**А. В. Жилін, О. М. Шаповал, О. А. Успенський**

# ТЕХНОЛОГІЇ ЗАХИСТУ ІНФОРМАЦІЇ В ІНФОРМАЦІЙНО-ТЕЛЕКОМУНІКАЦІЙНИХ СИСТЕМАХ

**Навчальний посібник**

**Київ - 2020**

**МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ**
**НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ**
**«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ**
**імені ІГОРЯ СІКОРСЬКОГО»**
**ІНСТИТУТ СПЕЦІАЛЬНОГО ЗВ'ЯЗКУ**
**ТА ЗАХИСТУ ІНФОРМАЦІЇ**

**А. В. Жилін, О. М. Шаповал, О. А. Успенський**

# ТЕХНОЛОГІЇ ЗАХИСТУ ІНФОРМАЦІЇ В ІНФОРМАЦІЙНО-ТЕЛЕКОМУНІКАЦІЙНИХ СИСТЕМАХ

**Навчальний посібник**

*Рекомендовано Вченою радою ІСЗЗІ КПІ ім. Ігоря Сікорського для використання в освітньому процесі з підготовки фахівців першого (бакалаврського) рівня вищої освіти зі спеціальностей «Комп'ютерні науки» та «Кібербезпека»*

Київ
КПІ ім. Ігоря Сікорського
2020

УДК 004.056.5 (075.8)
Ж72

*Рекомендовано Вченою радою ІСЗЗІ*
*КПІ ім. Ігоря Сікорського*
*(Протокол № 12 від 27.08.2020 р.)*

**Рецензенти:** *І. Ю. Субач*, д-р техн. наук, доц.
*О. М. Рома*, д-р техн. наук, ст. наук. співроб.

**Жилін А. В.**

Ж72 Технології захисту інформації в інформаційно-телекомунікаційних системах : навч. посіб. / А. В. Жилін, О. М. Шаповал, О. А. Успенський ; ІСЗЗІ КПІ ім. Ігоря Сікорського. – Київ : КПІ ім. Ігоря Сікорського, Вид-во «Політехніка», 2021. – 213 с.

Розглянуто базові питання технологій захисту інформації в ІТС, аналізу та моніторингу мережі, технологій міжмережевих екранів та систем виявлення атак, сучасних і перспективних технологій захисту інформації в ІТС та віртуальних приватних мереж.

Навчальний посібник призначений курсантам та студентам, які навчаються за спеціальністю «Кібербезпека» і вивчають дисципліну «Технології захисту інформації в ІТС», а також курсантам та студентам, які навчаються за спеціальністю «Комп'ютерні науки» і вивчають дисципліну «Безпека інформаційних систем» із циклів професійної підготовки.

**УДК 004.056.5 (075.8)**

ЗМІСТ

# ПЕРЕДМОВА

Цифрова епоха інформаційного суспільства привнесла з собою не тільки позитивні трансформації. Основними недоліками стали ризики кіберінцидентів, які набувають все більших масштабів. Так, за один день, в будь якій корпоративній інформаційно-телекомунікаційній системі виникає велика кількість подій, частина з яких є кіберінцидентами. Наявність кіберінцидентів в інформаційно телекомунікаційній системі свідчить про розвиток кібератак або вже їх здійснення. Наслідки цих атак можуть призвести до несанкціонованого доступу до інформації, яка циркулює в мережі, або ж до позбавлення її працездатності. Проте є низка заходів та технологій, які вже сьогодні можуть використовуватись для запобігання, виявлення кібератак та зменшення їх впливу на мережу. Саме огляду цих технологій й присвячений даний посібник.

Цей посібник підготовлено для бакалаврів ІСЗЗІ КПІ ім. Ігоря Сікорського, що навчаються за спеціальностями «Комп'ютерні науки» та «Кібербезпека». До нього включено вивчення базових питань технологій захисту інформації в ІТС, аналізу та моніторингу мережі, технологій міжмережевих екранів та систем виявлення атак, сучасних й перспективних технологій захисту інформації в ІТС та віртуальних приватних мереж. Приведено підходи до забезпечення захисту інформації в ІТС, надано дефініцію понять подія, кіберінцидент, кібератака та розглянуто методи виявлення атак в комп’ютерних мережах.

Посібник вимагає від читача знання основ побудови та експлуатації комп’ютерних мереж та операційних систем у межах програми підготовки бакалавра з «Комп'ютерних наук» та «Кібербезпеки».

Матеріал посібника поділено на п’ять розділів. Перший розділ присвячений введенню до технологій захисту інформації в ІТС. У другому розділі розглядаються питання аналізу та моніторингу мережі. В третьому розділі надано відомості щодо технологій міжмережевих екранів та систем виявлення атак. Четвертий розділ описує сучасні й перспективні технології захисту інформації в ІТС. П’ятий розділ містить питання технологій віртуальних приватних мереж. Кожен розділ містить окрему нумерацію рисунків та таблиць. Матеріали посібника базуються на власному досвіді авторів, а також використовуються літературні джерела, наведені в кінці посібника.

Розділ 2 – автор О.А. Успенський, розділ 5 – автор О.М. Шаповал, решта матеріалу – автор Жилін А.В.

# 1. ВВЕДЕННЯ ДО ТЕХНОЛОГІЙ ЗАХИСТУ ІНФОРМАЦІЇ В ІТС

## 1.1 Підходи до забезпечення захисту інформації в ІТС

Існує два підходи до проблеми забезпечення безпеки інформаційно-телекомунікаційних систем (ІТС): «фрагментарний» і комплексний.

**«Фрагментарний»** підхід спрямований на протидію чітко визначеним загрозам в заданих умовах. Як приклади реалізації такого підходу можна вказати окремі засоби управління доступом, автономні засоби шифрування, спеціалізовані антивірусні програми, тощо. Перевагою такого підходу є висока вибірковість до конкретної загрози. Істотний недолік - відсутність єдиного захищеного середовища обробки інформації. Фрагментарні заходи захисту інформації забезпечують захист конкретних об'єктів ІТС тільки від конкретної загрози. Навіть невелика видозміна загрози веде до втрати ефективності захисту.

**Комплексний** підхід орієнтований на створення захищеного середовища обробки інформації в ІТС, що об'єднує в єдиний комплекс різнорідні заходи протидії загрозам. Організація захищеного середовища обробки інформації дозволяє гарантувати певний рівень безпеки ІТС, що є безперечною перевагою комплексного підходу. До недоліків цього підходу відносяться: обмеження на свободу дій користувачів ІТС, чутливість до помилок встановлення і налаштування засобів захисту, складність управління.

Комплексний підхід застосовують для захисту ІТС великих організацій або невеликих ІТС, що виконують відповідальні завдання або обробляють особливо важливу інформацію. Порушення безпеки інформації в ІТС великих організацій може завдати величезної матеріальної шкоди як самим організаціям, так і їх клієнтам. Тому такі організації змушені приділяти особливу увагу гарантіям безпеки і реалізовувати комплексний захист. Комплексного підходу дотримуються більшість державних і великих комерційних підприємств і установ. Цей підхід знайшов своє відображення в різних стандартах.

Комплексний підхід до проблеми забезпечення безпеки заснований на розробленій для конкретної ІТС політики безпеки. Політика безпеки регламентує ефективну роботу засобів захисту ІТС. Вона охоплює всі особливості процесу обробки інформації, визначаючи поведінку системи в різних ситуаціях. Надійна система безпеки мережі не може бути створена без ефективної політики мережевої безпеки.

Для захисту інтересів суб'єктів інформаційних відносин необхідно поєднувати заходи наступних рівнів:

- законодавчого (стандарти, закони, нормативні акти, тощо);
- адміністративно-організаційного (дії загального характеру, що застосовується керівництвом організації, і конкретні заходи безпеки, що мають справу з людьми);
- програмно-технічного (конкретні технічні заходи).

Заходи **законодавчого рівня** дуже важливі для забезпечення інформаційної безпеки. До цього рівня відноситься комплекс заходів, спрямованих на створення і підтримання в суспільстві негативного (в тому числі карального) відношення до порушень і порушників інформаційної безпеки.

Заходи **адміністративно-організаційного рівня**. Адміністрація організації повинна усвідомлювати необхідність підтримки режиму безпеки і виділяти на ці цілі відповідні ресурси. Основою заходів захисту адміністративно-організаційного рівня є політика безпеки і комплекс організаційних заходів.

До комплексу організаційних заходів відносяться заходи безпеки, що реалізуються людьми. Виділяють наступні групи організаційних заходів:

- управління персоналом;
- фізичний захист;
- підтримання працездатності;
- реагування на порушення режиму безпеки;
- планування відновлювальних робіт.

Для кожної групи в кожній організації має існувати набір регламентів, що визначають дії персоналу.

Заходи і засоби **програмно-технічного рівня**. Для підтримки режиму інформаційної безпеки особливо важливі заходи програмно-технічного рівня, оскільки основна загроза комп'ютерних систем виходить від них самих: відмова обладнання, помилки програмного забезпечення, помилки користувачів і адміністраторів, тощо. В рамках сучасних інформаційних систем повинні бути доступні наступні механізми безпеки :

- ідентифікація та перевірка справжності користувачів;
- управління доступом;
- протоколювання і аудит;
- криптографія;
- екранування;
- забезпечення високої доступності.

**Необхідність застосування стандартів.** Інформаційні системи (ІС) організацій майже завжди побудовані на основі програмних і апаратних продуктів різних виробників. Поки немає жодної компанії-розробника, яка надала б споживачеві повний перелік засобів (від апаратних до програмних) для побудови сучасної ІС. Щоб забезпечити в різнорідної ІС надійний захист інформації потрібні фахівці високої кваліфікації, які повинні відповідати за безпеку кожного компонента ІС: правильно їх налаштовувати, постійно відслідковувати зміни, контролювати роботу користувачів. Очевидно, що чим різноманітніше ІС, тим складніше забезпечити її безпеку. Достаток в корпоративних мережах і системах пристроїв захисту, міжмережевих екранів (МЕ), шлюзів і VPN, а також зростаючий попит на доступ до корпоративних даних з боку співробітників, партнерів і замовників призводять до створення

складного середовища захисту, важкого для управління, а іноді і несумісного між собою .

Інтероперабельність (сумісність) продуктів захисту є невід'ємною вимогою для ІТС. Для більшості гетерогенних (неоднорідних) середовищ важливо забезпечити узгоджену взаємодію з продуктами інших виробників. Прийняте організацією рішення безпеки має гарантувати захист на всіх платформах в рамках цієї організації. Тому цілком очевидна потреба в застосуванні єдиного набору стандартів як постачальниками засобів захисту, так і компаніями - системними інтеграторами і організаціями, які виступають в якості замовників систем безпеки для своїх корпоративних мереж і систем.

Стандарти утворюють понятійний базис, на якому будуються всі роботи по забезпеченню інформаційної безпеки, і як один з критеріїв, яким має слідувати управління безпекою. Стандарти є необхідною основою, що забезпечує сумісність продуктів різних виробників, що надзвичайно важливо при створенні систем мережевої безпеки в гетерогенних середовищах.

Комплексний підхід до вирішення проблеми забезпечення безпеки, раціональне поєднанні законодавчих, адміністративно-організаційних і програмно-технічних заходів і обов'язкове дотримання промислових, національних і міжнародних стандартів - це той фундамент, на якому будується вся система захисту корпоративних мереж.

Взагалі, стандарти інформаційної безпеки можна поділити на три групи стандартів (фреймворків), які відрізняються одна від одної задачами й цілями застосування – контрольні (від англійського control, яке українською звучить як «міра захисту»), програмні та ризикові.

**Перша група** фреймоврків (**контрольні**) представлена не тільки іноземними прикладами - NIST SP800-53, CIS Controls (колишній SANS Top20), а це також вітчизняні документи НД ТЗІ. Ці фреймворки потрібні для вирішення важливих «технічних» завдань по ІБ:

- визначення базового набору захисних заходів, потрібних конкретній організації;
- оцінка поточного стану рівня технічного захисту інформації;
- пріоритезації захисних заходів і складання дорожньої карти розвитку.

Каталоги захисних заходів великі і налічують близько двох сотень найменувань, що в умовах браку ресурсів на їх впровадження породжують проблему вибору та пріоритезації тих заходів, що можуть дати максимальний ефект. В австралійських вимогах по інформаційній безпеці (Australian Government Information Security Manual) є така пріоритезація. Є Топ 4, Топ 8 і Топ 35 захисних заходів, які дозволяють досягти 80%, 90% і 98% результату. Аналогічна функція є і у NIST SP800-53 - проти кожного захисного заходу в полі "Пріоритет" стоїть значення від 0 до 3, що означають важливість реалізованого заходу.

В **NIST SP800-53r4** для упорядкування та забезпечення структурного підходу автори документа передбачили поділ контролів на різні типи, в залежності від їх призначення:

Common. Основні контролі, які можуть успадковуватися різними системами і мають вплив поза масштабів окремо взятої ІС. Система успадковує контроль безпеки в разі, якщо він здійснює свою функцію безпеки в цій ІС, проте був розроблений, реалізований, оцінений, авторизований поза цією ІС.

System-specific. Контроль знаходиться в зоні відповідальності власників конкретної ІС.

Hybrid. Частина контролю функціонує в якості загального, а частина в якості системного контролю. Наприклад, контроль IR-1 може визначати політики реагування на інциденти в масштабах всієї організації, проте конкретні процедури реагування визначаються для окремих систем.

Автори вводять поділ контролів на загальні, змішані і системні, так як в певних ситуаціях це може забезпечити організації скорочення вартості процесів реалізації і оцінки, а також забезпечити цілісність підходу в масштабах організації. Це логічне рішення дозволяє, наприклад, спростити процес визначення відповідальності за конкретний контроль і визначити сфери, в яких контроль буде ефективно здійснювати свою роботу.

Зокрема **NIST SP800-53r4** має наступні контролі:

- AT Awareness and Training (Обізнаність і навчання).
- AU Audit and Accountability (Аудит і звітність).
- CA Security Assessment and Authorization (Авторизація та оцінка безпеки).
- CM Configuration Management (Управління конфігурацією).
- CP Contingency Planning (Планування безперервності бізнесу).
- IA Identification and Authentication (Ідентифікація та аутентифікація).
- IR Incident Response (Реагування на інциденти).
- MA Maintenance (Обслуговування / технічна підтримка).
- MP Media Protection (Захист носіїв інформації).
- PE Physical and Environmental Protection (Захист від стихійних лих і фізична безпека)
- PL Planning (Планування).
- PS Personnel Security (Безпека персоналу).
- RA Risk Assessment (Оцінка ризиків).
- SA System and Services Acquisition (Придбання систем і сервісів).
- SC System and Communications Protection (Захист систем і комунікацій).
- SI System and Information Integrity (Інформаційна та системна цілісність).
- PM Program Management (Управління програмою ІБ).

Що стосується **CIS Controls**, то в цьому фреймоворку міри захисту розділені на три категорії, які враховують сучасний ландшафт кіберзагроз – базові, фундаментальні, організаційні.

**Базові**. У цій категорії містяться рекомендації, необхідні для забезпечення інформаційної безпеки організації. У цю категорію входять такі пункти:

- інвентаризація авторизованих і неавторизованих пристроїв;
- інвентаризація авторизованого і неавторизованого програмного забезпечення;
- засоби управління вразливостями;
- використання адміністративних привілеїв;
- захищені конфігурації для мобільних пристроїв, ноутбуків, АРМ і серверів;
- обслуговування, моніторинг та аналіз журналів аудиту.

**Фундаментальні**. У цю категорію входять рекомендації, необхідні для застосування кращих практик для забезпечення переваг і використання передових технологій кібербезпеки. У цю категорію входять такі пункти:

- захист електронної пошти та веб-браузера;
- захист від шкідливих програм;
- обмеження і контроль мережевих портів;
- можливість відновлення даних;
- захищені конфігурації для мережевих пристроїв;
- захист периметра;
- захист даних;
- контроль доступу;
- контроль доступу бездротових мереж;
- контроль облікових записів.

**Організаційні**. У цю категорію входять рекомендації, орієнтовані на організаційні процеси і адміністративні заходи, пов'язані із забезпеченням інформаційної безпеки, з метою підвищення обізнаності персоналу та проведення Red Team/Blue Team операцій. У цю категорію входять такі пункти:

- контроль рівня обізнаності персоналу;
- контроль прикладного програмного забезпечення;
- реагування на інциденти;
- тестування на проникнення/Red Team.

Ці рекомендації дозволяють скласти чіткі і пріоритетні керівництва для вирішення завдань щодо забезпечення інформаційної безпеки організації і можуть бути використані як основа політик інформаційної безпеки.

«**Програмні**» фреймворки призначені для:

- оцінки поточного стану програми інформаційної безпеки в органзації;
- побудови всебічної програми інформаційної безпеки;

- вимірювання зрілості програми інформаційної безпеки і забезпечення можливості бенчмаркінгу (порівняння себе з іншими по галузі).

Найвідомішим в Європі і популярним фреймворком є ISO 27001, а в Північній Америці - NIST CSF. Перший цікавий тим, що у нього безліч розширень під різні завдання. Наприклад, є ISO 27019 для енергетики, ISO 27011 для телекому або ISO 27701 для персональних даних. На відміну від технічних заходів захисту, в ISO 27001 або NIST CSF заходи носять більш високорівневий характер - взаємовідношення з постачальниками, вибір засобів захисту, управління активами, робота з персоналом, підвищення обізнаності, управління інцидентами і т.п. Частина цих заходів є і в НД ТЗІ, але вони все-таки орієнтовані саме на технічну складову, ніж на вибудовування програми.

Якщо перші два типи фреймворків добре доповнюють один одного, то третій, **ризиковий**, можна застосовувати в дуже зрілих організаціях, в яких впроваджено управління ризиками ІБ. В цілому даний тип фреймворків, наприклад FAIR або серія NIST SP 800-39/37/30, дозволяє:

- ідентифікувати і вимірювати ризики ІБ;
- пріоритезувати захисні заходи;
- визначити ключові етапи управління ризиками;
- структурувати програму управління ризиками.

Окрім зазначених груп стандартів інформаційної безпеки також можна виділити галузеві стандарти, вимоги яких встановлюються певною галуззю, наприклад PCI DSS, ISA-99, TR-99.00.XX, тощо, та технологічні стандарти, що описують різні технології ІБ, наприклад, X.509, різні RFC, PKCS, біометричні стандарти і безліч інших для сотень різних технологій.

В загальному випадку, у той час як NIST визначає сім'ї контролю безпеки, кожна сім'я належить до класу – адміністративного (керуючого), оперативного (фізичного) або технічного.

Адміністративні (керуючі) контролі безпеки зосереджуються на питаннях, які потрібно вирішити керівництву, та на зниженні ризиків. Оперативні (фізичні) звертають увагу на правильній реалізації та використання політик безпеки. Відносяться до механізмів і процедур, які в основному реалізуються людьми, а не системами. Технічні ж контролі безпеки зосереджуються на правильному використанні апаратних та програмних можливостей безпеки в системах.

При чому слід пам’ятати, що виконання 20 відсотків зазначених вимог можуть дати 80 відсотків ефективності системи захисту (так званий принцип Парето). Далі наведемо цей список важливих вимог безпеки:

- Інвентаризуйте свої пристрої. Необхідно визначати, що підключено до мережі, який із пристроїв дозволений, а який ні. Використовуйте активні та пасивні сканери, щоб зрозуміти це якомога всебічніше. Розробіть політику щодо відстеження та керування пристроями.

- Переконайтесь, що в системі може працювати лише авторизоване програмне забезпечення. Сформуйте білий список додатків.
- Переконфігуруйте конфігурації обладнання та програмного забезпечення згідно стандартів безпеки.
- Необхідна постійна оцінка вразливості та усунення недоліків - управління патчем, що охоплює ОС та сторонні додатки.
- Контрольоване використання привілеїв адміністратора - використання найменших привілеїв - лише те, що потрібно для виконання роботи, не більше, не менше. Видаліть непотрібні системні права чи дозволи.
- Реалізовувати системні та лог журнали.
- Захист електронної пошти та веб-браузера.
- Захист від зловмисних програм.
- Обмеження та контроль мережевих портів.
- Можливість відновлення даних.
- Безпечні конфігурації мережевих пристроїв.
- Захист периметру мережі.
- Захист даних.
- Контрольований доступ на основі політики мінімальних потреб.
- Контроль доступу в бездротовій мережі.
- Моніторинг та контроль облікових записів.
- Оцінка навичок безпеки та навчання.
- Безпека прикладного програмного забезпечення.
- Реагування на випадки безпеки.
- Тести на проникнення та вправи для червоної команди

**Багаторівневий підхід в побудові системи захисту ІТС (Defense in Depth).**

Побудова системи захисту ІТС повинна починатися з політики та процедур як самого зовнішнього шару. Потім заходи фізичної безпеки (наприклад, охоронці, замки на дверях тощо), захист мережі від зовнішніх мереж й внутрішня безпека мережі. В середині мережі ми стикаємося з окремими хостами, програмами і, нарешті, в основі - це дані (рис. 1.1).

В свою чергу для надійного захисту всієї мережі потрібно організувати захист кожного рівня. Але не існує єдиного способу реалізації контролів управління, перелічених у NIST SP 800-53. Багато різних технологій можуть забезпечувати контроль доступу, механізми аутентифікації, управління сеансами, тощо. Наприклад, такими інструментами та методами мережевої безпеки, що використовуються для забезпечення зовнішньої безпеки мережі, є DMZ, VPN, агрегація журналів (SIEM), аудит, тестування на проникнення, аналіз вразливостей, тощо.


Рисунок 1.1 - Багаторівневий підхід в побудові системи захисту ІТС (Defense in Depth).

