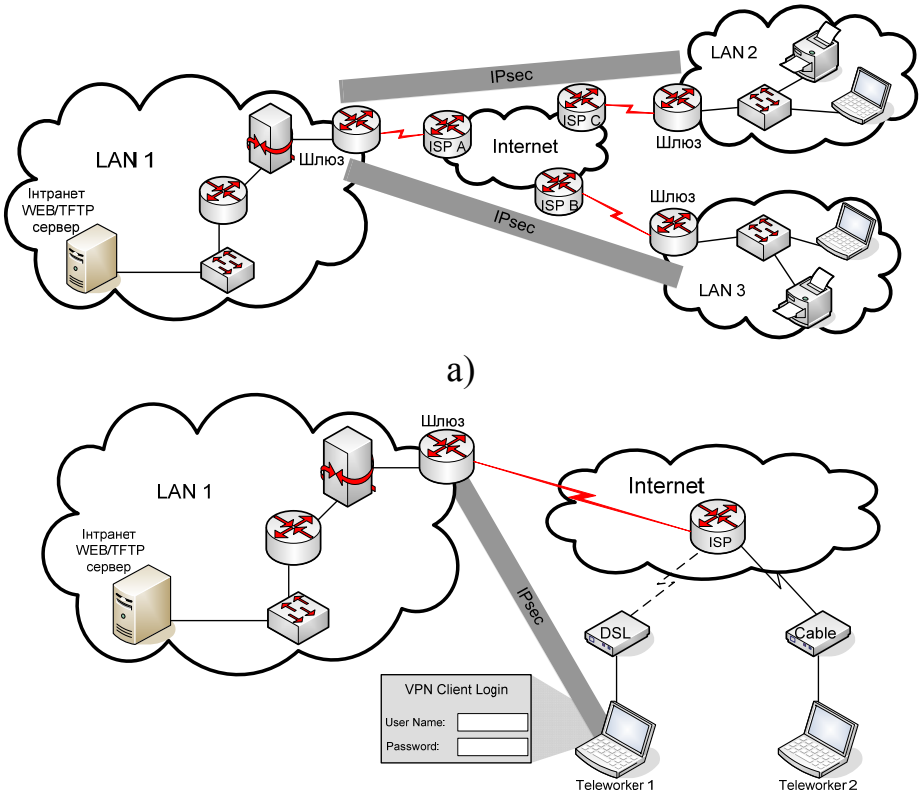


Рисунок 8.1 – Схеми застосування протоколу IPsec:
а) шлюз–шлюз; б) хост–шлюз

У **схемі шлюз–шлюз** захищений канал установлюється між двома проміжними вузлами, так званими, шлюзами безпеки, на кожному з яких працює протокол IPsec. Захищений обмін даними може проходити між будь-якими двома кінцевими вузлами, що розташовані позаду шлюзів без-

пеки. Від кінцевих вузлів підтримка протоколу IPsec не вимагається, вони передають свій трафік у незахищеному вигляді через внутрішні мережі підприємства. Трафік, спрямований у загальнодоступну мережу, проходить через шлюз безпеки, який і забезпечує його захист за допомогою протоколу IPsec. Шлюзам доступний тільки тунельний режим роботи. **Схема хост–шлюз** часто застосовується під час віддаленого доступу. У цьому випадку захищений канал прокладається між віддаленим хостом, на якому працює протокол IPsec, і шлюзом, що захищає трафік для всіх хостів, які входять у внутрішню мережу підприємства. У **схемі хост–хост** захищений канал установлюється між двома кінцевими вузлами мережі, а протокол IPsec працює на кінцевих вузлах і захищає дані, що передаються між ними. Для схеми хост–хост найчастіше застосовують транспортний режим захисту.

## 8.4 Протокол AH

Протокол AH дозволяє на боці отримувача перевірити, що: пакет був відправлений учасником безпечної асоціації; уміст пакета не був змінений під час його проходження через мережу; пакет не є копією уже отриманого пакета. Перші дві функції є обов'язковими для протоколу AH, а остання вибирається при встановленні асоціації за бажанням. Для реалізації вказаних функцій протокол AH використовує спеціальний заголовок (рис. 8.2). Поле **наступного заголовку** (next header) ідентифікує протокол вищого рівня (TCP, UDP, ICMP). Поле **довжини корисного навантаження** (payload length) містить довжину AH заголовка в тридцятидвобітних словах. **Індекс параметрів безпеки** (Security Parameters Index, SPI) це поле, яке разом з IP-адресою отримувача унікальним чином ідентифікує безпечну асоціацію, у межах якої передається пакет.

| Наступний заголовок | Довжина | Резерв |
|---|---|---|
| Індекс параметрів безпеки (SPI) | | |
| Порядковий номер (SN) | | |
| Дані аутентифікації | | |

Рисунок 8.2 – Структура заголовка протоколу AH

Поле **порядкового номеру** (Sequence Number, SN) вказує на порядковий номер пакета й застосовується для захисту від атак, що базуються на повторному використанні даних процесу аутентифікації. На боці відправника значення цього поля в кожному новому пакеті послідовно збільшується, тому надходження дубліката буде виявлено на боці отримувача. Поле **даних аутентифікації** (authentication data) призначене для аутентифікації й перевірки цілісності пакета. Це значення є дайджестом оригі-

нального IP-пакета й обчислюється за допомогою однієї з двох однобічних функцій шифрування MD5 або SAH-1, які підтримує протокол АН. Проте може використовуватись і будь-яка інша функція, про яку сторони домовились під час установлення асоціації. При обчисленні дайджесту пакета параметром ОФШ є симетричний секретний ключ, який був заданий для такої асоціації вручну або автоматично за допомогою протоколу IKE. Оскільки довжина дайджесту залежить від обраної ОФШ, це поле в загальному випадку має змінний розмір. Місце розташування заголовку АН в пакеті залежить від того, у якому режимі – транспортному чи тунельному – працює захищений канал. Кінцевий пакет в **транспортному режимі** має вигляд, як показано на рис. 8.3.

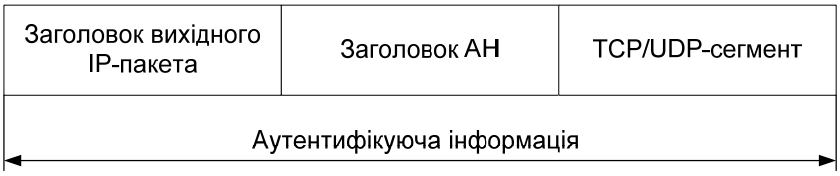


Рисунок 8.3 – Структура IP-пакета, обробленого протоколом АН у транспортному режимі

При використанні **тунельного режиму**, коли шлюз IPsec приймає пакет, що проходить через нього транзитом, і створює для нього зовнішній IP-пакет, протокол АН захищає всі поля вихідного пакета, а також незмінні поля нового заголовку зовнішнього пакета (рис. 8.4).

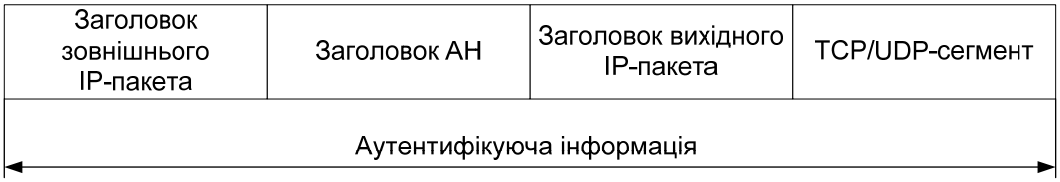


Рисунок 8.4 – IP-пакет, оброблений АН у тунельному режимі

## 8.5 Протокол ESP

Протокол ESP вирішує дві групи задач: 1) **забезпечення аутентифікації й цілісності даних** на основі дайджесту, аналогічні до АН; 2) **захист даних від несанкціонованого перегляду** шляхом їхнього шифрування. Заголовок ESP (рис. 8.5) поділяється на дві частини, що розділяються полем даних. Перша частина, що власне має назву **заголовок ESP**, утворюється полями SPI і SN, призначення яких аналогічне однойменним полям протоколу АН, і розташовується перед полем даних. Інші службові поля протоколу ESP, що мають назву **кінцевик ESP**, розташовані в кінці пакета.


Рисунок 8.5 – IP-пакет, оброблений ESP у транспортному режимі

Два поля кінцевика – **наступного заголовка** і **даних аутентифікації** – також аналогічні полям заголовка AH. Поле даних аутентифікації може бути відсутнім, якщо під час установлення безпечної асоціації прийняте рішення не контролювати цілісність. Кінцевик також містить два додаткових поля – **заповнювача** й **довжини заповнювача**. Заповнювач може знадобитись, оскільки для нормальної роботи деяких алгоритмів шифрування необхідно, щоб текст, який шифрується, містив кратне число блоків певного розміру. Крім того, формат заголовка вимагає, щоб поле даних закінчувалось на кордоні чотирьох байтів. І нарешті, заповнювач можна використовувати, щоб приховати справжній розмір пакета з метою забезпечення, так званої, часткової конфіденційності трафіка. На рис. 8.6 показано розташування полів заголовку в транспортному режимі. У цьому режимі ESP не шифрує заголовок IP-пакета, оскільки маршрутизатор не зможе прочитати поля заголовку й коректно здійснити просування пакета. До переліку полів, що шифруються, не потрапляють також поля SPI і SN, які мають передаватись у відкритому вигляді для того, щоб пакет, який прибув, можна було віднести до певної асоціації й запобігти фальшивому відтворенню пакета.

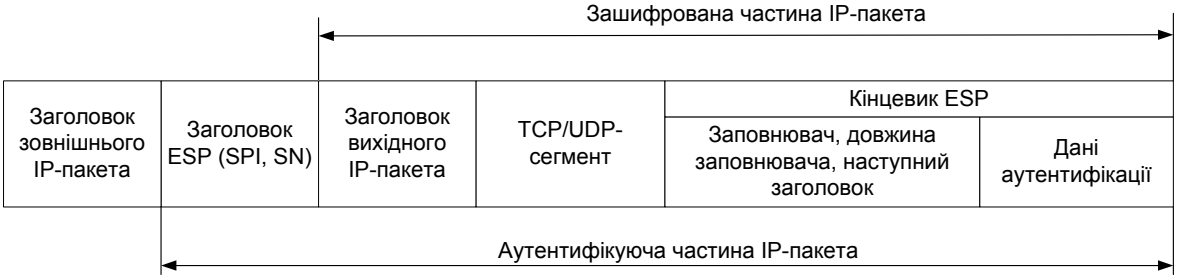


Рисунок 8.6 – IP-пакет, оброблений ESP у тунельному режимі

У тунельному режимі заголовок вихідного IP-пакета розташовується після заголовка ESP і потрапляє до складу захищених полів, а заголовок зовнішнього IP-пакета протоколом ESP не захищається.

# 9 ЛАБОРАТОРНИЙ ПРАКТИКУМ

## Лабораторна робота № 1. Ознайомлення з AAA, NTP та Syslog

**Мета роботи** – отримання навичок налаштування локальної AAA-аутентифікації на мережевому обладнанні, ознайомлення з протоколами NTP та Syslog.

**Хід роботи**

1. За допомогою програмного симулятора комп'ютерної мережі створити схему, показану на рис. 9.1. Призначити адреси, відповідно до табл. 1, налаштувати статичну маршрутизацію. Перевірити доступність хостів за допомогою команди ping.
2. Налаштувати аутентифікацію, відповідно до табл. 9.1 й табл. 9.2.
3. Налаштувати NTP з аутентифікацією на всіх маршрутизаторах.
4. Налаштувати Syslog-сервіс на всіх маршрутизаторах з використанням часових позначок у log-повідомленнях.

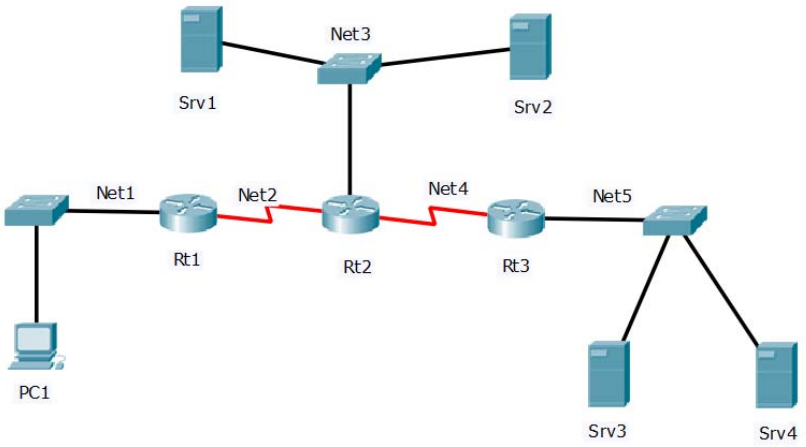


Рисунок 9.1 – Схема мережі для лабораторної роботи № 1

Таблиця 9.1 – Варіанти індивідуальних завдань

| № варіанта | Net1 | Net 2 | Net 3 | Net 4 | Net5 | Mask | Tacacs | Radius | NTP | Syslog |
|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 20.2.0.0 | 20.2.1.0 | 20.2.2.0 | 20.2.3.0 | 20.2.4.0 | /24 | Srv1 | Srv2 | Srv3 | Srv4 |
| 2 | 21.0.0.0 | 21.1.0.0 | 21.2.0.0 | 21.3.0.0 | 21.4.0.0 | /16 | Srv2 | Srv3 | Srv4 | Srv1 |
| 3 | 23.5.0.0 | 23.6.0.0 | 23.7.0.0 | 23.8.0.0 | 23.9.0.0 | /16 | Srv3 | Srv4 | Srv1 | Srv2 |
| 4 | 25.0.0.0 | 26.0.0.0 | 27.0.0.0 | 28.0.0.0 | 29.0.0.0 | /8 | Srv4 | Srv1 | Srv2 | Srv3 |
| 5 | 24.1.0.0 | 25.1.0.0 | 26.1.0.0 | 27.1.0.0 | 30.1.0.0 | /16 | Srv1 | Srv2 | Srv3 | Srv4 |
| 6 | 22.2.1.0 | 22.3.1.0 | 22.4.1.0 | 22.5.1.0 | 22.6.1.0 | /24 | Srv2 | Srv3 | Srv4 | Srv1 |
| 7 | 28.0.0.0 | 28.1.0.0 | 28.2.0.0 | 28.3.0.0 | 28.4.0.0 | /16 | Srv3 | Srv4 | Srv1 | Srv2 |
| 8 | 30.0.0.0 | 31.0.0.0 | 32.0.0.0 | 33.0.0.0 | 34.0.0.0 | /8 | Srv4 | Srv1 | Srv2 | Srv3 |
| 9 | 30.1.0.0 | 30.2.0.0 | 30.3.0.0 | 30.4.0.0 | 30.5.0.0 | /16 | Srv1 | Srv2 | Srv3 | Srv4 |
| 10 | 30.0.1.0 | 30.0.2.0 | 30.0.3.0 | 30.0.4.0 | 30.0.5.0 | /24 | Srv2 | Srv3 | Srv4 | Srv1 |
| 11 | 32.2.0.0 | 32.2.1.0 | 32.2.2.0 | 32.2.3.0 | 32.2.4.0 | /24 | Srv3 | Srv4 | Srv1 | Srv2 |
| 12 | 35.0.0.0 | 35.1.0.0 | 35.2.0.0 | 35.3.0.0 | 35.4.0.0 | /16 | Srv4 | Srv1 | Srv2 | Srv3 |
| 13 | 33.5.0.0 | 33.6.0.0 | 33.7.0.0 | 33.8.0.0 | 33.9.0.0 | /16 | Srv1 | Srv2 | Srv3 | Srv4 |
| 14 | 9.0.0.0 | 11.0.0.0 | 12.0.0.0 | 13.0.0.0 | 14.0.0.0 | /8 | Srv2 | Srv3 | Srv4 | Srv1 |
| 15 | 60.1.0.0 | 60.2.0.0 | 60.3.0.0 | 60.4.0.0 | 60.5.0.0 | /16 | Srv3 | Srv4 | Srv1 | Srv2 |