Деякими інструментами та методами мережевої безпеки, що використовуються для забезпечення **безпеки мережевого периметра**, є: міжмережеві екрани, проксі сервера, системи агрегації журналів (SIEM), глибокий аналіз мережевих пакетів, тестування на проникнення, аналіз вразливостей.

Для забезпечення **безпеки внутрішньої мережі** використовуються наступні інструменти та методи мережевої безпеки: системи виявлення та запобігання вторгнень (IDS), системи агрегації журналів (SIEM), аудит, тестування на проникнення, аналіз вразливостей.

Щодо забезпечення **безпеки рівня хоста** то використовують наступні інструменти та методи мережевої безпеки: засоби аутентифікації, антивірусні засоби, міжмережеві екрани, системи виявлення та запобігання вторгнень (IDS), хешування паролів, системи агрегації журналів (SIEM), аудит, тестування на проникнення, аналіз вразливостей.

Деякі інструменти та методи мережевої безпеки, що використовуються для **забезпечення безпеки додатків**, є: політика єдиного входу в додаток, фільтрація контенту, перевірка даних, аудит, системи агрегації журналів (SIEM), тестування на проникнення, аналіз вразливостей.

Для забезпечення рівня **безпеки даних** застосовуються наступні інструменти та методи мережевої безпеки: шифрування, контроль доступу, резервне копіювання, тестування на проникнення, аналіз вразливостей.

Тільки сумісне використання наведених методів та засобів захисту на кожному рівні можуть дати максимальний результат захисту інформації в ІТС.

1.2 Структуризація та зонування ІТС.

При розробці та організації мережі виникає необхідність в структуризації потоків інформації. В досягненні цієї цілі допоможе введення ієрархічної структури розподілу використовуваного обладнання і кінцевих вузлів, ґрунтуючись на їх функціональних особливостях. Типова ієрархічна модель, яка використовується при розробці мережі має три рівні – ядро (core), розподільчий рівень (distribution) і рівень доступу (access). Приклад такої структури наведений на рисунку 1.2:

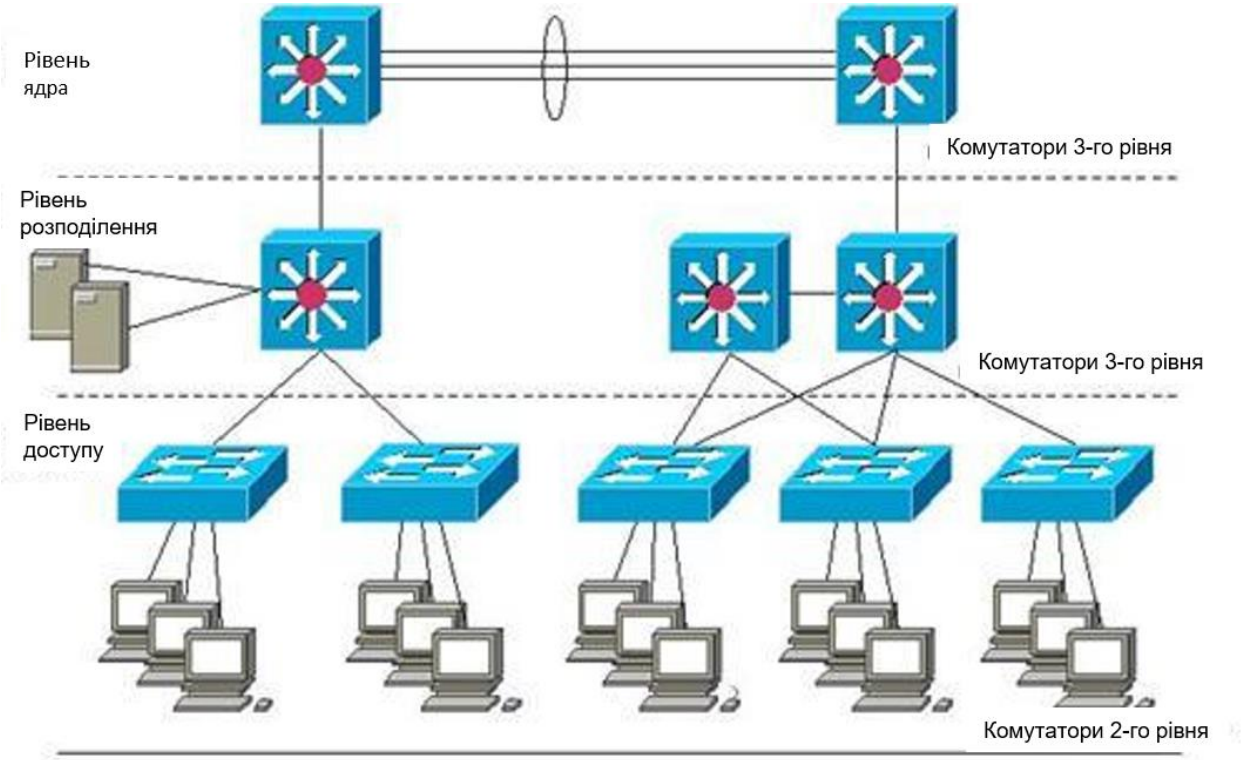



Рисунок 1.2 – Трирівнева мережа організації

**Рівень ядра** знаходиться на вершині ієрархії і відповідає за надійну і швидку передачу великих обсягів даних. Трафік, що передається через ядро, є загальним для більшості користувачів. Дані користувача оброблюються на рівні розподілу, який, при необхідності, пересилає запити до ядра. Для рівня ядра велике значення має його відмовостійкість, оскільки збій на цьому рівні може призвести до втрати зв'язності між рівнями розподілу мережі.

**Рівень розподілу** є сполучною ланкою між рівнями доступу та ядра. Залежно від способу реалізації рівень розподілу може виконувати наступні функції:

- забезпечення маршрутизації, якості обслуговування та безпеки мережі;

- агрегацію каналів;
- перехід від однієї технології до іншої.

На рівні розподілу реалізується маршрутизація пакетів і їх фільтрація (на основі списків доступу, тощо).

**Рівень доступу** управляє доступом користувачів і робочих груп до ресурсів об'єднаної мережі. На рівні доступу забезпечується підключення кінцевих робочих станцій. Основним завданням рівня доступу є створення точок входу/виходу користувачів в мережу. Рівень виконує наступні функції:

- управління доступом користувачів і політикою мережі;
- створення окремих доменів колізій (сегментація);
- підключення робочих груп до рівня розподілу;
- використання технології комутованих локальних мереж.

Якщо розглядати типову мережу невеликої організації, що займає кілька поверхів однієї будівлі, то рівень розподілу буде відповідати обладнанню, що об'єднує комутатори кожного поверху, а рівень ядра - активному обладнанню, що розміщується зазвичай в головній серверній.

Класична схема ієрархічної структури (рис. 1.2) на практиці часто модифікується з урахуванням специфіки організації, обладнання і т. д. Так, в залежності від розмірів організації, може бути відсутнім будь-який рівень, і структура мережі стане дворівневою. Маршрутизацію даних можна реалізувати на рівні ядра, а обладнання рівня розподілу буде тільки пересилати дані всередині сегмента мережі. Все залежить від вирішуваних завдань, розподілу потоків інформації і вимог, що пред'являються до інформаційної системи.

Часто в схемі мережі виділяють серверну ферму. Серверна ферма (англ. Server farm ) – з'єднання декількох серверів між собою в єдину мережу передачі даних, в якій вони працюють як єдине ціле. Принципово серверна ферма являє собою звичайний вузол розподілу, але реалізований на швидкодіючому обладнанні і, як правило, зі 100% резервованим рішенням.

В малих організаціях часто практикується підключення серверів безпосередньо до ядра мережі передачі даних, так як трирівнева схема більше властива великим мережам.

Для середніх і невеликих організацій найчастіше створюється дворівнева схема: існує один, зазвичай, найпотужніший комутатор, до якого підключаються як сервери, так і робочі станції. До цього комутатора підключені комутатори другого рівня, що розподіляють дані на інші робочі станції.

Переваги ієрархічної моделі:

1. Масштабованість. Модульний дизайн дає можливість легко додавати комутатори на рівень доступу без необхідності часто розширювати рівень розподілу або ядра.

2. Відмовостійкість. Вихід з ладу будь-якого з компонентів мережі, не позначається на працездатності окремих її частин.

3. Продуктивність. Використання агрегованих каналів для зв’язку між рівнями дозволяє збільшити пропускну здатність мережі.

4. Безпека. Комутатори на кожному з рівнів дозволяють виконувати контроль доступу як до самої мережі (port security), так і до різних мережевих служб при допомозі політик доступу (acl).

5. Керованість. Комутатори на кожному рівні мають схожу функціональність і тому при впровадженні нових комутаторів досить просто копіювати конфігурацію із вже налагоджених пристроїв.

6. Підтримка. Модульна архітектура дозволяє досить легко розширювати мережу, не вдаючись до глобальної заміни комутаторів у разі нестачі портів, на кожному з рівнів, включаючи ядро.

Якщо введення ієрархічної структури розподілу використовуваного обладнання і кінцевих вузлів відображає функціональні особливості організації, то зонування мережевої безпеки призначене для управління та розгортання архітектури організації в різних зонах безпеки (див. рис. 1.3). В цих зонах безпеки встановлюються мережеві пристрої, що мають визначені пристрої захисту. Різні зони безпеки можуть мати подібні або різні рівні захисту. Визначення різних зон безпеки з їхніми рівнями захисту допомагають в моніторингу та контролю вхідного та вихідного трафіку мережі.

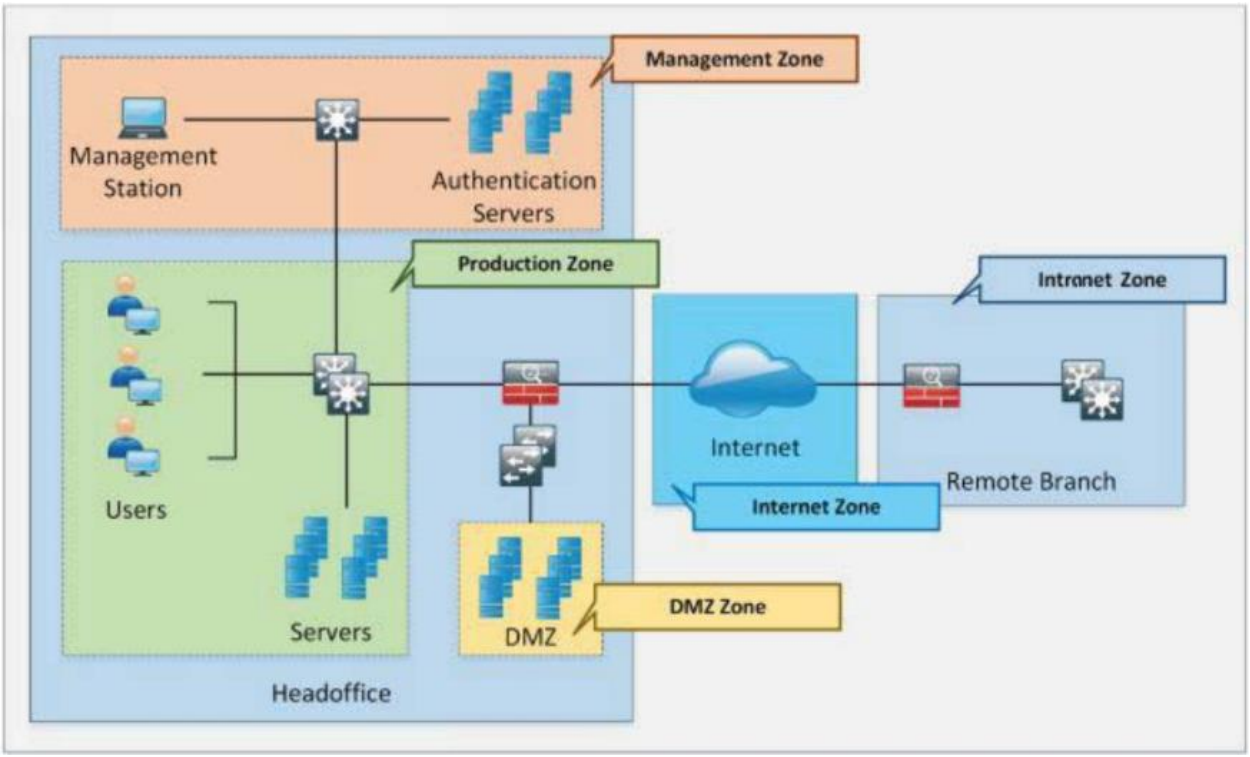



Рисунок 1.3 – Зонування мережевої безпеки

Відповідно виділяють наступні зони:

**Management Zone** – Зона управління.

В залежності від потреб безпеки, створюється зона управління, яка дозволить визначеному персоналу отримати доступ до визначених функцій і задач. В зоні управління доступ суворо контролюється і надається невеликій кількості авторизованих користувачів. Доступ до однієї зони області не обов'язково відноситься до доступу другої області зони.

**Production Zone** – Виробнича зона.

Виробнича зона підтримує визначені функції, доступ до яких має суворо контролюватися, прямий доступ з неконтрольованої мережі не повинен бути дозволений. У великій організації кілька мережевих зон можуть бути визначені як обмежені. Зона обмеженого доступу зазвичай обмежена одним або декількома міжмережевими екранами, які фільтрують вхідний і вихідний трафік.

**DMZ Zone** – Демілітаризована зона.

В більшості випадків, організації потрібно мати у себе деякі мережеві ресурси, до яких відкритий доступ з мережі Інтернет. Такими є поштовий, DNS і WEB сервери. Механізм їх роботи допускає, що до них повинен бути дозволений вільний або слабо обмежений доступ з Інтернету. Відповідно ймовірність їх зламу вища, ніж інших комп'ютерів мережі. Із цієї причини розміщати їх усередині зони, що захищається, недоцільно з погляду безпеки, тому що у випадку зламу вони можуть стати воротами для атаки внутрішніх комп'ютерів. Для мінімізації ризику і збереження функціональності такі сервери встановлюють за основним шлюзом мережі, але перед міжмережевим екраном, що забезпечує захист внутрішніх комп'ютерів. Логічну область їх розміщення називають демілітаризованою зоною.

**Internet Zone** – Інтернет зона

Зазвичай, Інтернет зона – це частина глобального Інтернету, яка знаходиться за межами організації. Являється ненадійною зоною, є найбільш вразливою до порушення безпеки, оскільки можуть бути відсутні елементи управління для блокування або недостатня їх кількість.

**Intranet Zone** – Інтранет зона

Інтранет зона, яка може знаходитись за одним чи декількома мережевими екранами не сильно обмежена у використанні, але має відповідний діапазон контролю, для гарантій того, що мережевий трафік не ставить під загрозу роботу визначених функцій. Щоб максимізувати продуктивність передачі даних або доступність визначених компонентів чи додатків, варто розмістити в Інтранет зоні такі компоненти як сервер бази даних або сервер каталогів і впевнитись, що відсутня загроза безпеки при доступі до цих компонентів.

1.3 Дефініція визначень подія, кіберінцидент, кібератака.

Відповідно до Закону України «Про основні засади забезпечення кібербезпеки України» к**іберпростір** - середовище, яке надає можливість для здійснення комунікацій або здійснення реалізації суспільних відносин, утворене

в результаті функціонування сумісних комунікаційних систем та забезпечення електронних комунікацій з використанням мережі Інтернет або інших глобальних мереж передачі даних. В кіберпросторі наявні потенційно можливі явища і чинники, що створюють небезпеку національним інтересам держави. Основними з них є кіберінциденти та кібератаки. Але кіберінцидент зазвичай характеризується певними відхиленнями в типовій поведінці об'єкта, які називаються аномаліями.

**Аномалія** - відступ або ухилення від правила, тому аномальним називають все, що відступає або уникає правильного або нормального.

При виявленні мережевої аномалії, з метою прийняття рішення про подальші дії, необхідно ретельно вивчити її природу, потенційну небезпеку та можливі наслідки, тобто вирішити задачу **класифікації**. На рис.1.4 відображено класифікацію мережевих аномалій.

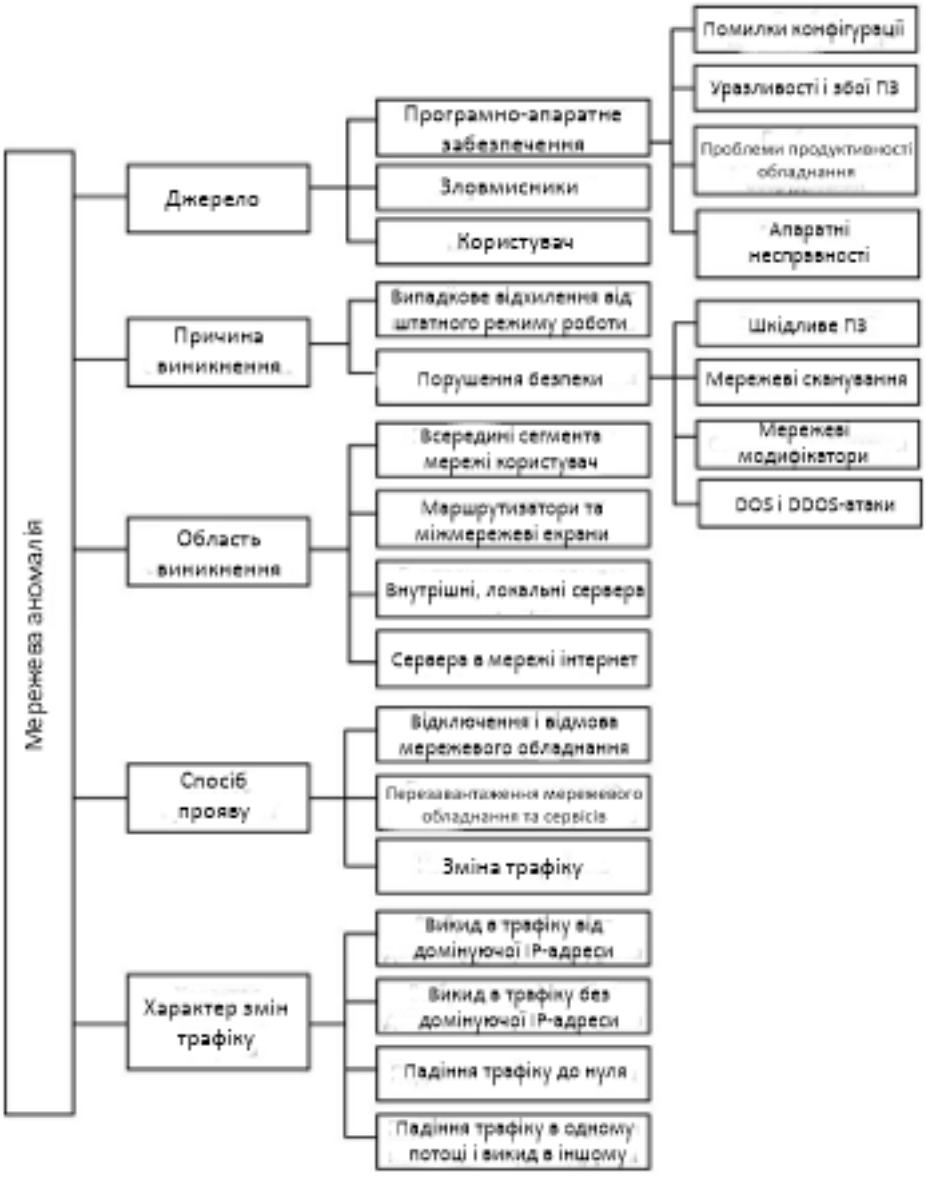



Рисунок 1.4 – Класифікація мережевих аномалій

**Інцидент кібербезпеки (кіберінцидент)** - подія або ряд несприятливих подій ненавмисного характеру (природного, технічного, технологічного, помилкового, у тому числі внаслідок дії людського фактора) та/або таких, що мають ознаки можливої (потенційної) кібератаки, які становлять загрозу безпеці систем електронних комунікацій, систем управління технологічними процесами, створюють імовірність порушення штатного режиму функціонування таких систем (у тому числі зриву та/або блокування роботи системи, та/або несанкціонованого управління її ресурсами), ставлять під загрозу безпеку (захищеність) електронних інформаційних ресурсів.

На рисунку 1.5 зображено діаграму виникнення інцидентів.

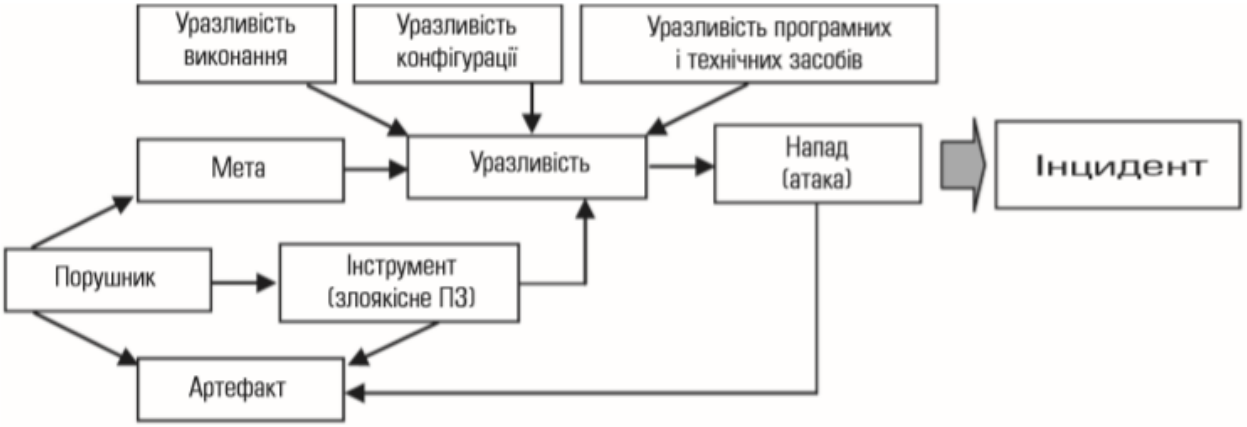



Рисунок 1.5 – Діаграма виникнення інцидентів

**Кіберзагроза** (на рисунку порушник) – наявні та потенційно можливі явища і фактори, що створюють небезпеку життєво важливим національним інтересам України у кіберпросторі, справляють негативний вплив на стан кібербезпеки держави, кібербезпеку та кіберзахист її об’єктів.

**Кіберзагроза** – фактори (події, явища), які мають місце або можуть виникнути в кіберпросторі, за умови умисного цілеспрямованого або випадкового впливів та створити небезпеку порушення процесів управління і передачі інформації, що відбуваються у кібернетичних системах різних сфер (соціальної, технічної, соціотехнічної), або можуть зашкодити елементам таких систем. Як приклад класифікації – **WASC Threat Classification, OWASP Top 10.**

**Загроза (threat)** – будь-які обставини або події, що можуть бути причиною порушення політики безпеки інформації і/або нанесення збитків АС (НД ТЗІ 1.1-003–99 «Термінологія в галузі захисту ін-формації в комп’ютерних системах від несанкціонованого доступу»).

Таблиця 1.1

Системна класифікація загроз безпеці інформації

| Параметри класифікації | Значення параметрів | Зміст значення параметрів |
|---|---|---|
| Види | Фізична цілісність<br>Логічна структура<br>Зміст<br>Конфіденційність<br>Право власності | Знищення (викривлення)<br>Викривлення структури<br>Несанкціонована модифікація<br>Несанкціоноване отримання<br>Привласнення чужого права |
| Природа походження | Випадкова<br><br>Навмисна | Відмови, збої, помилки, стихійні лиха, побічні впливи<br>Злочинні дії людей |
| Передумова появи | Об'єктивні<br><br><br>Суб'єктивні | Кількісна недостатність еле-ментів системи, якісна недос-татність елементів системи<br>Розвідувальні органи іноземних держав, промисловий шпіонаж, кримінальні елементи, «недоброякісні» співробітники |
| Джерело загроз | Люди<br>Технічні пристрої<br>Моделі, алгоритми, програми<br>Технологічні схеми обробки<br>Зовнішнє середовище | Сторонні особи, користувачі, персонал<br>Реєстрації, передачі, зберіган-ня, видачі<br>Загального призначення, прик-ладні, допоміжні<br>Ручні, інтерактивні, внутріш-ньомашинний, мережеві<br>Стан атмосфери, побічні шуми, побічні сигнали |

**Вразливість системи (system vulnerability)** — нездатність системи протистояти реалізації певної загрози або сукупності загроз.

**Вразливості класифікуються** відповідно до класу активів, до яких вони відносяться — апаратного забезпечення, програмного забезпечення, комп'ютерної мережі, персоналу, організації, тощо.

При виявлені вразливостей їх описують згідно певних вимог баз вразливостей, де вони й зберігаються для подальшого використання аналітиками безпеки. Наразі найбільш поширені наступні бази вразливостей:

- MITRE — Common Vulnerabilities and Exposures (**CVE**) та Common Weakness Enumeration (**CWE**) [http://cve.mitre.org];

- NIST — National Vulnerabilities Database (**NVD**) [http://nvd.nist.gov];
- United State Computer Emergency Readiness Team — Vulnerability Notes Database (**VND**) [http://www.us-cert.gov];
- IBM — **XForce** [http://xforce.iss.net];
- Open Source Vulnerabilities Data Base — **OSVDB** [http://osvdb.org].

**Кібератака** – спрямовані дії в кіберпросторі, які здійснюються за допомогою електронних комунікацій та спрямовані на досягнені однієї або сукупності таких цілей: порушення конфіденційності, цілісності, доступності електронних інформаційних ресурсів, що обробляються в технологічних системах, отримання несанкціонованого доступу до таких ресурсів. Тобто різниця понять кіберінцидент й кібератака визначається наявністю мети в дії та спрямованістю в здійсненні. Так кіберінцидент має ймовірнісний характер й не завжди, на противагу кібератаці, має мету здійснення. Кібератака ж виникає з наявної кіберзагрози й спрямована на експлуатацію вразливості в системі захисту або ж в самому об'єкті захисту з метою порушення однієї з властивостей об'єкту захисту.

**Класифікація кібератак**.

Атаки на інформаційні системи характеризуються декількома ознаками. Для розгляду їх сутності та умов здійснення пропонується наступна класифікація (рис. 1.6).


Рисунок 1.6 – Класифікація кібератак

За характером впливу віддалені атаки діляться на:

- пасивний вплив;
- активний вплив.

Пасивним впливом на інформаційну систему є вплив, який не має безпосереднього впливу на роботу системи, але може порушувати політику доступу до захищених даних. Саме відсутність впливу на роботу розподіленої системи призводить до того, що пасивний віддалений вплив практично неможливо виявити. Прикладом пасивного типового віддаленого впливу на

інформаційну систему служить прослуховування каналу зв'язку в мережі і перехоплення переданої інформації.

Активна дія на ресурси системи - це безпосередній вплив на роботу системи (зміна конфігурації, порушення працездатності і т. д.) і порушує прийняту в ній політику безпеки. Очевидною особливістю активного впливу, в порівнянні з пасивним, є можливість його виявлення (з більшим чи меншим ступенем складності). Прикладом результату такого впливу є відмова в обслуговуванні системи.

За метою реалізації впливу віддалені атаки можуть бути спрямовані на:

- порушення конфіденційності інформації;
- порушення цілісності інформації;
- порушення доступності системи.

Основним результатом практично будь-якого зловмисного впливу на інформаційну систему є отримання несанкціонованого доступу до інформації. Такий доступ досягається шляхом перехоплення або спотворення інформації. Можливість перехоплення інформації означає отримання до неї доступу, але неможливість її зміни (модифікації). Отже, перехоплення інформації веде до порушення її конфіденційності. Прикладом перехоплення інформації є аналіз мережевого трафіку (не слід плутати з санкціонованим проведенням аналізу мережевого трафіку).

Спотворення інформації можливо в тому випадку, якщо зловмисник має повний контроль над інформаційним потоком між об'єктами системи і/або має можливість передавати дані під ім'ям довіреного користувача. В цьому випадку, спотворення інформації веде до порушення її цілісності. Даний інформаційний руйнівний вплив відноситься до активного впливу. Прикладом віддаленої атаки, мета якої порушення цілісності інформації, може служити віддалена атака «Помилковий об'єкт мережі».

І зовсім інший вид атаки, коли отримання атакуючим несанкціонованого доступу до інформації не передбачається - це порушення працездатності (доступності) системи. В даному випадку основна мета зловмисника - різке зниження продуктивності системи об'єкта (або виведення її з ладу) на який здійснюється атака, і як наслідок - неможливість доступу користувачів до її ресурсів. Прикладом віддаленої атаки, метою якої є порушення працездатності системи, може служити віддалена атака «Відмова в обслуговуванні» (DoS-атака, Denial of Service). І найбільш популярний різновид DoS-атаки - DDoS-атака (Distributed Denial of Service), розподілена атака типу «відмова в обслуговуванні».

Для того щоб здійснити віддалений вплив на об'єкт інформаційної системи, необхідне настання певних умов. У розподілених мережах виділяються три умови для початку здійснення впливу:

- атака по запиту. В даному випадку атакуючий чекає від потенційного

об'єкта атаки передачі запиту певного типу, який і буде умовою початку здійснення впливу. Прикладом подібних запитів можуть бути DNS і ARP-запити;

- атака по події. Атакуючий веде постійне спостереження за станом операційної системи віддаленої цілі атаки і при виникненні певної події в цій системі починає вплив;
- безумовна атака. В даному випадку початок здійснення атаки безумовно по відношенню до мети атаки, тобто атака здійснюється негайно і незалежно від стану системи чи об'єкта. Атакуючий є ініціатором початку здійснення атаки.

Атаки на ІТС можна класифікувати за наявністю зв’язку з об’єктом. Розрізняють атаки з зворотним зв'язком і без зворотного зв'язку (односпрямована атака).

Зворотній зв'язок формується відповіддю об'єкта, який атакується на певний запит, відправлений до нього зловмисником, і дозволяє останньому реагувати на всі зміни, що відбуваються на об'єкті.

Метою віддаленої атаки без зворотного зв'язку не є отримання даних від об'єкта атаки. Атакуючий відправляє запити, не чекаючи відповіді від об'єкта. Тому подібну віддалену атаку називають односпрямованою. Як приклад односпрямованої атаки можна привести типову віддалену атаку «Відмова в обслуговуванні».

По розташуванню порушника щодо об'єкта атаки вплив реалізується як всередині сегментна атака, так і між сегментна.

Розглянемо ряд визначень:

- хост (host) - мережевий пристрій (найчастіше комп'ютер);
- маршрутизатор (router) - пристрій, що забезпечує маршрутизацію пакетів обміну з однієї мережі в іншу;
- підмережа (subnetwork) - сукупність хостів, що є частиною мережі, для яких маршрутизатором виділено однаковий номер підмережі;
- сегмент мережі - фізична або логічне об'єднання хостів. Наприклад, бездротовий сегмент мережі утворює сукупність хостів, підключених до точки доступу за схемою «загальна шина». При такій схемі підключення кожен хост має можливість аналізувати будь-який пакет в своєму сегменті. Аналогічна картина буде спостерігатися, якщо сегмент мережі створений не фізично, а логічно (із застосуванням віртуальних мереж VLAN).

З точки зору віддаленої атаки важливе розташування зловмисника і об'єкта атаки по відношенню один до одного, тобто в одному або в різних сегментах вони знаходяться. У разі всередині сегмента атаки, як випливає з назви, зловмисник і об'єкт атаки знаходяться в одному сегменті. При між сегментній атаці зловмисник і об'єкт атаки знаходяться в різних сегментах. Дана класифікаційна ознака дозволяє судити про так звані «ступені віддаленості» атаки.

На практиці між сегментну атаку здійснити значно важче, ніж всередині сегменту. Але між сегментна віддалена атака представляє велику небезпеку, ніж

усередину сегмента атака, так як саме віддаленість порушника від об'єкта атаки може істотно перешкодити заходам по відображенню атаки.

Імовірність вдалих атак значно збільшується в разі наявності бездротових рішень. Одне з головних відмінностей між кабельними і бездротовими мережами пов'язано з тим, що здійснювати повний контроль над областю між кінцевими точками бездротової мережі досить складно. Саме цьому найбільш поширена проблема в відкритих і некерованих середовищах, таких як бездротові мережі - це можливість анонімних атак.

За співвідношенням кількості порушників і атакованих об'єктів, атака може бути віднесена до наступних класів впливу:

- вплив «один до одного» - атака здійснюється одним зловмисником щодо одного об’єкта;
- вплив «один до багатьох» - атака здійснюється одним зловмисником щодо кількох об'єктів;
- вплив «кілька до одного»;
- вплив «кілька до багатьох».

У двох останніх випадках атака здійснюється декількома зловмисниками з різних комп'ютерів щодо одного або декількох об'єктів (розподілений або комбіноване вплив).

Шаблони атак (бази даних) — KDD Cup ’99, CAIDA, NSL-KDD та Kyoto 2006+, CAPEC.

1.4 Індикатори компрометації та модель кібератаки.

Діяльність зловмисника в комп’ютерних мережах й хостах, а також діяльність шкідливого програмного забезпечення, залишає певні сліди, які прийнято називати артефактами, або індикаторами компрометації (ІК). **ІК** – це будь-яка інформація, яку можна використовувати для виявлення компрометації системи. ІК допомагають виявляти витік даних, зараження ІС ШПЗ або інші види загроз. Відстежуючи ІК, можна виявити атаки і попередити виникнення порушень або зменшити збитки, зупинивши атаки на її ранніх стадіях.

Виділяють такі типи ІК:

- Атомарні - до них відносяться IP-адреси, e-mail-адреси, DNS-адреси, повні URL і навіть статичні фрагменти в керуючих Command & Control каналах шкідливого коду або ботнетів. Така інформація є первинною інформацією, що дозволяє ідентифікувати щось підозріле, що відбувається в мережі або на окремих комп'ютерах. Так IP або DNS-адреси, з яких може поширюватися шкідливий код, можуть змінюватися часто (до декількох разів на день). Але і ІК, що містять цю інформацію, можуть також поширюватися в міру появи нової інформації.
- Обчислювані - до такого типу індикаторів відносяться геш-суми шкідливих файлів по стандартам MD5 або SHA1, ключі реєстру, списки процесів

або інформація для декодування протоколів, за якими працює шкідливий код.

- Поведінкові - це вже не окремі індикатори, а їх набори, що описують профіль поведінки чого-небудь або кого-небудь: зловмисника, вузла, підмережі. Наприклад, таким індикатором може служити формалізоване опис наступного профілю: «невідомий часто використовує IP-адреси в діапазоні «такому» для спроби підключитися до Web-сайту в «такий» час, або невідомий відправив вкладення в форматі MS Word в повідомлення e-mail з поштового сервера, створеного 10 хвилин тому в Україні».

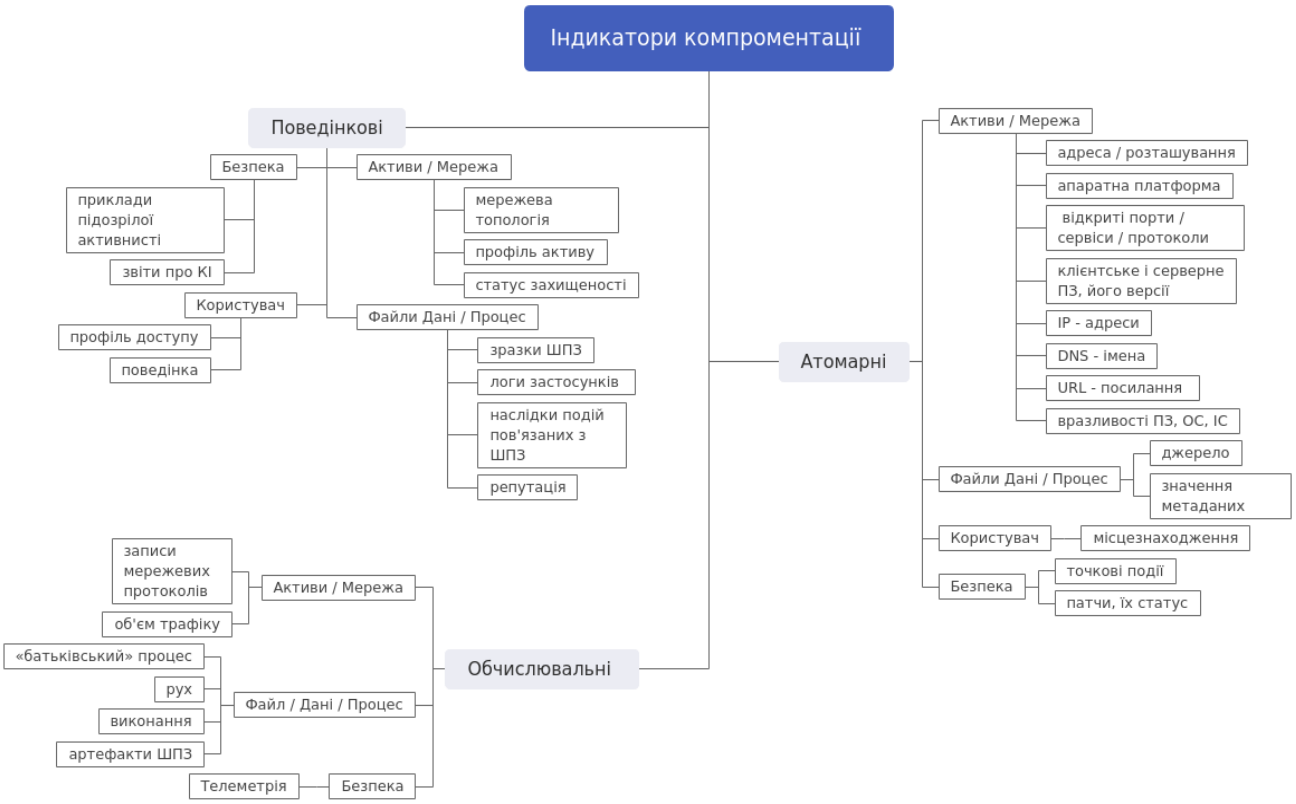



Рисунок 1.7 – Класифікація індикаторів компрометації

Можна сформувати класифікацію ІК, яка представлена у вигляді схеми (рис. 1.7):

- Атомарні
  - Активи / мережа:
    - адреса / розташування;
    - апаратна платформа;
    - відкриті порти / Сервіси / Протоколи;
    - клієнтське і серверне ПЗ і його версія, ОС;
    - попередження систем виявлення вторгнень;
    - IP-адреси;
    - DNS-імена;
    - URL-посилання;

        - вразливості ПЗ, ОС або ІС;
    - Користувач:
        - місцезнаходження;
    - Файл / Дані / Процес:
        - джерело;
        - конкретні значення метаданих, в залежності від стандарту або формату, як приклад заголовок електронної пошти;
    - Безпека:
        - точкові події;
        - патчи та їх статус;
- Обчислювальні:
    - Активи / мережа
        - записи мережевих протокол або цілі захоплені пакети;
        - об’єм трафіку;
    - Файли / дані /процес
        - рух;
        - виконання;
        - «батьківський» процес;
        - артефакти, що відносяться до ШПЗ (криптографічний хеш (КХ), ключі реєстру, імпортовані функції, мережеві запити, назви файлів до яких зверталося ШПЗ, назви файлів ШПЗ);
    - Безпека:
        - телеметрія;
- Поведінкові:
    - Активи / мережа:
        - мережева топологія;
        - профіль активу;
        - статус захищеності;
- Користувач:
        - профіль доступу;
        - поведінка;
    - Файли / дані / процес:
        - зразки виконавчих файлів ШПЗ, або документів чи електронних листів, що містять ШПЗ;
        - логи застосунків;
        - наслідки подій, такі системні виклики, пов’язані з ШПЗ;
        - репутація;
    - Безпека:
        - приклади підозрілої активності на хості або в мережі;
        - звіти про КІ.

Типовий порядок дій зловмисника для досягнення поставлених цілей

можна описати за допомогою моделі **Cyber Kill-Chain,** яка була запропонована корпорацією Lockheed Martin, як частина моделі Intelligence Driven Defense. Так, для досягнення успіху зловмисник повинен пройти усі вісім етапів (рис.1.8):

**1. Розвідка**. Цей етап може бути визначений як фаза вибору мети, виявлення особливостей організації, специфічних вимог в даній галузі, вибір технологій, вивчення активності компанії в соціальних мережах або через розсилки. По суті, зловмисник намагається знайти точки входу в систему організації, отримати відповіді на питання: Які методи атаки будуть працювати?, де, що або хто є найслабшим елементом захисту?, як вдасться діяти непомітно та результативна?.

**2. Озброєння**. Підбір інструментарію, створення експлойтів, оснащення шкідливим вмістом файлу (наприклад, PDF або MS Office) або іншого контенту, який повинен бути прочитаний/відкритий жертвою.

**3. Доставка**. Донесення шкідливого контенту до жертви, використовуючи для цього e-mail, web-сайти або USB-флешки.

**4. Зараження**. Запуск шкідливого коду, використовуючи наявні на цільовому комп'ютері уразливості, з подальшим його зараженням.

**5. Інсталяція**. Відкриття віддаленого доступу для непомітного управління і поновлення шкідливого коду, додавання функціональних модулів.

**6. Отримання управління**. Отримання оновлень з новим функціоналом ззовні, а також управляючих команд для досягнення поставлених цілей.

**7. Виконання дій**. Збір і крадіжка даних, шифрування файлів, перехоплення управління, підміна даних та інші завдання, які можуть стояти перед порушником.

**8. Знищення слідів**. Після успішно виконаної атаки, зловмиснику необхідно знищити сліди своєї активності.

Зловмисник не обов’язково повинен дотримуватись проходження усіх наведених кроків, але ефективність його діяльності при цьому може знизитись.

Модель Cyber Kill-Chain також можна використовувати не лише для побудови й здійснення кібератак, а й для організації системи захисту. Чим раніше системи захисту виявлять спрямовані дії зловмисника, тим більш ефективно працює вся система захисту.

В NIST SP 800-150 представлені наступні стратегії та методи захисту, відповідно до Cyber Kill Chain:

- Розвідка. Виконайте моніторинг і аналіз даних NetFlow, darknet і пасивного DNS, щоб виявити і досліджувати загальні схеми мережевої розвідки, такі як сканування портів або зондування. Застосовуйте заходи контррозвідки, такі як перенаправлення зловмисника в «чорну діру» або блокування певних IP-адрес або доменів.
- Озброєння. Розробка, розгортання та узгодження сигнатур високої точності на основі аналізу аномалій, виявлених корисних навантаженнях

шкідливих програм. Методи виявлення на основі сигнатур, як правило “крихкі”; зловмисники можуть ухилитись від виявлення таким шляхом незначної модифікації експлойта. Проводячи глибший аналіз виявлених шкідливих аномалій, можна створити більш точні та надійні сигнатури для виявлення нових шкідливих програм та варіантів існуючих шкідливих програм.