Таблиця 9.2 – Варіанти індивідуальних завдань

| Варіант | Rt1 | | Rt2 | | Rt3 | |
|---|---|---|---|---|---|---|
| | Tacacs | Radius | Tacacs | Radius | Tacacs | Radius |
| | console | telnet | SSH | console | telnet | console |
| | console | SSH | console | telnet | SSH | console |
| | SSH | console | console | SSH | console | telnet |
| | console | telnet | SSH | console | telnet | console |
| | telnet | console | console | telnet | SSH | console |
| | SSH | console | console | SSH | console | telnet |
| | console | telnet | SSH | console | telnet | console |
| | console | SSH | console | telnet | SSH | console |
| | SSH | console | console | SSH | console | telnet |
| | console | telnet | SSH | console | telnet | console |
| | telnet | console | console | telnet | SSH | console |
| | SSH | console | console | SSH | console | telnet |
| | console | telnet | SSH | console | console | SSH |
| | telnet | console | console | telnet | SSH | console |
| | SSH | console | console | SSH | console | telnet |

**Контрольні питання.** Призначення та основні компоненти AAA-сервісу. Опишіть процес локальної та серверної AAA-аутентифікації. Опишіть процес серверної AAA-аутентифікації. Поясніть функції авторизації й аудиту в AAA-сервісі. Протокол TACACS+. Протокол RADIUS. Основні етапи конфігурування аутентифікації, авторизації й аудиту. Призначення, основні компоненти та етапи налаштування Syslog та NTP-сервісів.

## Лабораторна робота № 2. Захист мережі за допомогою ACL

**Мета роботи** – засвоєння принципів фільтрації трафіка за допомогою ACL, отримання практичних навичок налаштування ACL.

**Хід роботи**

1. За допомогою програмного симулятора комп’ютерних мереж створити схему, як показано на рис. 9.2.
2. Призначити IP-адреси мережам й інтерфейсам відповідно до табл. 9.3.
3. Налаштувати маршрутизацію статичну або динамічну.
4. Перевірити працездатність мережі за допомогою команди ping.
5. За допомогою розширених ACL установити фільтри, що забороняють проходження трафіку, відповідно табл. 9.4. Розташувати фільтри в найбільш вдалому місці.

6. За допомогою програми GNS створити схему, як показано на рис. 9.3, де PC1 та PC2 – віртуальні робочі станції, а Host 1 – комп'ютер або ноутбук, за яким ви працюєте.

7. Призначити IP-адреси мережам Net1 і Net2, відповідно табл. 9.4, а інтерфейсу маршрутизатора, що знаходиться в мережі Net9 – адресу, що належить до простору локальної мережі (WiFi або Ethernet).

8. Налаштувати динамічний ACL таким чином, щоб ping проходив з Host1 на PC1 тільки після встановлення Telnet-сесії з PC1 до R1.

9. Налаштувати рефлективні ACL таким чином, щоб ping проходив з PC1 на PC2 та не проходив з PC2 на PC1.

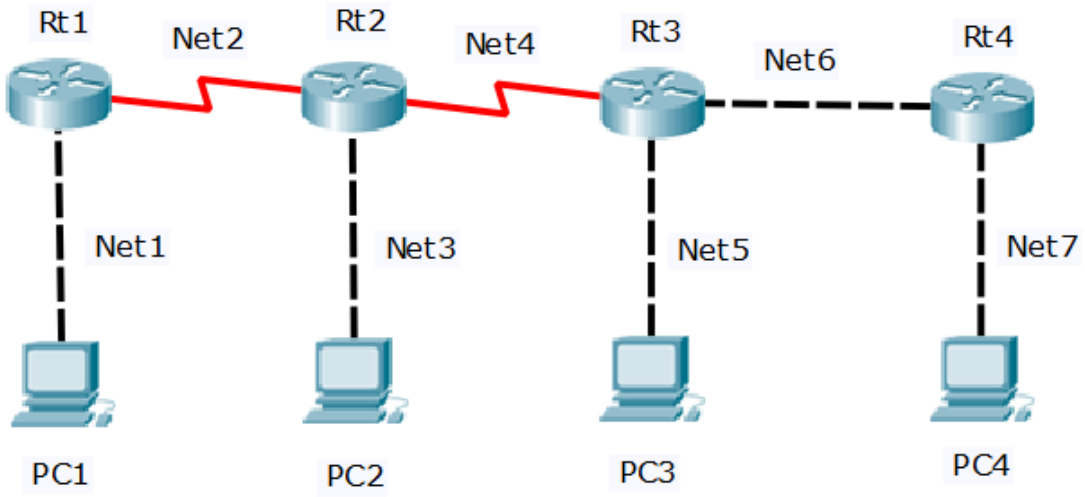


Рисунок 9.2 – Схема мережі (симулятор 1)
для лабораторної роботи № 2

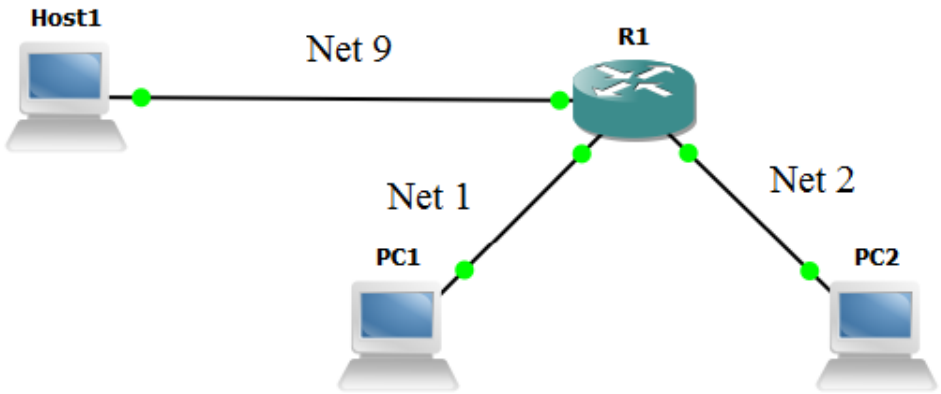


Рисунок 9.3 – Схема мережі (симулятор 2)
для лабораторної роботи № 2

Таблиця 9.3 – Варіанти індивідуальних завдань

| № вар. | Net1 | Net2 | Net3 | Net4 | Net5 | Net6 | Net7 |
|---|---|---|---|---|---|---|---|
| 1. | 10.2.3.0/24 | 186.3.64.0/20 | 200.4.5.32/28 | 27.192.0.0/10 | 153.2.16.0/20 | 195.1.1.8/29 | 18.3.0.0/16 |
| 2. | 18.3.0.0/16 | 10.2.3.0/24 | 186.3.64.0/20 | 200.4.5.32/28 | 27.192.0.0/10 | 153.2.16.0/20 | 195.1.1.8/29 |
| 3. | 195.1.1.8/29 | 18.3.0.0/16 | 10.2.3.0/24 | 186.3.64.0/20 | 200.4.5.32/28 | 27.192.0.0/10 | 153.2.16.0/20 |
| 4. | 153.2.16.0/20 | 195.1.1.8/29 | 18.3.0.0/16 | 10.2.3.0/24 | 186.3.64.0/20 | 200.4.5.32/28 | 27.192.0.0/10 |
| 5. | 27.192.0.0/10 | 153.2.16.0/20 | 195.1.1.8/29 | 18.3.0.0/16 | 10.2.3.0/24 | 186.3.64.0/20 | 200.4.5.32/28 |
| 6. | 200.4.5.32/28 | 27.192.0.0/10 | 153.2.16.0/20 | 195.1.1.8/29 | 18.3.0.0/16 | 10.2.3.0/24 | 186.3.64.0/20 |
| 7. | 186.3.64.0/20 | 200.4.5.32/28 | 27.192.0.0/10 | 153.2.16.0/20 | 195.1.1.8/29 | 18.3.0.0/16 | 10.2.3.0/24 |
| 8. | 18.3.0.0/16 | 10.2.3.0/24 | 153.2.16.0/20 | 195.1.1.8/29 | 27.192.0.0/10 | 186.3.64.0/20 | 200.4.5.32/28 |
| 9. | 195.1.1.8/29 | 18.3.0.0/16 | 186.3.64.0/20 | 200.4.5.32/28 | 27.192.0.0/10 | 153.2.16.0/20 | 10.2.3.0/24 |
| 10. | 186.3.64.0/20 | 200.4.5.32/28 | 153.2.16.0/20 | 195.1.1.8/29 | 18.3.0.0/16 | 10.2.3.0/24 | 27.192.0.0/10 |
| 11. | 13.4.8.0/22 | 196.4.32.0/20 | 186.1.9.32/28 | 38.128.0.0/10 | 173.2.32.0/20 | 183.6.9.8/29 | 56.82.0.0/16 |
| 12. | 196.4.32.0/20 | 186.1.9.32/28 | 38.128.0.0/10 | 173.2.32.0/20 | 183.6.9.8/29 | 56.82.0.0/16 | 13.4.8.0/22 |
| 13. | 186.1.9.32/28 | 38.128.0.0/10 | 173.2.32.0/20 | 183.6.9.8/29 | 56.82.0.0/16 | 13.4.8.0/22 | 196.4.32.0/20 |
| 14. | 38.128.0.0/10 | 173.2.32.0/20 | 183.6.9.8/29 | 56.82.0.0/16 | 13.4.8.0/22 | 196.4.32.0/20 | 186.1.9.32/28 |
| 15. | 173.2.32.0/20 | 183.6.9.8/29 | 56.82.0.0/16 | 13.4.8.0/22 | 196.4.32.0/20 | 186.1.9.32/28 | 38.128.0.0/10 |

Таблиця 9.4 – Варіанти індивідуальних завдань

| № вар. | http | | ftp | | smtp | | telnet | | tftp | |
|---|---|---|---|---|---|---|---|---|---|---|
| | відпр. | отр. | відпр. | отр. | відпр. | отр. | відпр. | отр. | відпр. | отр. |
| 1. | Net1 | Net3 | Net5 | Net7 | Net1 | Net5 | Net7 | Net3 | Net7 | Net1 |
| 2. | Net5 | Net7 | Net1 | Net5 | Net7 | Net3 | Net3 | Net5 | Net3 | Net7 |
| 3. | Net1 | Net5 | Net7 | Net3 | Net3 | Net5 | Net7 | Net1 | Net7 | Net3 |
| 4. | Net7 | Net3 | Net3 | Net5 | Net7 | Net1 | Net1 | Net7 | Net1 | Net5 |
| 5. | Net3 | Net5 | Net7 | Net1 | Net1 | Net7 | Net5 | Net1 | Net5 | Net1 |
| 6. | Net7 | Net1 | Net1 | Net7 | Net5 | Net1 | Net3 | Net1 | Net3 | Net7 |
| 7. | Net1 | Net7 | Net5 | Net1 | Net3 | Net1 | Net7 | Net5 | Net7 | Net1 |
| 8. | Net5 | Net1 | Net3 | Net1 | Net7 | Net5 | Net1 | Net3 | Net5 | Net1 |
| 9. | Net3 | Net1 | Net7 | Net5 | Net1 | Net3 | Net5 | Net7 | Net3 | Net5 |
| 10. | Net7 | Net5 | Net1 | Net3 | Net5 | Net7 | Net1 | Net5 | Net1 | Net3 |
| 11. | Net1 | Net5 | Net7 | Net3 | Net1 | Net3 | Net5 | Net7 | Net5 | Net7 |
| 12. | Net7 | Net3 | Net3 | Net5 | Net5 | Net7 | Net1 | Net5 | Net1 | Net3 |
| 13. | Net3 | Net5 | Net7 | Net1 | Net1 | Net5 | Net7 | Net3 | Net1 | Net7 |
| 14. | Net7 | Net1 | Net1 | Net7 | Net7 | Net3 | Net3 | Net5 | Net3 | Net5 |
| 15. | Net1 | Net7 | Net5 | Net1 | Net3 | Net5 | Net7 | Net1 | Net7 | Net3 |

**Контрольні питання.** Дайте визначення списку керування доступом. Використання інвертованої маски. Стандартні, розширені, динамічні, тимчасові ACL, їхнє застосування. Правила налаштування та розташування ACL.

## Лабораторна робота № 3. Захист мережі на базі CBAC-firewall

**Мета роботи** – засвоєння принципів використання мережевих екранів, що базуються на аналізі контексту (CBAC-firewall), отримання практичних навичок налаштування CBAC-firewall.

**Хід роботи**

1. За допомогою програмного симулятора комп'ютерних мереж створити схему, як показано на рис. 9.4.

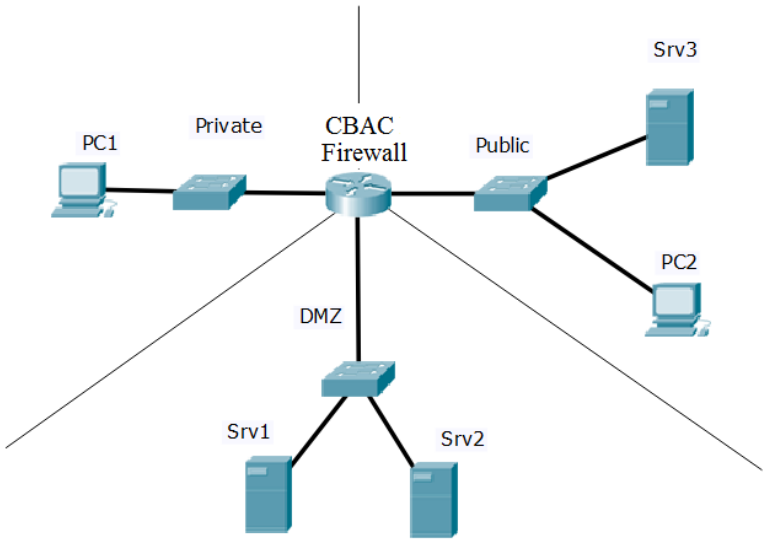


Рисунок 9.4 – Схема мережі для лабораторної роботи № 3

2. Виконати початкові налаштування, призначити адреси відповідно до табл. 9.5.