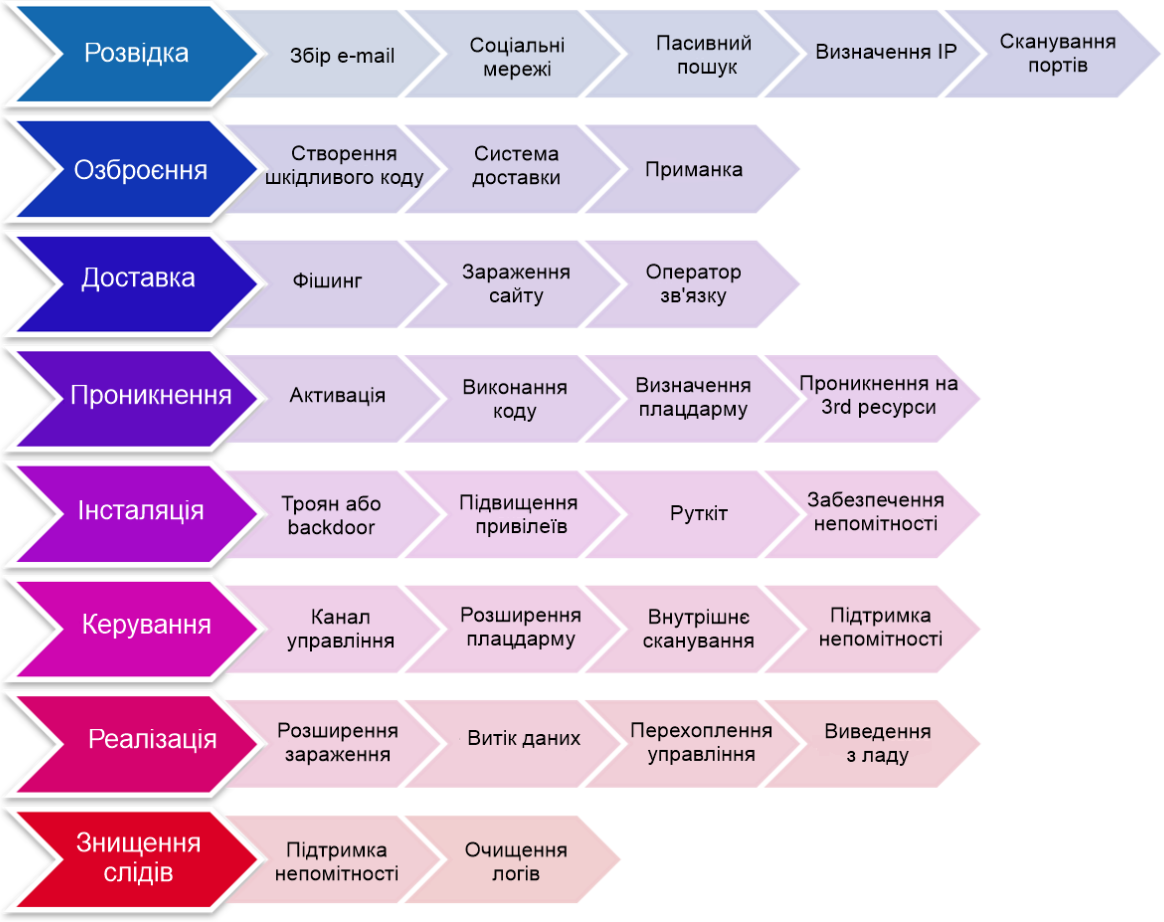



Рисунок 1.8 – Модель Cyber Kill-Chain

- Доставка. Розуміти інструменти і методи, які зловмисник використовує для доставки шкідливих корисних навантажень, а також розробляти і застосовувати детективні та захисні заходи, які порушують канали доставки зловмисників. Ці заходи можуть бути технічними (наприклад, занесення в чорний список сайту, пов’язаного з атакою типу “watering hole") або процедурними (наприклад своєчасного надання обізнаності про виникаючі загрози).

- Доставка. Розуміти інструменти і методи, які зловмисник використовує для доставки шкідливих корисних навантажень, а також розробляти і застосовувати детективні та захисні заходи, які порушують канали доставки зловмисників. Ці заходи можуть бути технічними (наприклад, занесення в чорний список сайту, пов’язаного з атакою типу “watering hole") або процедурними (наприклад своєчасного надання обізнаності про виникаючі загрози).

- Проникнення. Протидія спробам нульового дня шляхом розгортання засобів захисту, які допомагають запобігти впровадження коду зловмисниками в працюючу програму, використання умов переповнення буфера, впровадження команд операційної системи або використання слабких місць контролю доступу для розширення системи. Організації можуть також використовувати розширене моделювання загроз для характеристики своєї поверхні атаки і використовувати нечітке тестування для виявлення вразливості в ймовірних напрямках атаки.
- Інсталяція (Встановлення). Виявлення та активне реагування на недавно встановлене шкідливе програмне забезпечення за допомогою сигнатур і засобів виявлення вторгнень на рівні хоста та мережі, таких як цілісності файлів, виявлення руткітів і моніторинг змін конфігурацій.
- Керування (Командування та управління). Встановити базові показники нормальної роботи мережі і пристроїв, налаштувати внутрішню мережу для виявлення аномального вхідного та вихідного мережевого трафіку і змін в поведінці користувачів і пристроїв. Моніторинг за базової лінії забезпечують засоби моніторингу, для виявлення маяка (вихідного трафіку через регулярні інтервали), який можу бути пов'язаним з взаємодіями команд та керуючого серверу.
- Реалізація (Дія на ціль). Розгортання передових рішень щодо запобігання втрат даних, методів ухилення і спроб видалення даних для запобігання несанкціонованої передачі або копіювання конфіденційних даних.

Для розробки стратегій захисту для боротьби з цільовими кібератаками, зіставляючи елементи управління та кроками, які повинен пройти зловмисник, щоб успішно виконати кібератаку Lockheed Martin представляє наступні рівні реалізації управління:

- Виявити (Detect): визначити, коли та як зловмисник проводить розвідку проти організації або мережі.
- Заборонити (Deny): запобігти атаці шляхом запобігання розкриття інформації або несанкціонованого доступу.
- Порушити (Disrupt): змінити або залишити потік інформації або відфільтрувати дані для зловмисника.
- Погіршити (Degrade): обмежити ефективність або результативність атаки.
- Обманути (Deceive): вмішатись в атаку, використовуючи неправильні напрямки та дезінформацію.
- Стримати (Contain): обмежити область атаки визначеними сегментами мережі або організації.

Оскільки будь-яка організація обмежена в ресурсах, і відповідно не може реалізувати всі можливі засоби контролю, технології та практики кібербезпеки, то для підтримки пошуку технологій застосовуються матричні підходи. Матриця Cyber Kill Chain Controls (рис. 1.9) призначена для визначення елементів

управління, які організація впровадила на різних етапах атаки, а також того, як елемент управління допоможе порушити потік або зупинити кібератаку.

В матриці розглядається співставлення семи етапів Cyber Kill Chain з шістьма рівнями реалізації управління. Так зіставлення цих фаз з рівнями реалізації управління демонструють, як можливо реалізувати різні елементи управління або методи для запобігання інциденту. Тобто це просто показує приклад різних типів захисту, підходів, методів ідентифікації та інше.

| Phase | Detect | Deny | Disrupt | Degrade | Deceive | Contain |
|---|---|---|---|---|---|---|
| **Reconnaissance** | Web Analytics<br>Threat Intelligence<br>NIDS | Information Sharing Policy<br>Firewall ACLs | | | | |
| **Weaponization** | Threat Intelligence<br>NIDS | NIPS | | | | |
| **Delivery** | Endpoint Malware Protection | Change Management<br>Application Whitelisting<br>Proxy Filter<br>HIPS | Inline AV | Queuing | | Router ACLs<br>App-aware Firewall<br>Trust Zones<br>Inter-zone NIPS |
| **Exploitation** | Endpoint Malware Protection<br>HIDS | Secure Password<br>Patch Management | DEP | | | App-aware Firewall<br>Trust Zones<br>Inter-zone NIPS |
| **Installation** | SIEM<br>HIDS | Privilege Separation<br>Strong Passwords<br>Two-Factor Authentication | Router ACLs | | | App-aware Firewall<br>Trust Zones<br>Inter-zone NIPS |
| **Command & Control** | NIDS<br>HIDS | Firewall ACLs<br>Network Segmentation | HIPS | Tarpit | DNS Redirect | Trust Zones<br>DNS Sinkholes |
| **Actions on Objectives** | Endpoint Malware Protection | Data-at-Rest Encryption | Endpoint Malware Protection | Quality of Service | Honeypot | Incident Response |
| **Exfiltration** | DLP<br>SIEM | Egress Filtering | DLP | | | Firewall ACLs |

Рисунок 1.9 – Cyber Kill Chain Controls Matrix

В той же час при побудові систем захисту, їх налаштуванні й експлуатації потрібно враховувати які індикатори компрометації на якому етапі Cyber Kill Chain можуть залишити зловмисники (рис. – 1.10).


Рисунок 1.10 – IOCs на різних етапах Cyber Kill Chain

1.5 Методи виявлення атак в комп'ютерних мережах.

Визначенню кібератак в мережі передує виявлення кіберінцидентів на основі аналізу мережевих аномалій або зловживань в мережевому трафіку. Як було зазначено в 1.3, **аномалія** – це відступ або ухилення від правила, тому аномальним називають все, що відступає або уникає правильного або нормального. За своєю суттю аналіз аномалій дозволяє виявляти суттєві відхилення трафіку мережевих пристроїв від «нормального» профілю трафіку для даного пристрою або групи пристроїв. Як правило, шаблон «нормального» трафіку мережі складається протягом певного проміжку часу на основі статистичних даних та навчальної вибірки. Для виявлення поганої поведінки і аномалій в більшості випадків досить аналізувати основні параметри трафіку (телеметрію) і немає необхідності вивчати вміст кожного пакета. Прикладами аномалій, виявлених на основі аналізу телеметрії трафіку, є раптове збільшення інтенсивності трафіку від робочої станції або зміна структури трафіку в порівнянні зі звичайними щоденними показниками для даної мережі або пристрою.

В загальному випадку **методи виявлення кібератак** можна звести до двох схем – виявлення мережевих **аномалій** на основі показників мережевого трафіку та виявлення **зловживань** в мережевому трафіку.


Рисунок 1.11 – Схема виявлення мережевих **аномалій** на основі показників мережевого трафіку

Перевагою методу виявлення мережевих аномалій на основі показників мережевого трафіку (рис. 1.11) є агрегація й нормалізація даних за певний час, визначення ознак атрибутів й побудова профілю. В той же час це є трудомісткою задачею, невірне рішення якої може породжувати велику кількість помилок 1го (помилкова тривога) й 2го роду (пропуск події).

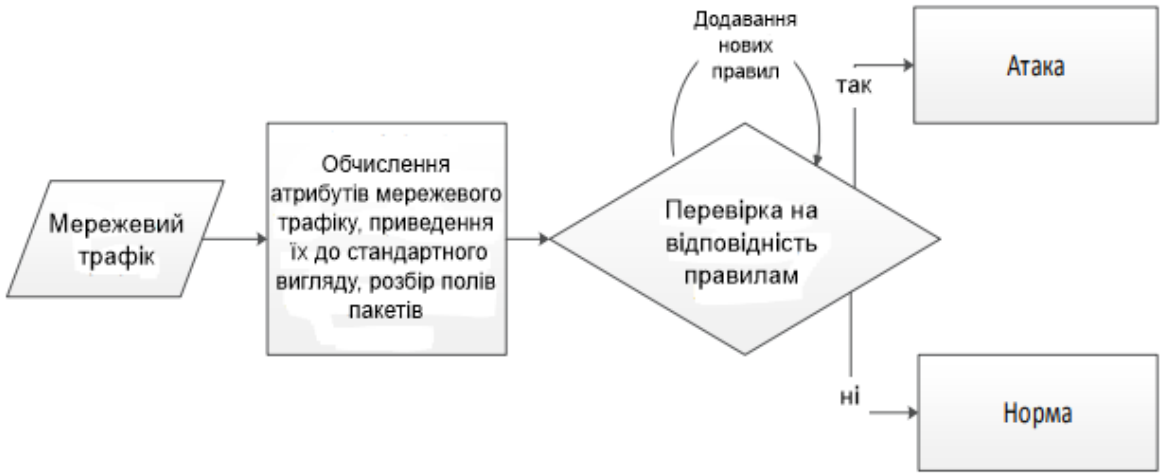



Рисунок 1.12 – Схема виявлення **зловживань** в мережевому трафіку

Перевагою методу виявлення **зловживань** в мережевому трафіку (рис. 1.12) є те, що він дозволяє ідентифікувати несанкціоновані дії, якщо є їх точне уявлення у вигляді шаблонів атак. Тут під шаблоном атаки розуміється деяка сукупність дій, які явно описують конкретну атаку (правил зіставлення, виведення), застосовуючи які до ознак і полів ідентифікованого об'єкта можна отримати однозначну відповідь про його приналежність до цієї атаки. Але в той же час є і доволі суттєві проблеми – це ефективність проектування механізму визначення правил й те що багато правил знижує швидкодію систем. Потрібні

більш загальні правила які покривають більшу кількість модифікацій атак. Також цей спосіб неефективний при виявлені невідомих атак. Представимо **класифікацію методів виявлення атак** (рис. 1.13):

**Поведінкові методи.**

Поведінковими методами називаються методи, які засновані на *використанні інформації про нормальну поведінку системи* та її порівнянні з параметрами поведінки, яка спостерігається. Представлена група методів орієнтована на побудову моделі штатного, або нормального функціонування системи або користувача. В процесі своєї роботи системи, що використовують даний підхід, порівнюють поточні показники активності з профілем нормальної діяльності, і випадок значних відхилень може розглядатися як свідчення наявності атаки.

Недоліки:

- наявність хибнопозитивних спрацювань, які пояснюються складністю точного і повного опису безлічі легітимних дій користувачів;
- необхідне проведення довгого етапу попереднього налаштування.

Поведінкові методи можуть ґрунтуватися на наступних математичних методах:

- вейвлет-аналіз;
- статистичний аналіз;
- аналіз ентропії;
- спектральний аналіз;
- фрактальний аналіз та кластерний аналіз.

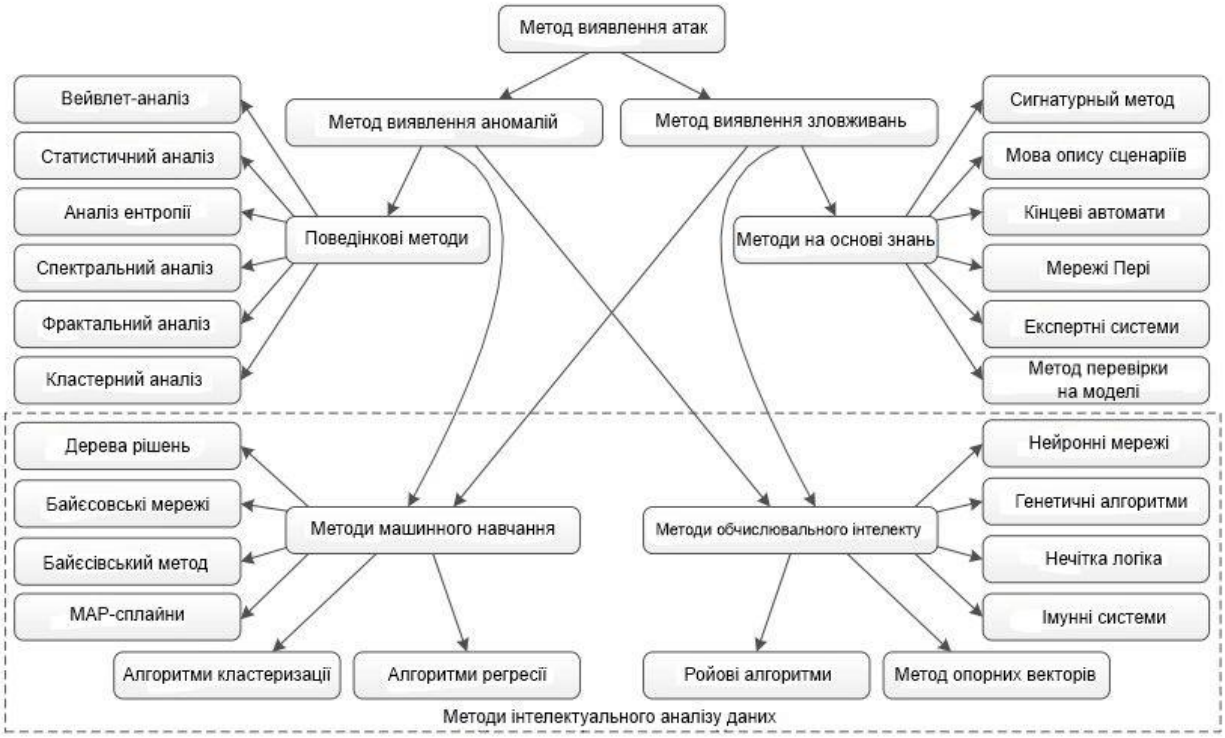



Рисунок 1.13 – Методи виявлення атак

**Вейвлет-аналіз** полягає в побудові коефіцієнтів, що використовуються в розкладанні вихідного сигналу по базисних функціях. Як сигнал може розглядатися інтенсивність мережевого трафіку або дані про кореляцію IP-адрес призначення. Виконання вейвлет-перетворення дозволяє виділити найбільш вагому інформацію як сигнал, що відповідає коливанням з високою амплітудою, і ігнорувати менш корисну інформацію в коливаннях з низькою амплітудою як шумову складову.

**Статистичний аналіз** є ядром методів виявлення аномалій в мережі. До цієї групи відносять такі методи:

- ланцюги Маркова;
- метод хі-квадрат ($\chi 2$);
- метод середньоквадратичних відхилень;
- аналіз розподілів інтенсивності передачі / прийому пакетів;
- аналіз часових рядів;
- пороговий аналіз.

*Важливий* правильний вибір контрольованих параметрів, що характеризують відмінності в нормальному і аномальному трафіку. Може вийти так, що через неправильний вибір кількості спостережуваних параметрів модель опису поведінки суб'єктів в системі виявиться неповною або надлишковою. Це призводить до пропуску атак або помилкових спрацьовувань в системі.

*Перевагами* статистичних систем є їх адаптація до зміни поведінки користувача, а також здатність до виявлення модифікацій атаки. Серед недоліків можна відзначити високу ймовірність виникнення помилкових повідомлень про атаки і залежність від порядку проходження подій.

**Аналіз ентропії** використовується в виявленні атак для формування статистичного критерію з метою перевірки приналежності досліджуваного екземпляра аномальному класу.

**Спектральний аналіз** є окремим випадком вейвлет перетворення і дозволяє виділяти найбільш інформативні складові досліджуваного процесу за допомогою зміни розмірності вихідного простору ознак.

**Фрактальний аналіз** заснований на припущенні, що мережевий трафік задовольняє властивості самоподібності.

Суть **кластерного аналізу** полягає у виділенні таких характеристик з мережевого трафіку, які дозволять розбити об'єкти, що класифікуються, (пакети, з'єднання) на групи, відповідні нормальному функціонуванню мережевого взаємодії. Всі інші примірники, які не потрапляють в побудовані області, класифікуються як аномальні.

**Методи на основі знань.**

До методів на основі знань відносять такі методи, які в контексті заданих фактів, правил виводу і зіставлення, що відображають ознаки заданих атак,

виробляють дії по виявленню атак на основі закладеного механізму пошуку. Як процедури пошуку може застосовуватися зіставлення за зразком, апарат регулярних виразів, логічний секвенційний висновок, аналіз переходу станів і т.д. Своєю назвою ці методи зобов'язані тим, що системи, засновані на їх застосуванні, працюють *з базою знань, в яку включені відомості щодо вже відомих атак*. Тут база знань представлена сховищем, що містить записи експертів з підтримкою логіки їх обробки і інтерпретації (тобто характеризується наявністю підсистеми логічного виведення).

У **сигнатурних методах** системні події подаються у вигляді ланцюжків символів з деякого алфавіту. Суть цих методів полягає в завданні безлічі сигнатур атак у вигляді *регулярних виразів* (regular expressions) або *правил на основі зіставлення зі зразком* (pattern matching) і перевірці відповідності спостережуваних подій цим виразами. Типовими представниками систем, в яких реалізований такий метод, є Snort і Suricata.

**Мови опису сценаріїв** надають гнучкий механізм обробки контрольованих даних, який полягає в можливості написання власних скриптів. Такий підхід дозволяє виявляти такі події, які важкі для опису за допомогою звичайних сигнатурних інструментів аналізу.

**Метод на основі кінцевих автоматів** (КА) дає можливість моделювати атаки у вигляді взаємозалежної мережі з станів і переходів. Вторгнення вважається успішно реалізованим, якщо послідовність дій атакуючого призводить систему з деякого стійкого стану в скомпрометований.

Функціонування **експертних систем** заснована на застосуванні правил виведення до даних про вхідних події. Вони являють собою системи, які приймають рішення про приналежність події до певного класу атак на підставі заданих продукційних правил.

Перевага застосування методу **мереж Пейтри** полягає в можливості візуального уявлення атаки у вигляді діаграм переходу станів, а також у здатності системи виявляти атаку до її фактичного здійснення. З недоліків можна відзначити складність реалізації.

**Метод перевірки на моделі** полягає в специфікуванні властивостей системи на мові алгебри процесів або у вигляді формул темпоральної логіки, побудові моделі у вигляді кінцевого автомата, автоматичної перевірки того, що ця модель задовольняє специфікації, причому при негативному висновку виробляється контрприклад. Метод перевірки моделі полягає в переборі всіх можливих переходів автомата з одного стану в інший з якогось початкового стану системи. Всі можливі траси з початкового набору станів перебираються, щоб переконатися в тому, що всі вони є безпечними, або довести, що живучість системи не може бути порушена. Метод перевірки моделі страждає недоліком, широко відомим під назвою "вибух кількості станів". При моделюванні часу як безперервної суті навіть найпростіша модель має нескінченне число станів.

**Методи обчислювального інтелекту.**

**Штучна нейронна мережа** являє собою набір обробних елементів - нейронів, які пов'язані між собою синапсами і які перетворюють набір вхідних значень в набір бажаних вихідних значень. Нейронні мережі застосовуються в широкому спектрі додатків: розпізнаванні образів, теорії управління, криптографії, стисненні даних. Нейронні мережі мають здатність навчання за зразком і узагальнення із зашумленних і неповних даних. У процесі навчання відбувається настройка коефіцієнтів, асоційованих з синаптичними вагами.

**Генетичні алгоритми** (ГА) засновані на імітації біологічних принципів природного відбору і призначені для вирішення завдань оптимізації. Робота ГА починається зі створення початкової популяції індивідуумів. Кожен представник популяції задається у вигляді набору генів (хромосом), що представляють собою символьні або бінарні послідовності (рядки) і піддаються перетворенням в процесі еволюції. Нове потомство генерується за допомогою операторів схрещування та мутації. Ці оператори виробляють рекомбінацію хромосом і зміну структури нащадків. Для кожної особини нового потомства обчислюється значення пристосованості, яке характеризує показник ефективності вирішення даного завдання.

**Обчислювальні імунні системи** є прототипом імунної системи людини, побудованим на основних принципах її роботи. Серед основних механізмів функціонування імунної системи можна назвати створення і навчання імунних детекторів, знищення детекторів, що викликають помилкові спрацьовування, відповідну реакцію на чужорідні патогени.

**Нечітка логіка –** розділ математики, який є узагальненням класичної логіки і теорії множин й вивчає об'єкти з функцією належності елемента до множини, яка приймає значення у інтервалі [0, 1], а не тільки 0 або 1. На основі цього поняття вводяться логічні операції над нечіткими множинами, і формулюється поняття лінгвістичної змінної, якою виступають нечіткі множини. Предметом нечіткої логіки вважається дослідження суджень в умовах нечіткості, які схожі з судженнями у звичайному сенсі, та їх застосування у обчислювальних системах.

Класифікатор, заснований на **методі опорних векторів** (англ. SVM, support vector machine), застосовується, як правило, для класифікації елементів з двох лінійно розділених множин. Суть алгоритму полягає в побудові оптимальної гіперплощини, яка задається лінійною комбінацією декількох опорних векторів з навчальної вибірки. Залежно від розташування елемента по відношенню до цієї площини приймається рішення про приналежність елемента до того чи іншого класу. У разі лінійної нероздільності цих множин вводиться або умова, що мінімізує помилку розпізнавання (штраф), або застосовуються різні ядра (відображення) для переходу до просторів, що спрямляються (можливо, більшої розмірності, ніж вихідне).

Серед інших методів обчислювального інтелекту варто відзначити **ройові алгоритми**, які моделюють поведінку і взаємодію біологічних особин у відповідних «колоніях» або «зграях».

**Методи машинного навчання**, як і методи обчислювального інтелекту, застосовуються як при виявленні аномалій, так і при виявленні зловживань. Це пояснюється тим, що зазначені підходи в якості вихідних даних для навчання часто використовують шаблони як нормальної, так і аномальної поведінки в мережі.

Деякі автори пропонують замінити стандартний модуль виявлення в системі Snort **деревами рішень**. **Дерево рішень** - графічний метод, що дозволяє пов'язати точки прийняття рішення, можливі стратегії, їх наслідки з можливими факторами, умовами зовнішнього середовища. Побудова дерева рішень починається з більш раннього рішення, потім зображуються можливі дії і наслідки кожного дії (подія), потім знову приймається рішення (вибір напрямку дії) і далі до тих пір, поки всі логічні наслідки результатів не будуть вичерпані.

Дерево рішень будується за допомогою п'яти елементів: 1) момент прийняття рішення; 2) точка виникнення події; 3) зв'язок між рішеннями і подіями; 4) імовірність настання події (сума ймовірностей в кожній точці повинна бути дорівнює 1); 5) очікуване значення (наслідки) - кількісне вираження кожної альтернативи, розташоване в кінці гілки.

Експерименти були проведені на наборі даних DARPA і показали збільшення швидкості обробки pcap-файлів, що використовуються для аналізу мережевих пакетів, в середньому на 40,3% в порівнянні зі стандартним модулем.

Одним з найбільш часто використовуваних підходів для виявлення вторгнень є **байесовські мережі**. Байесовська мережа - це модель, яка кодує імовірнісні відносини між розглянутими подіями (перемінними) і надає певний механізм для обчислення умовних ймовірностей їх настання. Окремий випадок цієї моделі - наївний байесовский класифікатор (**байесовский метод**) зі строгими припущеннями про незалежність вхідних змінних. Класифікатор дозволяє оцінити апостеріорну ймовірність приналежності примірника заданому класу на основі безумовної теореми Байеса.

Метод на основі **MAP-сплайнів** дозволяє побудувати досить точну апроксимацію поведінки звичайного користувача або зловмисника по заданих параметрах. З цією метою вибирається набір базисних функцій, і знаходяться коефіцієнти в лінійному розкладі по заданому базису і навчальним векторам.

**Алгоритми кластерізації** використовуються в кластерному аналізі. Кластерний аналіз – задача розбиття заданої вибірки об'єктів (ситуацій) на підмножини, які називаються кластерами, так, щоб кожен кластер складався з схожих об'єктів, а об'єкти різних кластерів істотно відрізнялися. Завдання кластеризації відноситься до статистичної обробки, а також до широкого класу завдань навчання без вчителя.

**Алгоритми регресії.** Регресійний аналіз — розділ математичної статистики, присвячений методам аналізу залежності однієї величини від іншої. На відміну від кореляційного аналізу не з'ясовує чи істотний зв'язок, а займається пошуком моделі цього зв'язку, вираженої у функції регресії. Регресійний аналіз використовується в тому випадку, якщо відношення між змінними можуть бути виражені кількісно у виді деякої комбінації цих змінних. Отримана комбінація використовується для передбачення значення, що може приймати цільова (залежна) змінна, яка обчислюється на заданому наборі значень вхідних (незалежних) змінних. У найпростішому випадку для цього використовуються стандартні статистичні методи, такі як лінійна регресія.

## 2. АНАЛІЗ ТА МОНІТОРИНГ МЕРЕЖ

### 2.1 Сканери безпеки комп’ютерних мереж та мережеві сканери.

Сканери безпеки - програмні або апаратні засоби, що служать для здійснення діагностики і моніторингу мережі та комп'ютерів, що дозволяє сканувати мережі, комп'ютери та програми на предмет виявлення можливих проблем у системі безпеки, оцінювати і усувати уразливості. Сканери вразливостей дозволяють перевірити різні додатки в системі на предмет наявності «дірок» , якими можуть скористатися зловмисники. Також можуть бути використані низькорівневі засоби, такі як сканер портів, для виявлення й аналізу можливих додатків і протоколів, що виконуються в системі. В загальному випадку мета сканування - виявити можливі загрози.

Роботу сканера безпеки можна розбити на 4 етапи:

- Зазвичай, сканер спочатку виявляє активні IP-адреси, відкриті порти, запущену операційну систему і додатки.
- Складається звіт про безпеку (необов'язковий крок).
- Спроба визначити рівень можливого втручання в операційну систему або додатки (може спричинити збій).
- На заключному етапі сканер може скористатися вразливістю, викликавши збій операційної системи або програми.

Сканери можуть бути шкідливими або «дружніми». Останні зазвичай зупиняються в своїх діях на кроці 2 або 3, але ніколи не доходять до кроку 4.

Серед сканерів безпеки можна виділити:

- Сканер портів.
- Сканери, що досліджують топологію комп'ютерної мережі.
- Сканери, що досліджують уразливості мережевих сервісів – сканери вразливостей.
- CGI-сканери ( «дружні» - допомагають знайти вразливі скрипти).

Далі розглянемо лише сканери мереж та методи які вони використовують. Основне призначення мережевих сканерів це виявлення працюючих хостів, побудова топології мережі, виявлення відкритих портів та служб що на них працюють, а також операційних систем.

### 2.2 Методи та методики мережевого сканування

**Відкриті методи сканування**

***TCP SYN scanning за допомогою функції connect ().***

Засновані на створенні TCP-з'єднання. По черзі на різні порти передаються на сервер TCP SYN-запити для установки з'єднання. Порт, вказується як параметр функції. Якщо він відкритий, то буде отримана відповідь TCP SYN ACK:

client → SYN

server → SYN/ACK

client → ACK
Якщо з'єднання не встановлено, то порт із зазначеним номером є закритим:
client → SYN
server → RST/ACK
client → RST

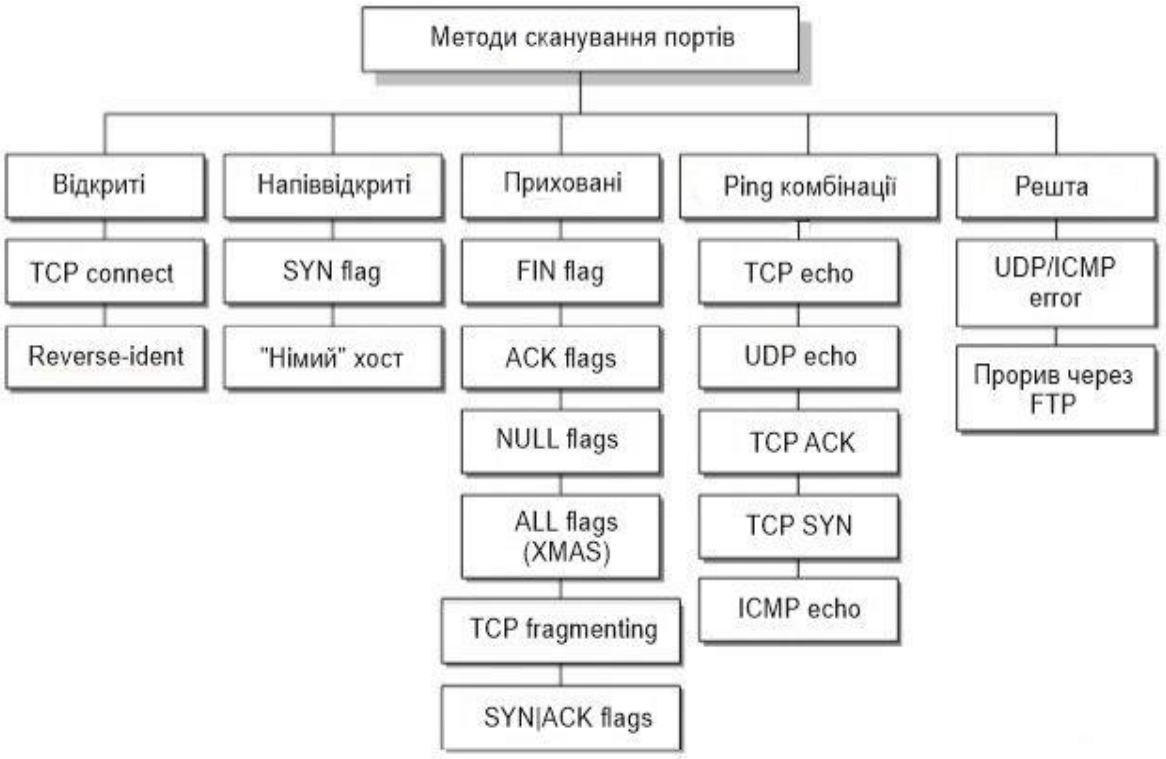



Рисунок 2.1 – Методи сканування портів

***Сканування TCP-портів методом reverse-ident (зворотної ідентифікації)***

Протокол ident дозволяє визначити ім'я (username чи login, вказане при вході в систему) власника будь-якого запущеного на сервері процесу, пов'язаного з ним, навіть якщо сам цей процес не ініціалізувати.

Процес

- Сервер прослуховує 113 порт і чекає приходу запиту на з'єднання.
- Сервер зчитує блок даних, що характеризує з'єднання, для якого необхідно отримати інформацію аутентифікації.
- <Порт сервера "," Порт клієнта> Наприклад: 6191, 23
- Відповідь сервера виглядає наступним чином: <Порт сервера "," Порт клієнта>: <Відповідь>: <Дод.інф.>

наприклад:

6193, 23: USERID: UNIX: stjohns 6195, 23: ERROR: NO-USE

**Переваги і недоліки**

- Переваги: його може застосувати будь-який користувач, що не володіє ніякими привілеями на хості; швидкість дослідження.
- Недоліки: легко виявляється і протоколюється

**Напіввідкриті методи сканування**

Під «напіввідкритими» розуміються методи сканування, які розривають з'єднання до закінчення тристороннього рукостискання (процедури повного встановлення з'єднання)

**SYN scan (TCP-сканування пакетами з встановленим прапором SYN)**

Встановлення повного TCP-з'єднання не проводиться. Хост відправляє на певний порт сервера SYN-пакет, як би маючи намір створити з'єднання, і очікує відповідь

client -> SYN
server -> SYN|ACK
client -> RST
---
client -> SYN
server -> RST|ACK

**Переваги і недоліки**

- Переваги: - деякі сервери здатні зареєструвати такого роду сканування
- Недоліки: - необхідні права адміністратора (root)

**Приховані методи сканування**

- Використовують установку якогось одного прапора (ACK, FIN, RST, ..) Використовують скидання всіх прапорів (Null scan).
- Використовують установку всіх прапорів.
- Обминають фільтри пакетів, фаєрволи, маршрутизатори.
- Не відрізнятимуться від звичайних трафіку.
- Використовують різні способи поділу пакетів.
- Використовують технологію «інвертованого сканування», тобто підтвердження приходить для закритих портів.

***Основні методи прихованого сканування***

- SYN | ACK scan (TCP-сканування пакетами з встановленими прапорами SYN і ACK)
- Fin scan (TCP-сканування пакетами з встановленим прапором FIN)
- Ack scan (TCP-сканування пакетами з встановленим прапором ACK)
- Null scan (TCP-сканування нульовими пакетами)
- Xmas Tree scan (TCP-сканування методом "різдвяної ялинки", коли всі флаги встановлені в одиницю)
- TCP Fragmenting (TCP-сканування з використанням IP-фрагментації)

**ACK-сканування**

Цей удосконалений метод зазвичай використовується для розкриття набору правил міжмережного екрана. Зокрема, він може допомогти визначити, чи є міжмережний екран брандмауером типу “stateful firewall”, чи простим пакетним фільтром, який блокує SYN-пакети, що надходять.

Цей тип сканування відсилає ACK-пакет (з номерами підтвердження/послідовності, що виглядають випадковими) на вказані порти. Якщо повертається RST, порти класифікуються як закриті безпосередньо на хості («нефільтровані»). Якщо нічого не повертається (або якщо повертається ICMP-повідомлення «недосяжний»), порт класифікується як «фільтрований», тобто закритий фільтром або міжмережним екраном. Це сканування очевидно ніколи не буде показувати відкриті порти.

**Сканування за допомогою ping - комбінації**

***Визначення стану сервера методом ICMP-сканування***

- Хост пересилає сервера ICMP-повідомлення і чекає отримання відповіді - ICMP-повідомлення.
- Якщо прийом відповіді успішний, значить маршрутизація виконана, сервер працездатний і швидше за все чекає запит на з'єднання.
- Щоб здійснити запиту застосовується програма ping.

2.3 Сканер nmap.

Nmap ("Network Mapper") – це утиліта з відкритим вихідним кодом для дослідження мережі та перевірки безпеки мереж та виявлення активних мережевих сервісів.

Він був розроблений для швидкого сканування великих мереж, але відмінно працює на окремих хостах. Nmap працює на всіх основних комп'ютерних операційних системах, а офіційні виконавчі пакети доступні для Linux, Windows і Mac OS X. На додаток до класичного виконуваного файлу Nmap з командного рядка, є утиліта з графічним інтерфейсом (Zenmap), гнучкий інструмент для передачі, перенаправлення і налагодження даних (Ncat), утиліта для порівняння результатів сканування (Ndiff) і інструмент генерації пакетів і аналізу відповідей (Nping).

Nmap використовує велику кількість різноманітних методів сканування, таких як UDP, TCP (connect), TCP SYN, FTP-proxy, Reverse-ident, ICMP (ping), FIN, ACK, Xmas tree, SYN- и NULL-сканування. Nmap також підтримує великий набір додаткових можливостей, а саме: визначення операційної системи віддаленого хоста, невидиме сканування, динамічне обчислення часу затримки та повтор передачі пакетів, паралельне сканування, визначення неактивних хостів методом паралельного ping-опитування, сканування з використанням підставних хостів, визначення наявності пакетних фільтрів, пряме (без використання portmapper) RPC-сканування, сканування з використанням IP-

фрагментації, швидкий пошук вразливостей SQL Injection, а також довільне вказування IP-адрес та номерів портів мереж, що скануються.

Коли Nmap запускається, без будь-яких опцій, на екрані можна побачити таку інструкцію:

Використання: nmap [тип сканування) [опції] {ціль сканування}

ВИЗНАЧЕННЯ ЦІЛІ СКАНУВАННЯ:

Можна використовувати мережеві імена, IP адреси, мережі та ін.

Приклад: scanme.Nmap.org, microsoft.com/24, 192.168.0.1; 10.0.0-255.1-254

-iL <ім'я_вхідного_файла>: Використовувати список хостів/мереж із файла;

-iR <кількість_хістів>: Обрати довільні цілі

ВИЯВЛЕННЯ ХОСТІВ:

-sL: Сканування з метою складання списку - просто скласти список цілей для сканування

-sP: Пінг сканування - просто визначити, чи працює хост

-PN: Розцінювати все хости як працюючі - пропустити виявлення хостів

-PS/PA/PU [список_портів]: TCP SYN/ACK чи UDP пінгування заданих хостів

-PE/PP/PM: Пінгування з використанням ICMP exo запитів, запитів тимчасової мітки і мережевої маски

-PO [список_протоколів]: Пінгування з використанням IP протоколу

-n/-R: Никогда не производить DNS разрешение/Всегда производить разрешение

РІЗНІ ПРИЙОМИ СКАНУВАННЯ:

-sS/sT/sA/sW/sM: SYN/з використанням системного виклику Connect()/ACK/ Window/Maimon сканування

-sU: UDP сканування

-sN/sF/sX: TCP Null, FIN и Xmas сканування

--scanflags <прапорці>: Задати власні TCP флаги

-sI <зомбі_хост[:порт]>: "Ліниве" (Idle) сканування

-sO: Сканування IP протоколу

-b <FTP_хост>: FTP bounce сканування

ВИЗНАЧЕННЯ ПОРТІВ І ПОРЯДКУ СКАНУВАННЯ:

-p <діапазон_портів>: Сканування тільки певних портів

Приклад: -p22; -p1-65535; -p U:53,111,137,T:21-25,80,139,8080

-F: Швидке сканування - Сканування обмеженої кількості портів

-r: Сканувати порти послідовно - не використовувати випадковий порядок портів

ВИЗНАЧЕННЯ ОС:

-O: Активувати функцію визначення ОС

2.4 Мережеві аналізатори. Принципи функціонування мережевих аналізаторів.

**Аналізатор траафіку**, або **сні́фер** (від англ. to sniff — нюхати) — програма або програмно-апаратний пристрій, призначений для перехоплення і подальшого аналізу, або тільки аналізу мережного трафіку, призначеного для інших вузлів.

Перехоплення трафіку може здійснюватися:

- звичайним «прослуховуванням» мережевого інтерфейсу (метод ефективний при використанні в сегменті концентраторів (хабів) замість комутаторів (світчей), інакше метод малоефективний, оскільки на сніфер потрапляють лише окремі фрейми);
- підключенням сніфера в розрив каналу;
- відгалуженням (програмним або апаратним) трафіку і спрямуванням його копії на сніфер;
- через аналіз побічних електромагнітних випромінювань і відновлення трафіку, що таким чином прослуховується;
- через атаку на канальному (MAC-spoofing) або мережевому рівні (IP-spoofing), що приводить до перенаправлення трафіку жертви або всього трафіку сегменту на сніфер з подальшим поверненням трафіку в належну адресу.

Аналіз трафіку, що пройшов через сніфер, дозволяє:

- Виявити паразитний, вірусний і закільцований трафік, наявність якого збільшує завантаження мережного устаткування і каналів зв'язку (сніфери тут малоефективні; як правило, для цих цілей використовують збір різноманітної статистики серверами і активним мережним устаткуванням і її подальший аналіз).
- Виявити в мережі шкідливе і несанкціоноване ПЗ, наприклад, мережеві сканери, флудери, троянські програми, клієнти пірінгових мереж та інші (це зазвичай роблять за допомогою спеціалізованих сніферів — моніторів мережної активності).
- Перехопити будь-який незашифрований (а деколи і зашифрований) призначений для користувача трафік з метою отримання паролів і іншої інформації.
- Локалізувати несправність мережі або помилку конфігурації мережних агентів (для цієї мети сніфери часто застосовуються системними адміністраторами).