Таблиця 9.5 – Варіанти індивідуальних завдань

| № вар. | Private/24 | DMZ/24 | Public | Srv1 | Srv2 | Srv3 |
|---|---|---|---|---|---|---|
| 1 | 172.24.26.0 | 192.168.26.0 | 18.3.0.0/16 | HTTP | DNS | SMTP |
| 2 | 10 .8.27.0 | 172.25.27.0 | 173.2.32.0/20 | HTTPS | POP3 | FTP |
| 3 | 192.168.28.0 | 10 .9.28.0 | 183.6.9.8/29 | DNS | SMTP | TFTP |
| 4 | 172.25.29.0 | 192.168.29.0 | 56.82.0.0/16 | POP3 | FTP | HTTP |
| 5 | 10 .9.30.0 | 172.26.30.0 | 13.4.8.0/22 | SMTP | TFTP | HTTPS |
| 6 | 192.168.31.0 | 10 .10.31.0 | 196.4.32.0/20 | FTP | HTTP | DNS |
| 7 | 172.26.32.0 | 192.168.32.0 | 186.1.9.32/28 | TFTP | HTTPS | POP3 |
| 8 | 10 .10.33.0 | 172.27.33.0 | 38.128.0.0/10 | HTTP | DNS | SMTP |
| 9 | 192.168.34.0 | 10 .11.34.0 | 173.2.32.0/20 | HTTPS | POP3 | FTP |
| 10 | 172.27.35.0 | 192.168.35.0 | 58.192.0.0/10 | DNS | SMTP | TFTP |
| 11 | 10 .11.36.0 | 172.28.36.0 | 216.5.64.0/20 | POP3 | FTP | HTTP |
| 12 | 192.168.37.0 | 10 .12.37.0 | 116.3.7.32/28 | SMTP | TFTP | HTTPS |
| 13 | 172.28.38.0 | 192.168.38.0 | 45.7.0.0/18 | FTP | HTTP | DNS |
| 14 | 10 .12.39.0 | 172.29.39.0 | 10.2.4.0/22 | TFTP | HTTPS | POP3 |
| 15 | 192.168.40.0 | 10 .13.40.0 | 173.2.32.0/19 | HTTP | DNS | SMTP |

3. Для протоколів, відповідно до табл. 9.5, налаштувати CBAC-firewall за правилами налаштування DMZ-зони: з мереж Private і Public має пропускатись трафік відповідного типу на сервери Srv1 та Srv2; з мережі Private в мережу Public має пропускатись увесь трафік; з мережі Public в мережу Private має проходити тільки трафік, що є відповіддю на ініційований з мережі Private.

**Контрольні питання.** Різновиди firewall та принципи їхнього застосування. Порівняння firewall з фіксацією стану та без фіксації. Особливості роботи firewall з керуванням доступом та з урахуванням контексту (CBAC-технологія). DMZ-зона та її призначення. Правила налаштування CBAC-firewall

## Лабораторна робота № 4. Захист мережі за допомогою firewall на основі зонних політик (ZBF)

**Мета роботи** – засвоєння принципів використання firewall на основі зонних політик, отримання практичних навичок налаштування ZBF.

**Хід роботи**

1. За допомогою програмного симулятора комп’ютерних мереж створити схему, показану на рис. 9.5.

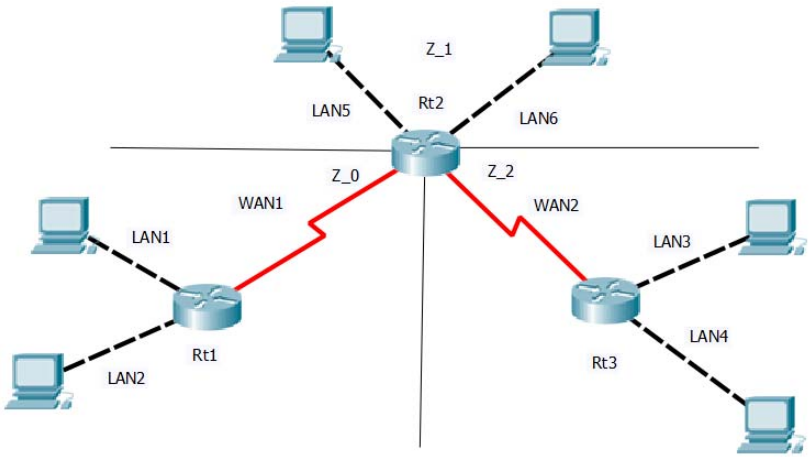


Рисунок 9.5 – Схема мережі для лабораторної роботи № 4

2. Призначити IP-адреси інтерфейсам маршрутизаторів і робочим станціям відповідно до табл. 9.6, причому IP-адреса інтерфейсу має бути останньою із допустимого діапазону, а IP-адреса робочої станції – першою.

3. Сконфігурувати протокол OSPF на всіх маршрутизаторах.

3. Перевірити працездатність мережі за допомогою команди ping.

4. Налаштувати зонні політики для передавання трафіку між зонами Z_0, Z_1, Z_2 відповідно до табл. 9.7.

5. Перевірити працездатність.

Таблиця 9.6 – Варіанти індивідуальних завдань

| № вар. | LAN 1 | LAN 2 | LAN 3 | LAN 4 | WAN 1 | WAN2 | LAN5 | LAN 6 | Mask |
|---|---|---|---|---|---|---|---|---|---|
| 1 | 10.0.0.0 | 11.0.0.0 | 12.0.0.0 | 13.0.0.0 | 14.0.0.0 | 15.0.0.0 | 16.0.0.0 | 17.0.0.0 | /8 |
| 2 | 10.1.0.0 | 10.2.0.0 | 10.3.0.0 | 10.4.0.0 | 10.5.0.0 | 10.6.0.0 | 10.7.0.0 | 10.8.0.0 | /16 |
| 3 | 10.0.1.0 | 10.0.2.0 | 10.0.3.0 | 10.0.4.0 | 10.0.5.0 | 10.0.6.0 | 10.0.7.0 | 10.0.8.0 | /24 |
| 4 | 12.2.0.0 | 12.2.1.0 | 12.2.2.0 | 12.2.3.0 | 12.2.4.0 | 12.2.5.0 | 12.2.6.0 | 12.2.7.0 | /24 |
| 5 | 15.0.0.0 | 15.1.0.0 | 15.2.0.0 | 15.3.0.0 | 15.4.0.0 | 15.5.0.0 | 15.6.0.0 | 15.7.0.0 | /16 |
| 6 | 13.5.0.0 | 13.6.0.0 | 13.7.0.0 | 13.8.0.0 | 13.9.0.0 | 13.1.0.0 | 13.2.0.0 | 13.3.0.0 | /16 |
| 7 | 15.0.0.0 | 16.0.0.0 | 17.0.0.0 | 18.0.0.0 | 19.0.0.0 | 20.0.0.0 | 21.0.0.0 | 22.0.0.0 | /8 |
| 8 | 14.1.0.0 | 15.1.0.0 | 16.1.0.0 | 17.1.0.0 | 10.1.0.0 | 11.1.0.0 | 12.1.0.0 | 13.1.0.0 | /16 |
| 9 | 12.2.1.0 | 12.3.1.0 | 12.4.1.0 | 12.5.1.0 | 12.6.1.0 | 12.7.1.0 | 12.8.1.0 | 12.9.1.0 | /24 |
| 10 | 18.0.0.0 | 18.1.0.0 | 18.2.0.0 | 18.3.0.0 | 18.4.0.0 | 18.5.0.0 | 18.6.0.0 | 18.7.0.0 | /16 |
| 11 | 19.0.0.0 | 20.0.0.0 | 21.0.0.0 | 22.0.0.0 | 23.0.0.0 | 24.0.0.0 | 25.0.0.0 | 26.0.0.0 | /8 |
| 12 | 19.1.0.0 | 19.2.0.0 | 19.3.0.0 | 19.4.0.0 | 19.5.0.0 | 19.6.0.0 | 19.7.0.0 | 19.8.0.0 | /16 |
| 13 | 19.0.1.0 | 10.9.2.0 | 10.9.3.0 | 10.9.4.0 | 10.9.5.0 | 10.9.6.0 | 10.9.7.0 | 10.9.8.0 | /24 |
| 14 | 20.2.0.0 | 20.2.1.0 | 20.2.2.0 | 20.2.3.0 | 20.2.4.0 | 20.2.5.0 | 20.2.6.0 | 20.2.7.0 | /24 |
| 15 | 21.0.0.0 | 21.1.0.0 | 21.2.0.0 | 21.3.0.0 | 21.4.0.0 | 21.5.0.0 | 21.6.0.0 | 21.7.0.0 | /16 |