Оскільки в «класичному» сніфері аналіз трафіку відбувається вручну, із застосуванням лише простих засобів автоматизації (аналіз протоколів, відновлення TCP-потоку), то він підходить для аналізу лише невеликих його обсягів. Принцип роботи мережевого аналізатора (див. рис. 2.2)

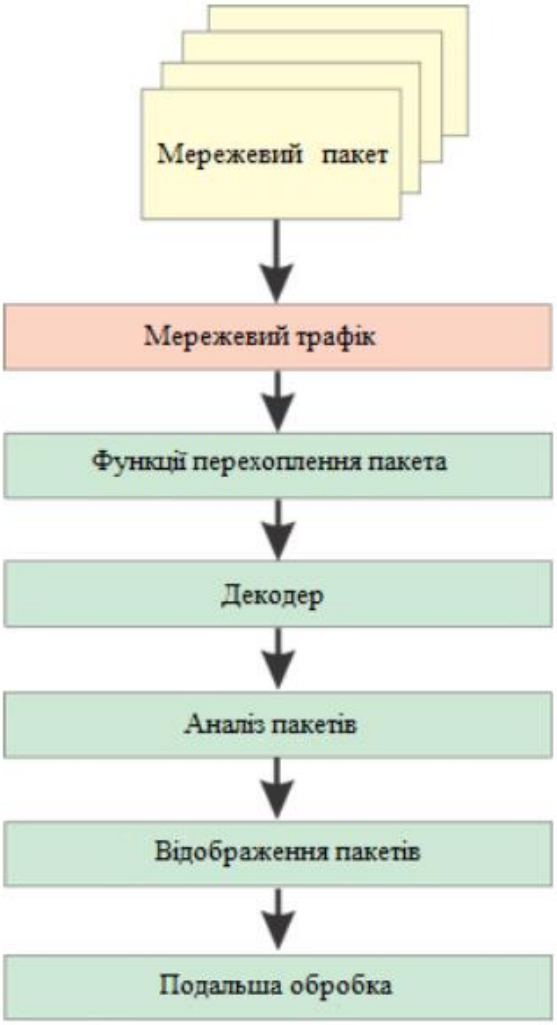



Рисунок 2.2 – Принцип роботи мережевого аналізатора

На рис.2.3 схематично показано практичне застосування мережевого аналізатору Wireshark. Схема відображає програмно-апаратний пристрій Wireshark, призначений для перехоплення і подальшого аналізу, або тільки аналізу мережного трафіку на даному вузлі. На схемі також демонстративно зображено застосування IDS/IPS та Firewall. Суть схеми полягає в тому що трафік який входить та виходить від групи серверів та групи комп'ютерів аналізується та фільтрується, а також інформація що знаходиться на серверах не доступна для групи комп'ютерів через встановлений Firewall.

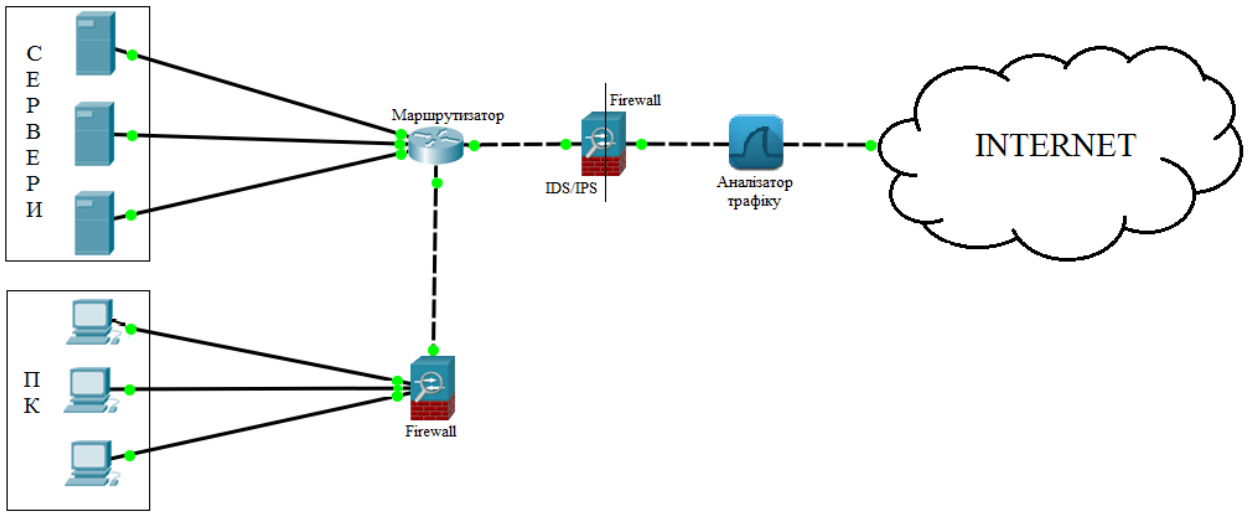



Рисунок 2.3 – Схема застосування мережевого аналізатору

2.5 Мережеві аналізатори tcpdump та wireshark.

**Tcpdump** (від TCP і англ. dump — звалище, скидати) — сніфер, утиліта UNIX, що дозволяє захоплювати і аналізувати мережний трафік, що проходить через комп'ютер, на якому запущена ця програма.

Основні призначення tcpdump:

- Налагодження мережевих програм
- Налагодження мережі і мережної конфігурації в цілому
- Програмна реалізація

Програма складається з двох основних частин: частини **захоплення пакетів** (звернення до бібліотеки, libcap (Linux) або pcap (Windows)) і частини **відображення захоплених пакетів** (яка на рівні вихідного коду є модульною і для підтримки нового протоколу досить додати новий модуль).

**Частина захоплення пакетів** (при запуску) передає «вираз вибору пакетів» (що йде після всіх параметрів командного рядка) безпосередньо бібліотеці захоплення пакетів, яка перевіряє правильність синтаксису виразу, компілює його (у внутрішній формат даних), а потім копіює у внутрішній буфер програми мережні пакети, які проходять через вибраний мережевий інтерфейс і задовольняють умовам у виразі.

**Частина відображення пакетів** по черзі вибирає захоплені пакети з внутрішнього буфера, і виводить їх на стандартний вивід в форматі зрозумілому людині, згідно із заданим рівнем детальності.

Якщо задано докладний вивід пакетів, програма перевіряє для кожного мережевого пакету, чи є у неї модуль розшифрування даних, і, у разі наявності, відповідною підпрограмою витягує (і відображає) тип та інші параметри пакету в протоколі.

**Синтаксис**:

Tcpdump [-aAdDeflLnNOpqRStuUvxX] [-c count] [-C file_size] [-E algo:secret] [-F file] [-i interface] [-M secret] [-r file] [-s snaplen] [-T type] [-w file] [-W filecount] [-y datalinktype] [-Z user] [expression]

**Найбільш популярні ключі:**

- **c** count вийти після отримання певної кількості пакетів.
- **F** File використовувати file для введення фільтруючого вираження. Вираз, вказаний в командному рядку, буде ігноруватися.
- **i** interface збирати пакети тільки на певному інтерфейсі. Якщо не вказано - береться мінімальний за номером інтерфейс (виключаючи loopback). Для Linux-ядер 2.2 і новіших, можливо вказати 'any', тоді буде відбуватися збір на всіх інтерфейсах, але вони не будуть переведені в режим promiscuous.
- **n** не перетворювати адресу хосту в ім'я. Може бути використано, якщо необхідно уникати DNS-запитів.
- **nn** не перетворювати протокол і номер порту в їх імена.
- **N** не виводити доменну частину імені хоста. Наприклад, при цьому ключі буде виводиться "nic" замість "nic.ddn.mil"
- **p** не переводити інтерфейс в режим promiscuous. Слід зауважити, що інтерфейс може бути в режимі promiscuous з інших причин.
- **r** file читати пакети з file (який, був створений з ключем -w). Якщо file вказаний як "-", то використовується стандартне введення.
- **t** не виводити тимчасової штамп (timestamp) в кожному рядку дампа (dump).
- **tt** висновок не форматований тимчасової штамп в кожному рядку дампа.
- **ttt** виводити різницю (в мікросекундах) між поточною і попередньої рядками дампа.
- **tttt** виводити тимчасової штамп разом з датою в форматі за замовчуванням в кожному рядку дампа.
- **v** докладний висновок. Для ще більш докладного виведення використовуються: -vv і -vvv.
- w file писати "сирі" пакети в file перед тим як зробити їх розбір і вивести. Вони можуть бути пізніше виведені з ключем -r. Якщо file вказаний як "-", то використовується стандартний вивід.
- x друкувати кожен пакет (без заголовків рівня з'єднання) в шістнадцятирічному вигляді.
- X крім шістнадцятирічному вигляду виводити їх ASCII-значення.

**Wireshark** (раніше звався Ethereal) – програма для аналізу мережевих пакетів Ethernet і інших мереж (сніфер) з вільним вихідним кодом. Має графічний інтерфейс користувача. У червні 2006 року проект був перейменований на Wireshark через проблеми з торговою маркою.

Програмна реалізація.

Функціональність, яку надає Wireshark, дуже схожа з можливостями програми tcpdump, проте Wireshark має графічний інтерфейс користувача і значно більше можливостей із сортування і фільтрації інформації. Програма дозволяє користувачеві переглядати весь трафік, що проходить по мережі, в режимі реального часу, переводячи мережну карту в promiscuous mode режим.

Wireshark – це програма, яка розпізнає структуру найрізноманітніших мережевих протоколів, і тому дозволяє розібрати мережевий пакет, відображаючи значення кожного поля протоколу будь-якого рівня. Оскільки для захоплення пакетів використовується pcap, існує можливість захоплення даних тільки з тих мереж, які підтримуються цією бібліотекою. Проте, Wireshark вміє працювати з безліччю форматів початкових даних, відповідно, можна відкривати файли даних, захоплених іншими програмами, що розширює можливості захоплення.

Програма розповсюджується під вільною ліцензією GNU GPL і використовує для формування графічного інтерфейсу кросплатформову бібліотеку GTK+. Існують версії для більшості типів UNIX, зокрема GNU/Linux, Solaris, FreeBSD, NetBSD, OpenBSD, Mac OS X, а також для Microsoft Windows.

**Компоненти Wireshark.**

На рис. 2.4 представлена архітектура інструменту Wireshark, яка складається з двох компонентів:

1. Бібліотека, за допомогою якої здійснюється перехоплення мережевого трафіку (WinPcap для ОС Windows, libpcap для ОС Linux). Бібліотека Pcap (Packet Capture) дозволяє створювати програми аналізу мережевих даних, що надходять на мережеву карту комп'ютера. Вона призначена для використання спільно з мовами C / C ++, а для роботи з бібліотекою на інших мовах, таких як Java, .NET, використовується shell. Програмне забезпечення мережевого моніторингу може використовувати libpcap або WinPcap, щоб захопити пакети, які передаються по мережі, і (в новіших версіях) для передачі пакетів в мережі. Libpcap і WinPcap також підтримують збереження захоплених пакетів в файл і читання файлів, що містять збережені пакети. Програми, написані на основі libpcap або WinPcap, можуть захопити мережевий трафік, аналізувати його. Файл захопленого трафіку зберігається в форматі, зрозумілому для додатків, що використовують Pcap.

2. Прикладна програма, що надає функції розбору протоколів і графічний інтерфейс для візуалізації результатів аналізу і взаємодії з системою.

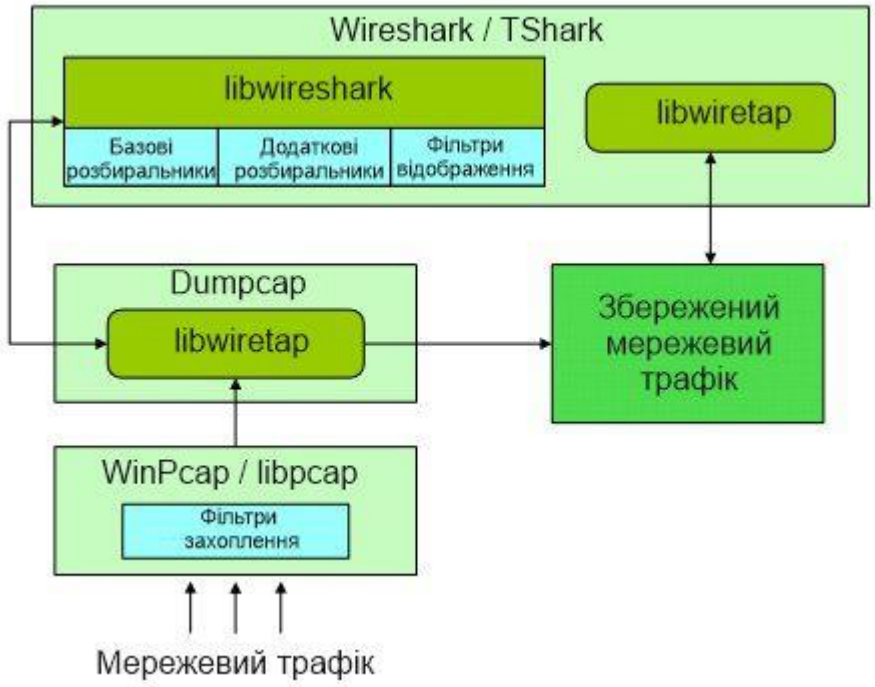



Рисунок 2.4 – Архітектура Wireshark

**Фільтри**

Wireshark містить **два види фільтрів - захоплення** (Capture Filters) і **відображення** (Display Filters).

**Capture Filters**. Вони служать для фільтрації ще на етапі захоплення трафіку. Але в такому випадку, безумовно, можна безповоротно втратити частину потрібного трафіку. Фільтр являє собою вираз, що складається з вбудованих значень, які при необхідності можуть об'єднуватися логічними функціями (and, or, not). Для того, щоб його задіяти, потрібно зайти в меню Capture, потім Options, і в поле Capture Filter набрати, наприклад, host 8.8.8.8 (або, наприклад, net 192.168.0.0./24). Так само, звичайно, можна вибрати і заздалегідь створений фільтр (за це відповідає кнопка Capture Filter). У будь-якому з варіантів фільтр з'явиться біля інтерфейсу, можна жати Start.

**Display Filters**. Вони фільтрують виключно уже захоплений трафік за такими параметрами як протоколи, адреси, специфічні поля в протоколах.

Особливу увагу необхідно приділити функції розбору, яка надає три сценарії взаємодії:

- прямий виклик (безпосередній виклик розбирача);
- зворотній виклик (викликаний розбірник визначає значення деякого ключового поля, що отримане при розборі більш низькорівневого протоколу; наприклад, значення поля «порт» в заголовку TCP пакету);
- евристична прив'язка (викликаний разбірник визначається шляхом пошуку патернів в уже згадуваному буфері);

Wireshark надає можливість для підключення додаткових (в тому числі самостійно розроблених) модулів розбору трафіку.

Найбільш поширеними фільтрами є:

- ip.dst - цільової IP-адреса;
- ip.src - IP-адреса відправника;
- ip.addr - IP відправника або одержувача;
- ip.proto - протокол;
- tcp.dstport - порт призначення;
- tcp.srcport - порт відправника;
- ip.ttl - фільтр по ttl, визначає мережеве відстань;
- http.request_uri - запитуваний адресу сайту.

Для вказівки відносини між полем і значенням в фільтрі можна використовувати такі оператори:

- == - рівно;
- ! = - не дорівнює;
- <- менше;
- - більше;
- <= - менше або дорівнює;
- = - більше або дорівнює;
- matches - регулярний вираз;
- contains - містить.

Для об'єднання декількох виразів можна застосовувати:

- && - обидва вирази повинні бути вірними для пакета;
- || - може бути вірним один з виразів.

**Основні переваги та недоліки**

Основні **переваги** інструменту:

- Підтримка великої кількості мережевих протоколів (в тому числі протоколів IP-телефонії);
- Підтримка різних форматів мережевих трас;
- Можливість розширення (можливість створення і підключення додаткових модулів розбору);
- Детальна система фільтрації мережевих пакетів;
- Можливість відновлення потоків TCP;

**Недоліки**:

- Відновлений потік не розглядається інструментом як єдиний буфер пам'яті, внаслідок чого його подальша обробка неможлива;
- Код модулів розбору містить функції, що відповідають за візуалізацію результатів (логіка розбору перемішується з логікою відображення в графічному інтерфейсі);
- Відсутня можливість виконання деякого дії в разі виявлення сигнатур в

трафіку.

2.6 Система моніторингу мережі Zabbix.

Система моніторингу мережі - називають систему, яка виконує постійний нагляд за комп'ютерною мережею в пошуках повільних або несправних систем і яка при виявленні збоїв повідомляє про них адміністратора за допомогою засобів оповіщення. Ці завдання є підмножиною завдань управління мережею. Наприклад, для того, щоб визначити стан веб-сервера, програма, що виконує моніторинг, може періодично відправляти запит HTTP на отримання сторінки; для поштових серверів можна відправити тестове повідомлення по SMTP і отримати по IMAP або POP3. Невдалі запити (наприклад, в тому випадку, коли з'єднання не може бути встановлено, воно завершується по тайм-ауту, або коли повідомлення не було доставлено) зазвичай викликають реакцію з боку системи моніторингу. Як реакція може бути:

- відправлений сигнал тривоги системного адміністратора.
- активована система захисту від збоїв, яка тимчасово виведе проблемний сервер з експлуатації, до тих пір, поки проблема не буде вирішена.

Постійний контроль за роботою локальної мережі, що становить основу будь-якої корпоративної мережі, необхідний для підтримки її в працездатному стані. Контроль - це необхідний перший етап, який повинен виконуватися при управлінні мережею. Зважаючи на важливість цієї функції її часто відокремлюють від інших функцій систем управління і реалізують спеціальними засобами. Такий поділ функцій контролю і власне управління корисно для невеликих і середніх мереж, для яких установка інтегрованої системи управління економічно недоцільна. Використання автономних засобів контролю допомагає адміністратору мережі виявити проблемні ділянки і влаштування мережі, а їх відключення або реконфігурацію він може виконувати в цьому випадку вручну.

Процес контролю роботи мережі зазвичай ділять на два етапи:

- моніторинг.
- аналіз.

На етапі моніторингу виконується більш проста процедура - процедура збору первинних даних про роботу мережі: статистики про кількість циркулюючих в мережі кадрів і пакетів різних протоколів, стан портів концентраторів, комутаторів і маршрутизаторів та іншого проміжного обладнання.

Далі виконується етап аналізу, під яким розуміється більш складний і інтелектуальний процес осмислення зібраної на етапі моніторингу інформації, зіставлення її з даними, отриманими раніше, і вироблення припущень про можливі причини сповільненої або ненадійної роботи мережі.

Завдання моніторингу вирішуються програмними і апаратними вимірниками, тестерами, мережевими аналізаторами, вбудованими засобами

моніторингу комунікаційних пристроїв, а також агентами систем управління. Завдання аналізу вимагає більш активної участі людини і використання таких складних засобів, як експертні системи, що акумулюють практичний досвід багатьох мережевих фахівців.

Всі засоби моніторингу та аналізу мереж, можна розділити на кілька великих класів:

- Системи управління мережею (Network Management Systems) - централізовані програмні системи, які збирають дані про стан вузлів і комунікаційних пристроїв мережі, а також дані про трафік в мережі. Ці системи не тільки здійснюють моніторинг і аналіз, а й виконують в автоматичному чи напівавтоматичному режимі управління мережею - включення і відключення портів пристроїв, зміна параметрів мостів адресних таблиць мостів, комутаторів і маршрутизаторів і т.п. Прикладами систем управління можуть служити популярні системи HPOpenView, SunNetManager, IBMNetView.
- Засоби управління системою (System Management). Засоби управління системою часто виконують функції, аналогічні функціям систем управління, але стосовно інших об'єктів. У першому випадку об'єктом управління є програмне і апаратне забезпечення комп'ютерів мережі, а у другому - комунікаційне устаткування. Разом з тим, деякі функції цих двох видів систем управління можуть дублюватися, наприклад, засоби управління системою можуть виконувати найпростіший аналіз мережевого трафіку. До найбільш відомих систем управління системами відносяться LANDesk, IBM Tivoli, Microsoft Systems Management Server, HP OpenView, Novell ZENworks і CA Unicenter.
- Вбудовані системи діагностики і управління (Embedded Systems). Ці системи виконуються у вигляді програмно-апаратних модулів, які встановлюються в комунікаційне обладнання, а також у вигляді програмних модулів, вбудованих в операційні системи. Вони виконують функції діагностики і управління тільки одним пристроєм, і в цьому їх основна відмінність від централізованих систем управління. Прикладом засобів цього класу може служити модуль управління концентратором Distrebuted 5000, реалізує функції автосигментації портів при виявленні несправностей, приписування портів внутрішнім сегментам концентратора і деякі інші. Як правило, вбудовані модулі управління також виконують роль SNMP-агентів, які поставляють дані про стан пристрою системам управління.
- Аналізатори протоколів (Protocol analyzers). Представляють собою програмні або апаратно-програмні системи, які обмежуються на відміну від систем управління лише функціями моніторингу і аналізу трафіку в мережах. Хороший аналізатор протоколів може захоплювати і декодувати пакети великої кількості протоколів, що застосовуються в мережах - зазвичай кілька десятків. Аналізатори протоколів дозволяють встановити деякі логічні умови для захоплення окремих пакетів і виконують повне декодування захоплених пакетів,

тобто показувати в зручній для користувача формі вкладеність пакетів протоколів різних рівнів один в одного з розшифруванням змісту окремих полів кожного пакета.

- Обладнання для діагностики і сертифікації кабельних систем. Умовно це устаткування можна поділити на чотири основні групи: мережні монітори, прилади для сертифікації кабельних систем, кабельні сканери і тестери (мультиметри). Мережеві монітори (називають також мережевими аналізаторами) призначені для тестування кабелів різних категорій. Слід розрізняти мережеві монітори і аналізатори протоколів. Мережеві монітори збирають дані лише про статистичні показники трафіку - середньої інтенсивності загального трафіку мережі, середньої інтенсивності потоку пакетів з певним типом помилки і т.п. Призначення пристроїв для сертифікації кабельних систем, безпосередньо випливає з їх назви. Сертифікація виконується відповідно до вимог одного з міжнародних стандартів на кабельні системи. Кабельні сканери використовуються для діагностики мідних кабельних систем. Тестери призначені для перевірки кабелів на відсутність фізичного розриву.
- Експертні системи. Цей вид систем акумулює людські знання про виявлення причин аномальної роботи мереж і можливі способи приведення мережі у працездатний стан. Експертні системи часто реалізуються у вигляді окремих підсистем різних засобів моніторингу та аналізу мереж: систем управління мережами, аналізаторів протоколів, мережевих аналізаторів. Найпростішим варіантом експертної системи є контексно залежна help-система. Більш складні експертні системи являють собою так звані бази знань, що володіють елементами штучного інтелекту. Прикладом такої системи є експертна система, вбудована в систему управління Spectrum компанії Cabletron.
- Багатофункціональні пристрої аналізу та діагностики. У зв'язку з розповсюдженням локальних мереж виникла необхідність розробки недорогих портативних приладів, які суміщають функції декількох пристроїв: аналізаторів протоколів, кабельних сканерів і, навіть, деяких можливостей ПЗ мережного управління. Як приклад такого роду пристроїв можна привести Compas компанії MicrotestInc. або 675 LANMeterкомпанii FlukeCorp.

Розглянемо одне рішення із засобів управління системою, а саме Zabbix

**Zabbix** - це рішення розподіленого моніторингу корпоративного класу з відкритими початковими кодами; програмне забезпечення орієнтоване на моніторинг численних параметрів мережі, життєздатності та цілісності серверів. Zabbix використовує гнучкий механізм сповіщень, що дозволяє користувачам конфігурувати e-mail сповіщення практично для будь-якої події. Така можливість дозволяє швидко реагувати на проблеми з серверами. Zabbix пропонує відмінні функції звітності та візуалізації даних засновані на даних історії.

Zabbix - це високо інтегрований рішення для моніторингу мережі, що пропонує безліч функцій в одному пакеті:

Збір даних:

- перевірка доступності та продуктивності
- підтримка SNMP (відстеження та опитування), IPMI, JMX, моніторингу VMware
- митні перевірки
- збір бажаних даних через певні проміжки часу
- виконується сервером або проксі і агентами

Гнучкі визначення порогів:

- Ви можете визначити дуже гнучкі пороги проблеми, звані тригерами, що посилаються на значення з бази даних бекенда
- Налаштовуються оповіщення:
- відправка повідомлень може бути налаштована відповідно до розкладу ескалації, одержувачем, типом носія
- повідомлення можуть бути стислими та інформативними
- автоматичні дії включають віддалені команди

Графіки у реальному часі:

- відслідковують елементи негайно відображаються за допомогою вбудованої функції побудови графіків

Можливості веб-моніторингу:

- Zabbix може слідувати шляху імітації миші на веб-сайті і перевіряти функціональність і час відгуку.

Розширені можливості візуалізації:

- можливість створювати власні графіки, які можуть об'єднувати кілька елементів в одній виставі
- мережеві карти
- призначені для користувача екрани і слайд-шоу для огляду в стилі приладової панелі
- звіти
- високий рівень (бізнес) вид контрольованих ресурсів

Збереження історії подій:

- дані зберігаються в базі подій
- налаштовується історія
- вбудоване знищення подій

Просте налаштування

- додати відслідковують пристрою в якості хостів
- хости підібрані для моніторингу, один раз в базі
- застосовувати шаблони для відслідковується пристроїв

Використання шаблонів:

- угруповання перевірок в шаблонах
- шаблони можуть наслідувати інші шаблони

- виявлення мережі
- автоматичне виявлення мережевих пристроїв
- автоматична реєстрація агента
- виявлення файлових систем, мережевих інтерфейсів і ідентифікаторів SNMP

Швидкий веб-інтерфейс:

- веб-інтерфейс в PHP
- доступні з будь-якого місця
- Ви можете пройти через журнал аудиту

Zabbix API:

- Zabbix API надає програмований інтерфейс до Zabbix для масових маніпуляцій, інтеграції програмного забезпечення сторонніх виробників і інших цілей.

Система дозволів:

- безпечна аутентифікація користувача
- певні користувачі можуть бути обмежені певними уявленнями

Повнофункціональний і легко розширюваний агент:

- розгорнуть для певних цілей моніторингу
- може бути розгорнутий як на Linux, так і на Windows

Двійкові демони:

- написано на C, для продуктивності і невеликого обсягу пам'яті
- легко переноситься

Готовий для використання у складному середовищі:

- віддалений моніторинг став простіше завдяки використанню Zabbix проксі

Zabbix підтримує як опитування, так і захоплення. Усі звіти та статистичні дані, а також параметри конфігурації доступні через веб інтерфейс. Веб інтерфейс гарантує, що стан мережі та стан серверів можна оцінити з будь-якого місця. Правильно налаштований Zabbix може зіграти важливу роль у моніторингу IT-інфраструктури. Це однаково справедливо як для невеликих організацій з кількома серверами, так і для великих компаній з безліччю серверів. Zabbix безкоштовний. Комерційна підтримка надається та надається компанією Zabbix. Загальна публічна ліцензія GPL. Це означає, що його вихідний код вільно розповсюджується та доступний для широкої громадськості. Багато організацій різного масштабу по всьому світу покладаються на Zabbix як основну платформу моніторингу

**Архітектура Zabbix** складається з кількох підсистем, які можуть бути розміщені на різних машинах:

- Сервер моніторингу, в обов’язки якого входить отримання і обробка даних, їх аналіз і вироблення певних дій, залежно від ситуації. В ос-новному це

сповіщення адміністратора;

- База даних, в якості якої можуть використовуватися SQLite, MySQL, та Oracle;
- Веб-інтерфейс, який відповідає за управління моніторингом та ді-ями, а також за візуалізацію. Веб-інтерфейс системи написаний на PHP;
- Агент Zabbix, який запускається на тій машині/пристрої, з якої необхідно знімати дані. Його наявність хоч і бажана, але, якщо встановити його на пристрій неможливо, можна обійтися протоколом SNMP (Simple Network Management Protocol);
- Zabbix proxy - використовується в основному в тих випадках, коли необхідно вести моніторинг сотень і тисяч пристроїв для зниження навантаження на сервер моніторингу. Узагальнена архітектура системи Zabbix представлена на рисунку 2.5.

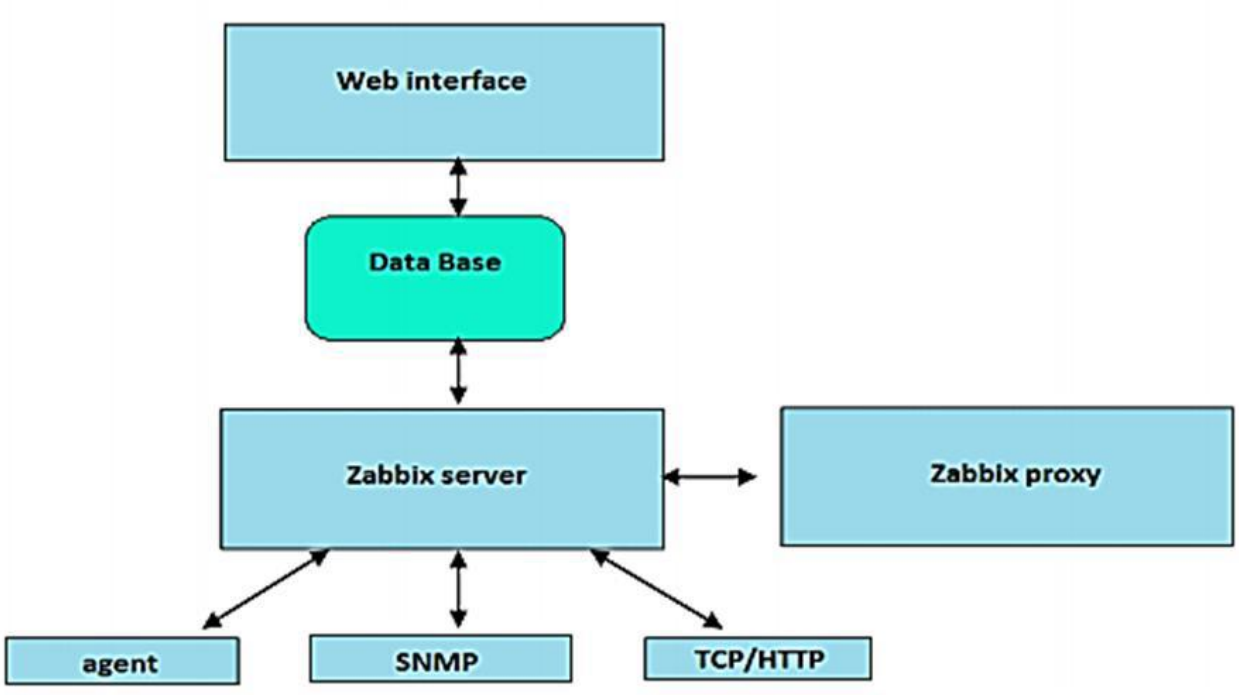



Рисунок 2.5 – Узагальнена архітектура системи моніторингу мережі Zabbix

2.7 Системи глибокого аналізу пакетів DPI та системи аналізу мережевого трафіку NTA.

**Система глибокого аналізу пакетів (Deep Packet Inspection – DPI)** - сукупна назва технології, що дозволяє проводити накопичення, аналіз, класифікацію, контроль і модифікацію мережевих пакетів в залежності від їх вмісту в реальному часі. Іноді вживають більш вузький термін - DPP (Deep Packet Processing), який має на увазі такі дії над пакетами, як модифікація, фільтрація

або перенаправлення. Сьогодні обидва терміни - DPI і DPP - часто використовуються як взаємозамінні.

**Модель розвитку DPI**

Технології інспекції трафіку розвивалися послідовно, кожна наступна успадковувала частина попередніх механізмів і додавала свої: 1) Shallow Packet Inspection, 2) Medium Packet Inspection, 3) Deep Packet Inspection.

**Shallow Packet Inspection** (Слабкий аналіз пакетів) - технологія аналізу трафіку, яка ґрунтується виключно на заголовках пакету (не аналізує вміст корисного навантаження пакета). Це перша реалізація технології інспектування трафіку. Менш вимоглива до ресурсів, ніж MPI і DPI, за рахунок чого, може обробляти набагато більші обсяги трафіку з високою точність визначення. Технологія широко поширена, на її основі працює більшість брандмауерів операційних систем, маршрутизаторів (ACL) і т.д. Але назву цієї технології не потрібно плутати з технологією stateful packet inspection - технологією перевірки трафіку на коректність.

**Medium Packet Inspection** (Середній аналіз пакетів) - технологія аналізу трафіку, яка ґрунтується на інспектуванні сесій і сеансів зв'язку ініційованих додатком, але встановлюваних шлюзом-посередником. Як правило, назва технології MPI заміщається позначенням «application proxy». Частково аналізує вміст пакетів по визначеним правилам. Не використовуються складні методи аналізу (сигнатурний і т.д.). Так само є однією з форм брандмауера.

Технологія **Deep Packet Inspection** (Глибокий аналіз пакетів) виникла через необхідність аналізувати, контролювати і управляти переданим трафіком. Технологія DPI отримала розвиток, перш за все, через стрімке зростання обчислювальних здібностей чіпів (процесорів), їх швидкодії.

**Покоління DPI систем**. У деяких джерелах зустрічається інформація про покоління DPI систем управління трафіком. Відзначається два покоління таких пристроїв:

Перше покоління: було пристосоване для вирішення більш вузьких завдань і мало тільки сигнатурний механізм аналізу.

Друге покоління: більш універсальні і масштабовані пристрої з аналізом, який має евристичний і поведінковий механізми, що мають в своєму складі інструменти класифікації та управліннями політиками.

Іноді поділяють за поколінням виходячи з продуктивності пристроїв: 1-е покоління до 10Гібіт / с, друге від 10 до 100 і третє покоління більш 100.

Технологію **DPI** використовують наступні сучасні рішення – Системи технічних засобів для забезпечення функцій оперативно-розшукових заходів (СОРЗ), Антивірусні рішення, Системи управління трафіком, сучасні міжмережеві екрани.

**Відмінність DPI систем від міжмережевих екранів:**

- DPI система аналізує не тільки заголовки пакетів, а й повний вміст трафіку на рівнях моделі OSI з другого і вище.

- DPI система може приймати рішення не тільки по вмісту пакетів, але і за непрямими ознаками, властивим якимось певним мережевим програмам і протоколам. Для цього може використовуватися статистичний аналіз (наприклад статистичний аналіз частоти зустрічі певних символів, довжини пакета, тощо).
- DPI система на відміну від брандмауера застосовує різні моделі дій над трафіком (класифікація, обмеження смуги, пріоритезація, маркування, кешування, тощо).

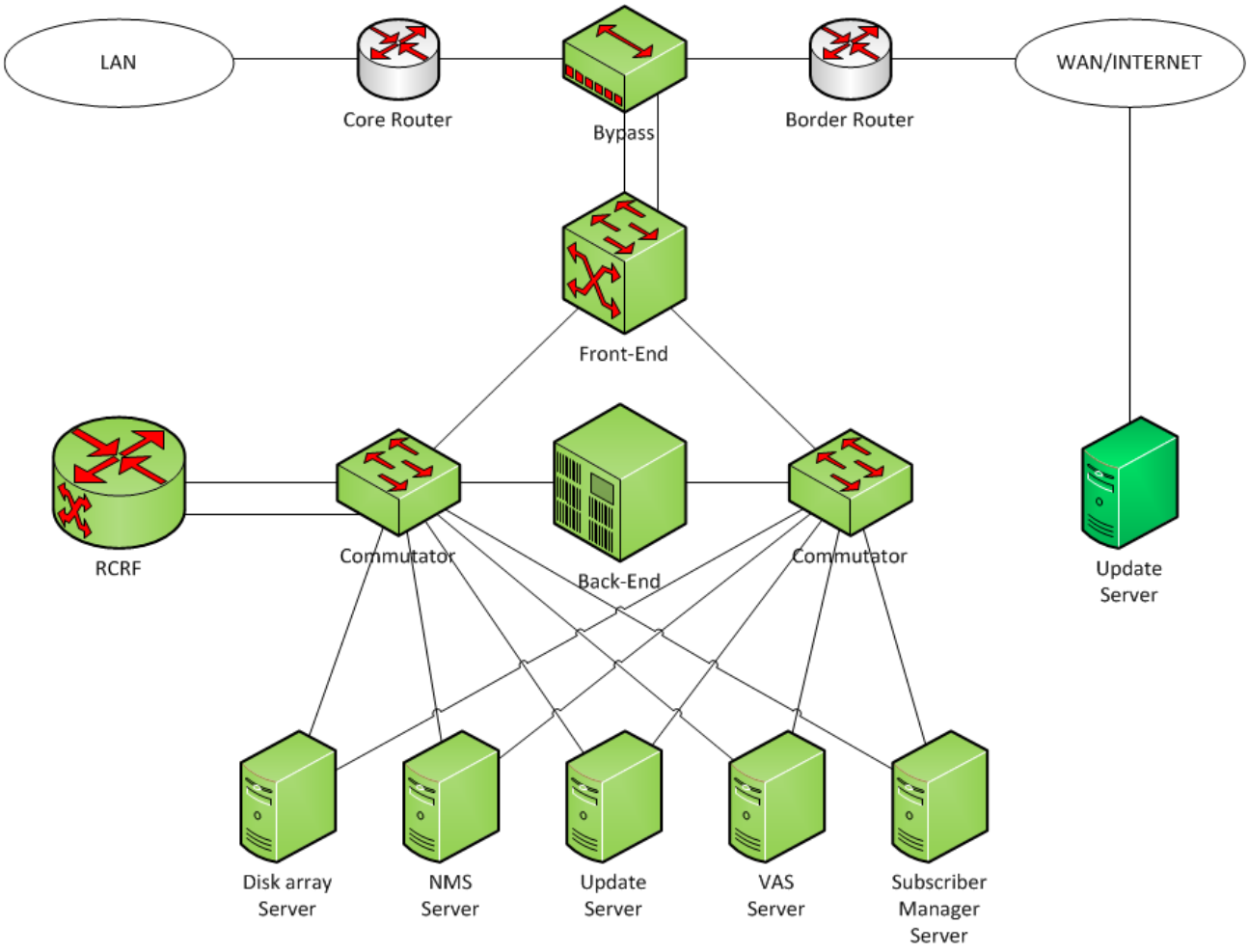



Рисунок 2.6 – Архітектура DPI систем

**Основні компоненти повноцінної DPI системи управління трафіком (рис. 2.6):**

**Bypass** - високопродуктивний комутатор, основне завдання якого пропускати трафік або безпосередньо (без обробки), або відправляти трафік на пристрій обробки і аналізу трафіку - **Front-End**, Bypass постійно стежить за зв'язком з Front-End компонентом і в випадках втрати зв'язку або отримання повідомлень від Front-End, про труднощі обробки трафіку, починає пропускати трафік безпосередньо до прикордонного маршрутизатора. Існують різні типи

Bypass інтерфейсів: мідні (не більше 1 Гбіт / с) і оптичні (від 1 до 100 Гбіт / с). Підтримується оптичне віддзеркалення трафіку «як є» і відповідно можливий збір статистики або підключення СОРЗ.

**Front-End** - могз DPI системи - центр обробки інформації відповідно до налаштованих / наявних політик. По суті це модульний центр обробки інформації.

**Back-End** - це високопродуктивний кеш-сервер для оперативного використання баз сигнатур, різної статистики, політик і різних правил перенаправлення трафіку. Основним завданням Back-End є моментальне надання інформації для Front-End компоненти. Як правило, кеш-сервера мають високу степ резервування своїх компонентів, підтримують гарячу заміну і вміють балансувати навантаження (для роботи в кластері).

**PCRF**-сервер (Policy and charging rules function) - один з основних компонентів DPI систем - сервер визначення правил і політик. Основна роль при отриманні від Front-End`а ідентифікатора користувача / абонента, повідомити Front-End`у відповідний номер політики. Далі Front-End запитує подробиці відповідної політики на Back-End'і. У деяких DPI системах роль PCRF компонента або частину функцій виконує Subscriber Manager сервер.

**Disk array Server** - Сервер дисковий масив призначений для зберігання великих обсягів інформації (статистики, різних баз даних, іноді копій трафіку). Як правило, взаємодіє тільки з Back-End`дом для актуалізації кеш-фонду останнього.

**NMS Server** (Network Management Server) - Сервер управління системою.

**Update Server** - Сервер оновлення. Основне завдання отримувати оновлення: системні оновлення, оновлення сигнатур і політик, моделей поведінки і т.д. з сервера поновлення виробника, через захищене з'єднання.

**Subscriber Manager** - Сервер і / або програмний компонент системи є координаційним центром DPI систем управління трафіком, реалізує можливості персоналізації і диференціювання абонентських послуг (прив'язка до користувача / пристрою / мережі).

**VAS Server** (Value Added Services Server) - Сервер додаткових сервісів. Виробники DPI систем управління трафіком інтенсивно розвивають VAS сервіси, через те, що це дає можливість використовувати, як свої додаткові сервіси, так і додаткові сервіси інших розробників / виробників, тим самим розширюючи можливості своєї системи.

**Системи аналізу мережевого трафіку (Network Traffic Analysis – NTA)** призначені для перехоплення потоків даних і виявлення ознак складних, найчастіше цільових атак (APT). З їх допомогою можна проводити ретроспективне вивчення мережевих подій, виявляти і розслідувати операції зловмисників в інформаційній інфраструктурі підприємства, а також ефективно реагувати на відповідні події. Такі системи відмінно доповнюють продукти класу Endpoint Detection and Response (EDR), можуть служити багатим джерелом

відомостей для SIEM-систем або центрів моніторингу та оперативного реагування на інциденти інформаційної безпеки.

Рішення класу NTA відстежують трафік і комунікації всередині корпоративної мережі і застосовують різні техніки (поведінковий аналіз, машинне навчання) для того, щоб швидко виявити, проаналізувати і обробити загрози, які в іншому випадку залишалися б прихованими. NTA може застосовуватися в мережах будь-якого масштабу, так само як і будь-якої архітектури: локальної, хмарної, гібридної.

Інтеграція з EDR- і SIEM-рішеннями забезпечує безшовну кореляцію даних. У такій схемі NTA відповідає за видимість інформації, переданої по мережі, EDR поставляє відповідні відомості від кінцевих точок, а SIEM агрегує журнали подій.

Основні функції NTA-рішень:

- аналіз трафіку як на периметрі мережі, так і всередині інфраструктури,
- виявлення атак за допомогою комбінації технологій виявлення,
- допомога в розслідуванні інцидентів.

Як правило, типова архітектура NTA-рішень включає:

- мережевий сенсор, який збирає трафік,
- сервери централізованого управління,
- консоль моніторингу (дашборда).

Деякі рішення NTA поєднують можливості пошуку проблем безпеки з автоматизованими завданнями з реагування на ризики і пом'якшенням наслідків інцидентів. Інструменти цього типу постійно шукають в мережі підозрілі або шкідливі дані. Якщо в результаті сканування вдалося щось виявити, то NTA діагностує проблему, щоб визначити, в чому саме полягає загроза безпеці. Ґрунтуючись на цьому діагнозі, продукт розгортає автоматизовані завдання, щоб допомогти нейтралізувати проблему, одночасно оповіщаючи співробітників інформаційної безпеки про неї.