Таблиця 9.7 – Варіанти індивідуальних завдань

| № вар. | Z_0→Z_1 | | | Z_1→Z_2 | | | Z_2→Z_0 | | |
|---|---|---|---|---|---|---|---|---|---|
| | прот. | трафік | дія | прот. | трафік | дія | прот. | трафік | дія |
| 1 | tcp | LAN1→LAN5 | insp | udp | LAN5→LAN3 | insp | icmp | LAN3→LAN1 | insp |
| 2 | udp | LAN1→LAN6 | pass | icmp | LAN5→LAN4 | insp | tcp | LAN3→LAN2 | pass |
| 3 | icmp | LAN2→LAN5 | insp | tcp | LAN6→LAN3 | pass | udp | LAN4→LAN1 | insp |
| 4 | tcp | LAN2→LAN6 | pass | udp | LAN5→LAN4 | insp | icmp | LAN4→LAN2 | pass |
| 5 | udp | LAN1→LAN5 | insp | icmp | LAN5→LAN3 | pass | tcp | LAN3→LAN1 | insp |
| 6 | icmp | LAN1→LAN6 | pass | tcp | LAN5→LAN4 | insp | udp | LAN3→LAN2 | pass |
| 7 | tcp | LAN2→LAN5 | insp | udp | LAN6→LAN3 | pass | icmp | LAN4→LAN1 | insp |
| 8 | udp | LAN2→LAN6 | pass | icmp | LAN5→LAN4 | insp | tcp | LAN4→LAN2 | pass |
| 9 | icmp | LAN1→LAN5 | insp | tcp | LAN5→LAN3 | pass | udp | LAN3→LAN1 | insp |
| 10 | tcp | LAN1→LAN6 | pass | udp | LAN5→LAN4 | insp | icmp | LAN3→LAN2 | pass |
| 11 | udp | LAN2→LAN5 | insp | icmp | LAN6→LAN3 | pass | tcp | LAN4→LAN1 | insp |
| 12 | icmp | LAN2→LAN6 | pass | tcp | LAN5→LAN4 | insp | udp | LAN4→LAN2 | pass |
| 13 | tcp | LAN1→LAN5 | insp | udp | LAN5→LAN3 | pass | icmp | LAN3→LAN1 | insp |
| 14 | udp | LAN1→LAN6 | pass | icmp | LAN5→LAN4 | insp | tcp | LAN3→LAN2 | pass |
| 15 | icmp | LAN2→LAN5 | insp | tcp | LAN6→LAN3 | pass | udp | LAN4→LAN1 | insp |

**Контрольні питання.** Класифікація firewall за рівнем моделі OSI. Основні типи firewall. Базові правила використання firewall. Переваги та недоліки firewall з фіксацією стану (Stateful) та без фіксації (Stateless). Особливості роботи firewall, що базується на зонах. Переваги ZBF, порівняно з CBAC-firewall. Етапи налаштування ZBF.

## Лабораторна робота № 5. VPN типу шлюз–шлюз

**Мета роботи** – засвоєння принципів функціонування віртуальних приватних мереж (VPN) та особливостей роботи, налаштування VPN типу шлюз–шлюз.

**Хід роботи**

1. За допомогою програмного симулятора комп'ютерних мереж створити схему, показану на рис. 9.6.

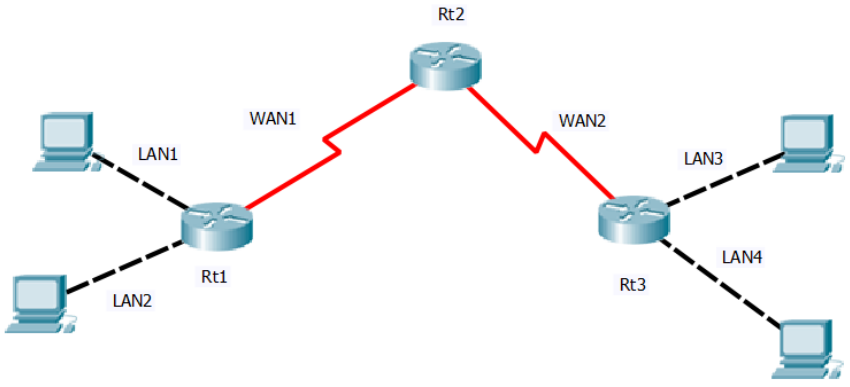


Рисунок 9.6 – Схема мережі для лабораторної роботи № 5

2. Призначити IP-адреси інтерфейсам маршрутизаторів і робочим станціям, відповідно до табл. 9.8, причому IP-адреса інтерфейсу має бути останньою із допустимого діапазону, а IP-адреса робочої станції – першою.

3. Сконфігурувати протокол EIGRP на всіх маршрутизаторах.

6. Перевірити працездатність мережі за допомогою команди ping.

7. Налаштувати VPN з параметрами, що вказані в табл. 9.8 та 9.9.

8. Перевірити працездатність IPsec, відправивши в режимі Simulation пакет, переглянути вміст пакета при входженні в тунель та виході з тунелю, пояснити відмінності.

Таблиця 9.8 – Варіанти індивідуальних завдань

| № вар. | LAN 1 | LAN 2 | LAN 3 | LAN 4 | WAN 1 | WAN2 | Mask | Трафік тунеля | |
|---|---|---|---|---|---|---|---|---|---|
| 1 | 10.0.0.0 | 11.0.0.0 | 12.0.0.0 | 13.0.0.0 | 14.0.0.0 | 15.0.0.0 | /8 | LAN1↔LAN3 | ip |
| 2 | 10.1.0.0 | 10.2.0.0 | 10.3.0.0 | 10.4.0.0 | 10.5.0.0 | 10.6.0.0 | /16 | LAN1↔LAN4 | tcp |
| 3 | 10.0.1.0 | 10.0.2.0 | 10.0.3.0 | 10.0.4.0 | 10.0.5.0 | 10.0.6.0 | /24 | LAN2↔LAN3 | udp |
| 4 | 12.2.0.0 | 12.2.1.0 | 12.2.2.0 | 12.2.3.0 | 12.2.4.0 | 12.2.5.0 | /24 | LAN2↔LAN4 | icmp |
| 5 | 15.0.0.0 | 15.1.0.0 | 15.2.0.0 | 15.3.0.0 | 15.4.0.0 | 15.5.0.0 | /16 | LAN1↔LAN3 | ip |
| 6 | 13.5.0.0 | 13.6.0.0 | 13.7.0.0 | 13.8.0.0 | 13.9.0.0 | 13.1.0.0 | /16 | LAN1↔LAN4 | tcp |
| 7 | 15.0.0.0 | 16.0.0.0 | 17.0.0.0 | 18.0.0.0 | 19.0.0.0 | 20.0.0.0 | /8 | LAN2↔LAN3 | udp |
| 8 | 14.1.0.0 | 15.1.0.0 | 16.1.0.0 | 17.1.0.0 | 10.1.0.0 | 11.1.0.0 | /16 | LAN2↔LAN4 | icmp |
| 9 | 12.2.1.0 | 12.3.1.0 | 12.4.1.0 | 12.5.1.0 | 12.6.1.0 | 12.7.1.0 | /24 | LAN1↔LAN3 | ip |
| 10 | 18.0.0.0 | 18.1.0.0 | 18.2.0.0 | 18.3.0.0 | 18.4.0.0 | 18.5.0.0 | /16 | LAN1↔LAN4 | tcp |
| 11 | 19.0.0.0 | 20.0.0.0 | 21.0.0.0 | 22.0.0.0 | 23.0.0.0 | 24.0.0.0 | /8 | LAN2↔LAN3 | udp |
| 12 | 19.1.0.0 | 19.2.0.0 | 19.3.0.0 | 19.4.0.0 | 19.5.0.0 | 19.6.0.0 | /16 | LAN2↔LAN4 | icmp |
| 13 | 19.0.1.0 | 10.9.2.0 | 10.9.3.0 | 10.9.4.0 | 10.9.5.0 | 10.9.6.0 | /24 | LAN1↔LAN3 | ip |
| 14 | 20.2.0.0 | 20.2.1.0 | 20.2.2.0 | 20.2.3.0 | 20.2.4.0 | 20.2.5.0 | /24 | LAN1↔LAN4 | tcp |
| 15 | 21.0.0.0 | 21.1.0.0 | 21.2.0.0 | 21.3.0.0 | 21.4.0.0 | 21.5.0.0 | /16 | LAN2↔LAN3 | udp |

Таблиця 9.9 – Варіанти індивідуальних завдань

| № вар. | authentication | encryption | hash | group | life-time | key | transform-set |
|---|---|---|---|---|---|---|---|
| 1 | pre-share | 3des | md5 | Group 1 | 100 | Cisco1 | ah-md5-hmac |
| 2 | pre-share | des | sha | Group 2 | 200 | Cisco2 | ah-sha-hmac |
| 3 | pre-share | aes | md5 | Group 5 | 300 | Cisco3 | esp-aes |
| 4 | pre-share | 3des | sha | Group 1 | 400 | Cisco4 | esp-3des |
| 5 | pre-share | des | md5 | Group 2 | 500 | Cisco5 | esp-des |
| 6 | pre-share | aes | sha | Group 5 | 600 | Cisco6 | esp-sha-hmac |
| 7 | pre-share | 3des | md5 | Group 1 | 700 | Cisco7 | esp-md5-hmac |
| 8 | pre-share | des | sha | Group 2 | 800 | Cisco8 | esp-sha-hmac |
| 9 | pre-share | aes | md5 | Group 5 | 900 | Cisco9 | ah-md5-hmac |
| 10 | pre-share | 3des | sha | Group 1 | 1000 | Cisco10 | ah-sha-hmac |
| 11 | pre-share | des | md5 | Group 2 | 1100 | Cisco11 | esp-md5-hmac |
| 12 | pre-share | aes | sha | Group 5 | 1200 | Cisco12 | esp-aes |
| 13 | pre-share | 3des | md5 | Group 1 | 1300 | Cisco13 | esp-3des |
| 14 | pre-share | des | sha | Group 2 | 1400 | Cisco14 | esp-des |
| 15 | pre-share | aes | md5 | Group 5 | 1500 | Cisco15 | ah-md5-hmac |

**Контрольні питання.** Різновиди VPN. GRE-тунелювання. IPsec-функції та framework. Забезпечення конфіденційності/цілісності даних та аутентифікації/захищеного обміну ключами в IPsec. Протоколи AH та ESP. Тунельний та транспортний режими VPN. Безпечна асоціація та фази її створення. Конфігурування VPN типу шлюз–шлюз

## Лабораторна робота № 6. VPN типу хост–шлюз

**Мета роботи** – засвоєння принципів функціонування VPN та особливостей роботи, налаштування VPN типу хост–шлюз.

**Хід роботи**

1. За допомогою програмного симулятора комп'ютерних мереж створити мережу, відповідно до рис 9.7.

2. У межах LAN розподілити адресний простір IPv4 таким чином: у мережах, що з'єднують маршрутизатори, використовувати префікс 30 (маска 255.255.255.252), увесь вільний простір, що залишається, рівномірно розподілити між мережами, у яких розташовані комутатори. Призначити IP-адреси інтерфейсам маршрутизаторів і робочим станціям (IP-адреса інтерфейсу має бути останньою, а IP-адреса робочої станції – першою).

3. Сконфігурувати протокол маршрутизації відповідно до варіанта (табл. 9.10).

4. Налаштувати VPN-шлюз відповідно варіанта (табл. 9.10 та 9.11).

Таблиця 9.10 – Варіанти індивідуальних завдань

| № вар. | № схеми | IP-адреса | Протокол | VPN -шлюз | username | User key | group name | Group key |
|---|---|---|---|---|---|---|---|---|
| 1 | 3 | 172.30.42.0/24 | RIP | Rt1 | VPN1 | vpn1 | REMOTE1 | remote1 |
| 2 | 4 | 10 .14.43.0/24 | OSPF | Rt2 | VPN2 | vpn2 | REMOTE2 | remote2 |
| 3 | 5 | 192.168.44.0/24 | EIGRP | Rt3 | VPN3 | vpn3 | REMOTE3 | remote3 |
| 4 | 6 | 172.31.45.0/24 | RIP | Rt4 | VPN4 | vpn4 | REMOTE4 | remote4 |
| 5 | 7 | 10.22.18.0/24 | OSPF | Rt5 | VPN5 | vpn5 | REMOTE5 | remote5 |
| 6 | 8 | 172.16.19.0/24 | EIGRP | Rt1 | VPN6 | vpn6 | REMOTE6 | remote6 |
| 7 | 9 | 192.168.20.0/24 | RIP | Rt2 | VPN7 | vpn7 | REMOTE7 | remote7 |
| 8 | 10 | 172.23.21.0/24 | OSPF | Rt3 | VPN8 | vpn8 | REMOTE8 | remote8 |
| 9 | 1 | 10 .7.22.0/24 | EIGRP | Rt4 | VPN9 | vpn9 | REMOTE9 | remote9 |
| 10 | 2 | 192.168.23.0/24 | RIP | Rt5 | VPN10 | vpn10 | REMOTE10 | remote10 |
| 11 | 3 | 172.24.24.0/24 | OSPF | Rt1 | VPN11 | vpn11 | REMOTE11 | remote11 |
| 12 | 4 | 10 .8.25.0/24 | EIGRP | Rt2 | VPN12 | vpn12 | REMOTE12 | remote12 |
| 13 | 5 | 192.168.100.0/24 | RIP | Rt3 | VPN13 | vpn13 | REMOTE13 | remote13 |
| 14 | 6 | 172.20.15.0/24 | OSPF | Rt4 | VPN14 | vpn14 | REMOTE14 | remote14 |
| 15 | 7 | 10 .5.16.0/24 | EIGRP | Rt5 | VPN15 | vpn15 | REMOTE15 | remote15 |

Таблиця 9.11 – Варіанти індивідуальних завдань

| № вар. | ISAKMP Policy | | | | | transform-set | |
|---|---|---|---|---|---|---|---|
| | authentication | encryption | hash | Group DH | life-time | | |
| 1 | pre-share | 3DES | md5 | Group 1 | 100 | esp-aes 128 | esp-sha-hmac |
| 2 | pre-share | DES | sha | Group 2 | 200 | esp-3des | esp-md5-hmac |
| 3 | pre-share | AES 128 | md5 | Group 5 | 300 | esp-des | esp-sha-hmac |
| 4 | pre-share | 3DES | sha | Group 1 | 400 | esp-aes 192 | esp-md5-hmac |
| 5 | pre-share | DES | md5 | Group 2 | 500 | esp-3des | esp-sha-hmac |
| 6 | pre-share | AES 192 | sha | Group 5 | 600 | esp-des | esp-md5-hmac |
| 7 | pre-share | 3DES | md5 | Group 1 | 700 | esp-aes 256 | esp-sha-hmac |
| 8 | pre-share | DES | sha | Group 2 | 800 | esp-3des | esp-md5-hmac |
| 9 | pre-share | AES 256 | md5 | Group 5 | 900 | esp-des | esp-sha-hmac |
| 10 | pre-share | 3DES | sha | Group 1 | 1000 | esp-aes 128 | esp-md5-hmac |
| 11 | pre-share | DES | md5 | Group 2 | 1100 | esp-3des | esp-sha-hmac |
| 12 | pre-share | AES 128 | sha | Group 5 | 1200 | esp-des | esp-md5-hmac |
| 13 | pre-share | 3DES | md5 | Group 1 | 1300 | esp-aes 192 | esp-sha-hmac |
| 14 | pre-share | DES | sha | Group 2 | 1400 | esp-3des | esp-md5-hmac |
| 15 | pre-share | AES 192 | md5 | Group 5 | 1500 | esp-des | esp-sha-hmac |

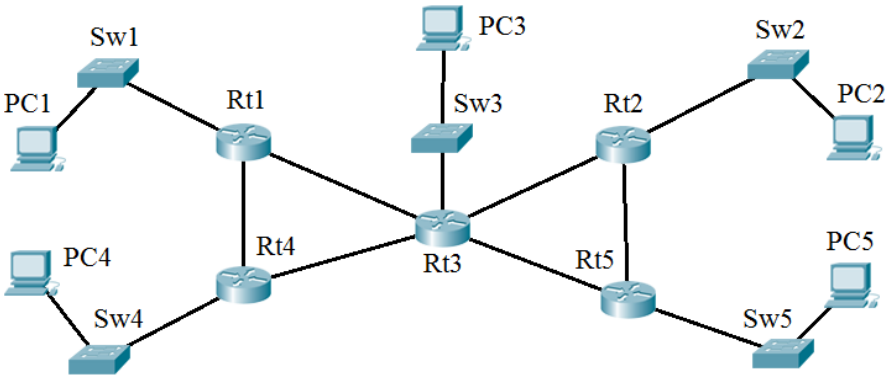


Рисунок 9.7 – Схема мережі для лабораторної роботи № 6

**Контрольні питання.** VPN. Методи шифрування в IPsec. Методи перевірки цілісності в IPsec. Протоколи IPsec. Алгоритми Діффі-Хелмана. Налаштування VPN типу хост–шлюз.

# ВИСНОВКИ

У цьому навчальному посібнику розглянуто особливості побудови захищених комп'ютерних мереж різного типу на основі використання обладнання компанії CISCO, яка є лідером-виробником мережевого обладнання у світі.

Видання складається з восьми розділів, що містять навчальний матеріал із зазначенням необхідних команд для виконання налаштування обладнання компанії CISCO та лабораторного практикуму.

Зокрема, наведені основи мережевої безпеки загалом, водночас із базовими поняттями інформаційної безпеки як такої та методами, засобами та політиками для її реалізації. Окремий розділ присвячено класифікації різних типв атак на комп'ютерні та інформаційні системи, запропоновані шляхи вирішення задач безпеки. Як відомо, важливою складовою організації захищених мереж є методи криптограічного захисту інформації, тому автори не могли оминути увагою симетричні, несиметричні алгоритми та однобічні функції шифрування даних. Компанія Cisco задля вирішення задач аутентифікації пропонує свій локальний та серверний AAA-сервіс, і у цьому навчальному посібнику для вивчення цього сервісу авторами пропонуються поняття та принципи мережевих служб аутентифікації, дайджест-аутентифікація, а також на основі паролів та сертифікатів. З метою повноти охоплення галузі організації безпеки систем та мереж запропоновані основні засоби організації безпечного периметра мережі водночас з їхніми міжмерережевими екранами, проксі-серверами, системами виявлення та запобігання вторгненням, а також принципи організації демілітаризованих зон. Окремо виділено базові поняття, характеристики та принципи фільтрації трафіку, наведено усі типи ACL та надано інструменти для їхньої організаї, контролю й конфігурування. Також висвітлені питання налаштування та застосування міжмережевих екранів. Важливим є детальний опис технологій захисту мережевих пристроїв, що охоплює таке: захист прикордонних маршрутизаторів, конфігурування захищеного адміністративного доступу (у тому числі VTY та SSH); гранулювання прав адміністратора в Cisco IOS; моніторинг та керування пристроями. Нині актуальним є реалізація захищеного передавання даних, і у видання наведено принципи утворення захищених каналів та застосування протоколів IPsec, AH, ESP.

Доданий до навчального посібника лабораторний практикум повністю покриває навчальний матеріал та містить авторські лабораторні роботи із розрахунком широкого спектру оригінальних варіантів завдань.

Цей навчальний посібник розрахований на студентів вищих навчальних закладів зі спеціальностями 123 – «Комп'ютерна інженерія», 125 – «Кібербезпека» і допоможе ґрунтовно підготувати фахівців з мережевої безпеки для роботи на обладнанні компанії Cisco.

## ЛІТЕРАТУРА

1) Cisco Guide to Harden Cisco IOS Devices [Електронний ресурс] / Електронні дані. – Режим доступу :

http://www.cisco.com /c/en/us/support/docs/ip/access-lists/13608-21.html.

2) Context-Based Access Control (CBAC): Introduction and Configuration [Електронний ресурс] / Електронні дані. – Режим доступу :

http://www.cisco.com/c/en/us/support/docs/security/iosfirewall/1381432.html.

3) Zone-Based Policy Firewall Design and Application Guide [Електронний ресурс] / Електронні дані. – Режим доступу :

http://www.cisco.com/c/en/us/support/docs/security/ios-firewall98628-zone-design-guide.html.

4) Configuring Basic AAA on an Access Server [Електронний ресурс] / Електронні дані. – Режим доступу :

http://www.cisco.com/c/en/us/support/docs/security-vpn/terminal-access-controller-access-control-system-tacacs-/10384-security.html.

5) Role-Based CLI Access Server [Електронний ресурс] / Електронні дані. – Режим доступу :

http://www.cisco.com/en/US/docs/ios/12_3t/12_3t7/feature/guide/gtclivws.html.

6) Комп'ютерні мережі : навч. посіб. / О. Д. Азаров, С. М. Захарченко, О. В. Кадук та ін. – Вінниця : ВНТУ, 2013. – 500 с.

7) Олифер В. Г. Компьютерные сети. Принципы, технологии, протоколы : учебник для вузов / В. Г. Олифер, Н. А. Олифер. – СПб. : Питер, 2010. – 944 с.

8) Палмер М. Проектирование и внедрение компьютерных сетей. Учебный курс / М. Палмер, Р. Синклер. – СПб. : БХВ, 2004. – 752 с.

9) Программ Сетевой Академии Cisco CCNA 3-4. – К. : Издательский дом «Вильямс», 2007. – 994 с.

10) Таненбаум Э. Компьютерные сети / Э. Таненбаум. – СПб. : Питер, 2012. – 960 с.

11) Шаньгин В. Ф. Информационная безопасность компьютерных систем и сетей : учебное пособие / Шаньгин В. Ф. – М. : ИД «ФОРУМ», 2008. – 416 с.

*Навчальне видання*

**Захарченко Сергій Михайлович**
**Трояновська Тетяна Іванівна**
**Олександр Володимирович Бойко**

# ОСНОВИ ПОБУДОВИ ЗАХИЩЕНИХ МЕРЕЖ НА БАЗІ ОБЛАДНАННЯ КОМПАНІЇ CISCO

**Навчальний посібник**

Редактор О. Ткачук

Оригінал-макет підготовлено Т. Трояновською

Підписано до друку 19.10.2017 р.
Формат 29,7×42¼. Папір офсетний.
Гарнітура Times New Roman.
Друк різографічний. Ум. друк. арк. 7,85.
Наклад 50 (1-й запуск1-20) пр. Зам. № 2017-379

Видавець та виготовлювач
інформаційний редакційно-видавничий центр.
ВНТУ, ГНК, к. 114.
Хмельницьке шосе, 95
м. Вінниця, 21021.
Тел. (0432) 59-85-32, 59-87-38
**pres.vntu.edu.ua;**
*E-mail*: kivc.vntu@gmail.com.

Свідоцтво суб'єкта видавничої справи
серія ДК № 3516 від 01.07.2009 р.