Мета автоматизованих завдань - спробувати вирішити ситуацію без участі людини. Це скорочує час між пошуком і зняттям проблеми безпеки, дозволяючи команді фахівців вирішувати інші важливі питання. Типова схема роботи NTA представлена на рисунку 2.7.


Рисунок 2.7 – Типова схема роботи NTA

# 3. ТЕХНОЛОГІЯ МІЖМЕРЕЖЕВИХ ЕКРАНІВ ТА СИСТЕМ ВИЯВЛЕННЯ АТАК

## 3.1 Класифікація та архітектура міжмережевих екранів

**Міжмережевий екран (нім. Brandmauer, англ. firewall)** - це комплекс апаратних і програмних засобів в комп'ютерній мережі, який здійснює контроль і фільтрацію мережевих пакетів, що проходять через нього, відповідно до заданих правил.

Основним завданням мережевого екрану є захист мережі або окремих її вузлів від несанкціонованого доступу. Також мережеві екрани часто називають фільтрами, так як їх основне завдання - не пропускати (фільтрувати) пакети, що не підходять під критерії, що визначені в конфігурації.

**Типові можливості міжмережевих екранів (далі – МЕ):**

- фільтрація доступу до відомих незахищених служб;
- перешкоджання отриманню закритої інформації з захищеної підмережі, а також впровадження в захищену підмережу помилкових даних за допомогою вразливих служб;
- контроль доступу до вузлів мережі;
- може реєструвати всі спроби доступу як ззовні, так і з внутрішньої мережі, що дозволяє вести облік використання доступу в Інтернет окремими вузлами мережі;
- регламентування порядку доступу до мережі;
- повідомлення про підозрілу діяльність, спробах зондування або атаки на вузли мережі або сам екран.

**Особливості міжмережевих екранів:**

- Трафік ззовні всередину і навпаки повинен проходити через брандмауер.
- Трафік дозволений (або ні) на основі політики, що застосовується на брандмауері.
- Очікується, що МЕ не застрахований від проникнення. Брандмауери повинні бути висококваліфікованими пристроями із захищеними операційними системами.

**Політики МЕ щодо трафіку визначається на основі:**

- Діапазону IP-адрес (явно дозволяти або забороняти);
- Мережевих протоколів;
- Програм що діють в мережі;
- Типу вмісту трафіку;
- Спрямованості трафіку (вхідний чи вихідний)

Їх слід визначати на основі оцінки ризику організацій, ділової місії, вимог тощо.

**Особливості, які використовують МЕ для фільтрації трафіку:**

- IP адреса й значення полів протоколів - цей тип фільтрації використовується пакетним фільтром та брандмауерами, що перевіряють стан. Зазвичай використовується для обмеження доступу до певних послуг.
- Прикладний протокол - цей тип фільтрації використовується шлюзом на рівні додатків, який ретранслює та контролює обмін інформацією для конкретних прикладних протоколів.
- Ідентифікатор користувача - зазвичай для внутрішніх користувачів, які ідентифікують себе за допомогою певної форми технології безпечної автентифікації.
- Мережева активність - керує доступом на основі таких міркувань, як час або запит, швидкість запитів або інші схеми діяльності.

**Класифікація міжмережевих екранів**.

**1. Пакетні фільтри**.

Управління трафіком здійснюється на основі аналізу насутупної інформації, що міститься в пакеті:

- IP-адреса джерела в пакеті - адреса хоста, з якого прийшов пакет.
- IP-адреса одержувача в пакеті - адреса хоста, якому призначений пакет.
- Транспортний протокол, який використовується для взаємодії хостів відправника і одержувача, такий як TCP, UDP або ICMP.
- Можливо деякі характеристики комунікаційної сесії транспортного рівня, такі як порти джерела і одержувача (наприклад, TCP 80 для порту одержувача і TCP 1320 для порту джерела).
- Інтерфейс, через який проходить пакет, і напрямок (вхідний або вихідний).

Пакетні фільтри можуть бути реалізовані в наступних компонентах мережевої інфраструктури (рис. 3.1):

- Прикордонні маршрутизатори.
- Операційні системи.
- Персональні міжмережеві екрани.

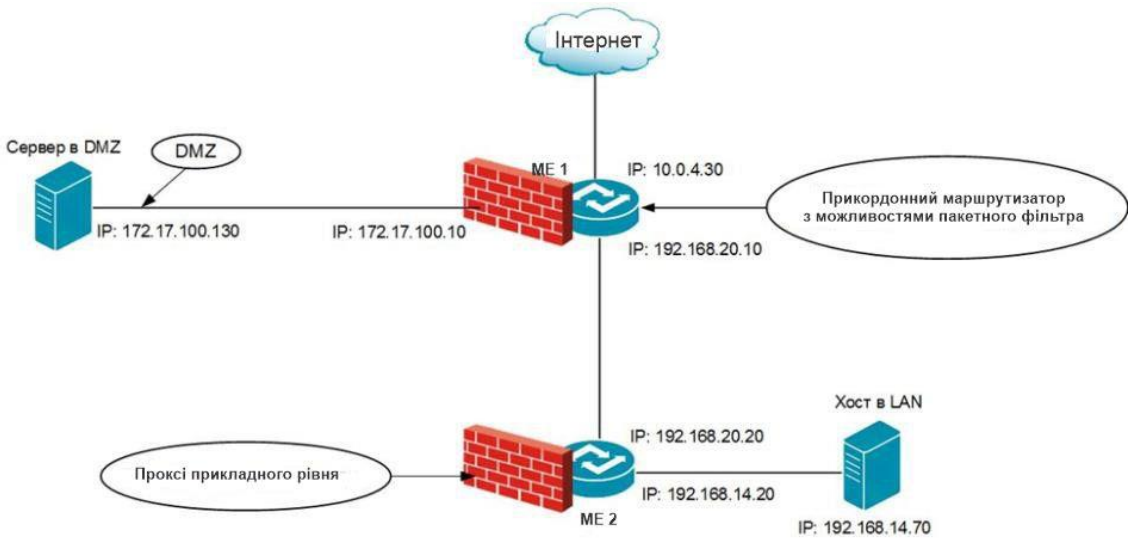



Рисунок 3.1 – Схема розташування пакетного фільтру в комп'ютерній мережі

**Переваги пакетних фільтрів:**

- основною перевагою пакетних фільтрів є їх швидкість;
- пакетний фільтр прозорий для клієнтів і серверів, так як не розриває TCP-з'єднання.

**Недоліки:**

- не аналізують дані більш високих рівнів;
- не підтримують можливість аутентифікації користувача;
- уразливі для атак, які використовують такі уразливості TCP/IP, як підробка (spoofing) мережевої адреси;
- важко конфігурувати.

**2. Пакетні фільтри з аналізом стану**.

Аналіз стану дає можливість відстеження стану з'єднання і блокування пакетів, які не відповідають очікуваному стану. Для цього виконується аналіз даних транспортного рівня. На відміну від фільтрації пакетів, інспекція стану *відстежує історію кожного з'єднання*, використовуючи для цього таблицю станів. Деталі записів таблиці станів зазвичай містять IP-адреса джерела, IP-адреса одержувача і інформацію про стан з'єднання.

**Переваги міжмережевих екранів з аналізом стану:**

- Дозволяють проходження пакетів тільки для встановлених з'єднань;
- Прозорі для клієнтів і серверів, так як не розривають TCP-з'єднання.

**Недоліки міжмережевих екранів з аналізом стану:**

- Реально використовуються тільки в мережевій інфраструктурі TCP/IP. Хоча треба зазначити, що міжмережеві екрани з аналізом стану можна реалізувати в інших мережевих протоколах тим же способом, що і пакетні

фільтри.

**3. Міжмережеві екрани прикладного рівня**

Аналізують протокол на прикладному рівні, порівнюючи поведінку протоколу з певними розробленими профілями і визначаючи відхилення в поведінці. Це дозволяє міжмережевому екрану дозволяти або забороняти доступ, ґрунтуючись на тому, як виконується програма.

Міжмережеві екрани прикладного рівня доступні для багатьох протоколів, включаючи HTTP, БД (SQL), поштові (SMTP, POP, IMAP), VoIP і XML.

**Переваги:**

- має можливість виконувати аутентифікацію користувача;
- менш уразливий для атак підробки адреси;
- мають більше можливостей аналізувати весь мережевий пакет, а не тільки мережеві адреси і номери портів;
- створюють більш докладні логи;

**Недоліки:**

- витрачається багато часу на аналіз кожного пакета;
- обробляють обмежену кількість мережевих додатків і протоколів і не можуть автоматично підтримувати нові мережеві додатки та протоколи.

**4. Проксі-шлюзи прикладного рівня**

Комбінують управління доступом на нижньому рівні з функціональністю верхнього рівня. Вони мають проксі-агента, що діє як посередник між двома хостами, які хочуть взаємодіяти один з одним, і ніколи не допускає прямої взаємодії між ними. Результатом успішної спроби встановлення з'єднання є створення двох окремих з'єднань - одне між клієнтом і проксі-агентом, інше - між проксі-агентом і реальним сервером.

**Переваги:**

- проксі має можливість запросити аутентифікацію користувача;
- є менш уразливими для атак підробки адреси;
- більше можливостей аналізувати весь мережевий пакет, а не тільки мережеві адреси і номери портів;
- створюють більш докладні логи;

**Недоліки:**

- витрачає багато часу на аналіз кожного пакета;
- обробляють обмежену кількість мережевих додатків і протоколів і не можуть автоматично підтримувати нові мережеві додатки та протоколи.

**5. Виділені проксі-сервера**

Використовуються для зменшення навантаження на міжмережевий екран і виконання спеціалізованого фільтрування та створення логів, яке може бути важко виконати самому міжмережевому екрану (рис. 3.2).

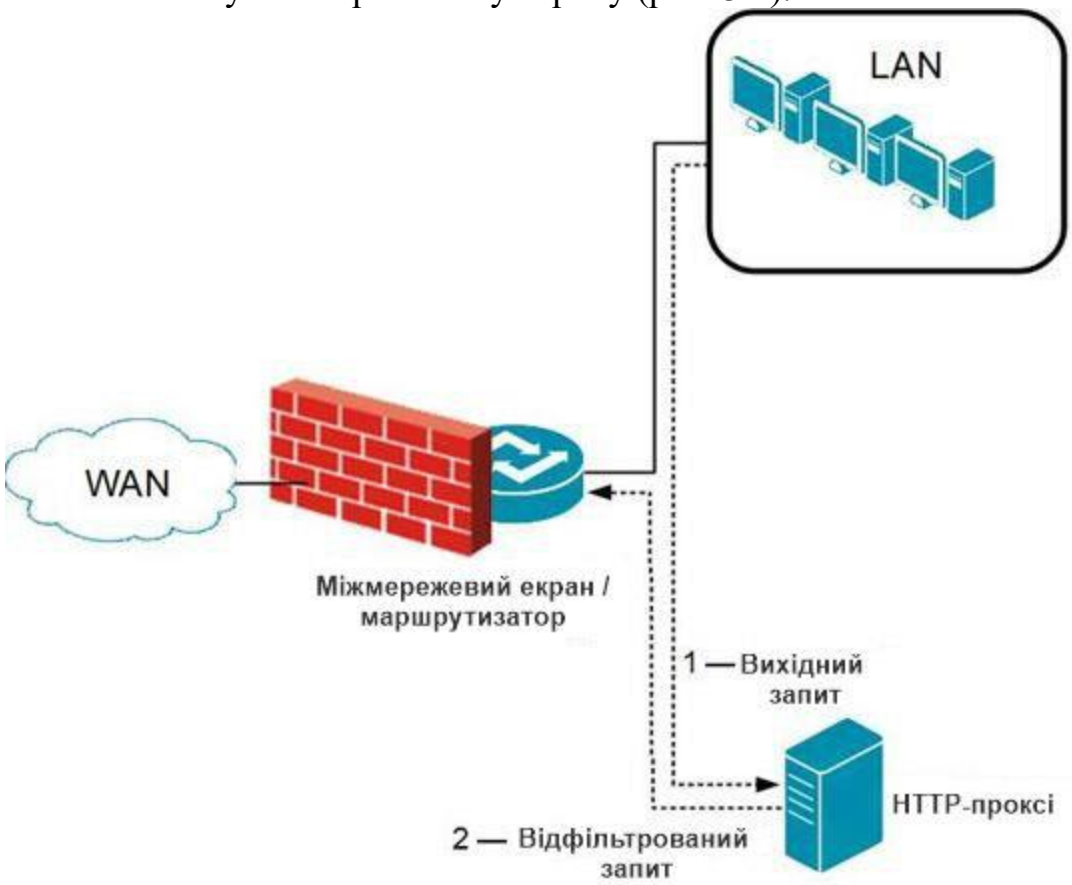



Рисунок 3.2 – Схема розташування виділеного проксі серверу в комп'ютерній мережі

**Крім зазначених типів міжмережевих екранів також виділяють наступні:**

**Кінцеві точки VPN.** Для шифрування певного трафіку між захищеними мережами. Двома найбільш часто використовуваними архітектурами VPN є шлюз-шлюз і хост-шлюз.

**Гібридні технології міжмережевих екранів.** Багато міжмережевих екранів поєднують функціональності кількох різних типів міжмережевих екранів. Наприклад, багато виробників прикладних проксі реалізують базову функціональність пакетних фільтрів.

**Додаткові можливості керування доступом в мережу.** Необхідність вирішення вхідних з'єднань не тільки після виконання аутентифікації віддаленого користувача, але і перевірки параметрів безпеки для користувача комп'ютера.

**Уніфіковане управління загрозами**

**Міжмережеві екрани для веб-додатків**

**Міжмережеві екрани для віртуальних інфраструктур**
**Міжмережеві екрани для окремих хостів і домашніх мереж**

Слід зазначити, що **міжмережеві екрани ефективні тільки для трафіку, який вони можуть аналізувати.** Незалежно від обраної технології брандмауера, якщо він не може розуміти трафік, який проходить через нього, він не може змістовно дозволити або заборонити його.

Багато мережевих протоколів використовують криптографію, щоб приховати вміст. Прикладами таких протоколів є IPsec, TLS, SSH і SRTP (Secure Real-time Transport Protocol). **Міжмережеві екрани не можуть також читати зашифровані прикладні дані**, наприклад, якщо поштове повідомлення зашифровано за допомогою протоколів S/MIME або OpenPGP. Іншим прикладом обмеження є те, що багато міжмережевих екраныв не розуміють тунелюваний трафік, навіть якщо він не зашифрований. Наприклад, IPv6 трафік може тунелюватись в IPv4 багатьма різними способами. Вміст навіть може бути незашифрований, але якщо міжмережевий екран не розуміє використовуваний механізм тунелювання, трафік не може бути проінтерпретований.

У всіх цих випадках правила брандмауера повинні визначити, що робити з трафіком, якщо вони не можуть його зрозуміти.

**Політика міжмережевого екрану.**

Політика брандмауера визначає, як міжмережевий екран буде обробляти мережевий трафік для певних IP-адрес і діапазонів адрес, протоколів, додатків і типів вмісту (наприклад, активного вмісту). Перед розробкою політики брандмауера слід проаналізувати ризики і визначити типи трафіку, які необхідні організації. Аналіз ризику повинен ґрунтуватися на оцінцы загроз і вразливостей.

**Існують наступні типи політик міжмережевого екрану:**

- політики, засновані на IP-адресації і протоколах (IP-адреси та інші характеристики IP, IPv6, протоколи TCP і UDP, протокол ICMP);
- політики, засновані на додатках;
- політики, засновані на ідентифікації користувача;
- політики, засновані на мережевій активності.

**Міжмережеві екрани з можливостями NAT.**

NAT (Network Address Translation – перетворення мережевих адрес) є методом, за допомогою якого виконується перетворення IP-адрес при пересиланні пакета з однієї області адрес в іншу, забезпечуючи прозору маршрутизацію між кінцевими хостами.

Маршрутизатор NAT розташований на кордоні між двома областями адрес і перетворює адреси в IP-заголовках таким чином, що пакет коректно маршрутизується при переході з однієї області в іншу. Маршрутизатор NAT має

з'єднання з декількома областями адрес, при цьому він не повинен поширювати некоректну для області інформацію (наприклад, за допомогою протоколів маршрутизації) про мережах з однієї області адрес в іншу (рис. 3.3).

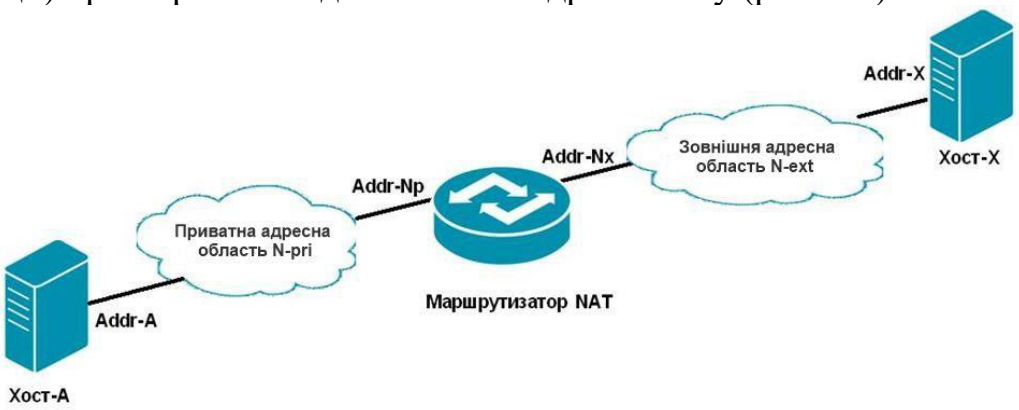



Рисунок 3.3 – Схема розташування міжмережевого екрану з можливостями NAT

**Існує 3 базових концепції трансляції адрес**: статична (Static Network Address Translation), динамічна (Dynamic Network Address Translation), маскарад (NAPT, NAT Overload, PAT).

**Статичний NAT** - Відображає незареєстровану IP-адресу на зареєстровану IP-адресу на підставі один до одного. Особливо корисно, коли пристрій повинен бути доступним ззовні мережі.

**Динамічний NAT** - Відображає незареєстровану IP-адресу на зареєстровану адресу з групи зареєстрованих IP-адрес. Динамічний NAT також встановлює безпосереднє відображення між незареєстрованою та зареєстрованою адресами, але відображення може змінюватися в залежності від зареєстрованої адреси, доступної в пулі адрес, під час комунікації.

**Перевантажений NAT** (NAPT, NAT Overload, PAT, маскарадінг) - форма динамічного NAT, який відображає кілька незареєстрованих адрес в єдиний зареєстрований IP-адреса, скориставшись різними портами. Відомий також як PAT (Port Address Translation). При перевантаженні кожен комп'ютер в приватній мережі транслюється в ту же саму адресу, але з різним номером порту.

**Топологія мережі при використанні міжмережевих екранів**

Так як початкової функцією брандмауера є запобігання небажаного вхідного трафіку в мережу (і в деяких випадках вихідного), міжмережеві екрани повинні бути розташовані в точках входу на логічних межах мережі. Зазвичай це означає, що міжмережевий екран є вузлом, в якому мережевий трафік розділяється на кілька потоків, або збирається разом в єдиний потік (рис. 3.4).

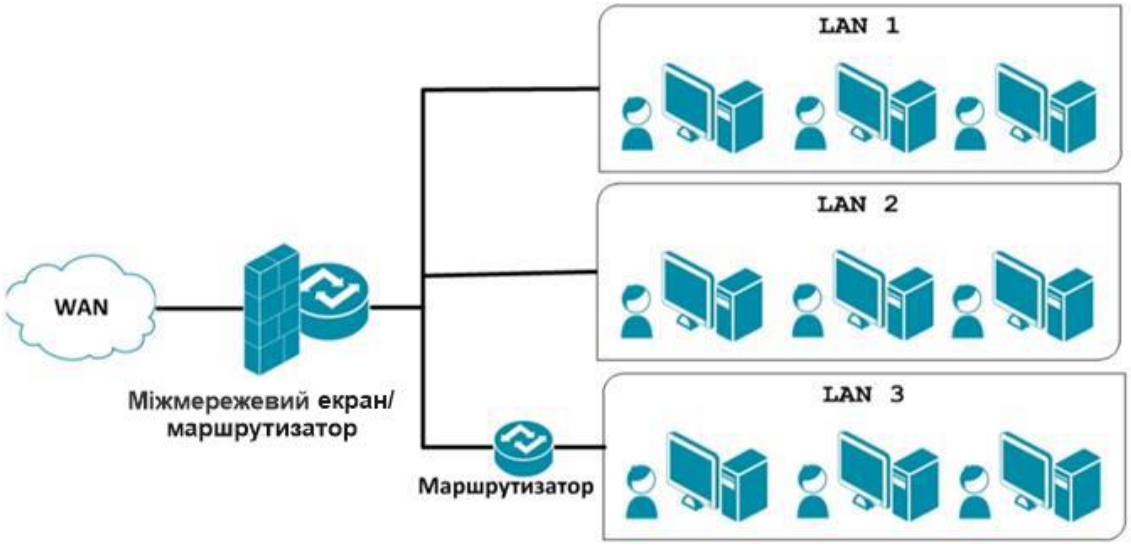



Рисунок 3.4 – Звичайна маршрутизуєма мережа з міжмережевим екраном

Багато апаратних пристроїв міжмережевого екранування мають функціональність, звану DMZ (демілітаризована зона). Хоча і не існує єдиного визначення для DMZ, зазвичай вони є інтерфейсами в міжмережевому екрані, для яких можливо задавати правила маршрутизації, і аналогічні інтерфейсам, що розташовані на захищеній стороні брандмауера (рис. 3.5). У DMZ розташовують публічно доступні сервера, такі як веб-сервер або поштовий сервер.

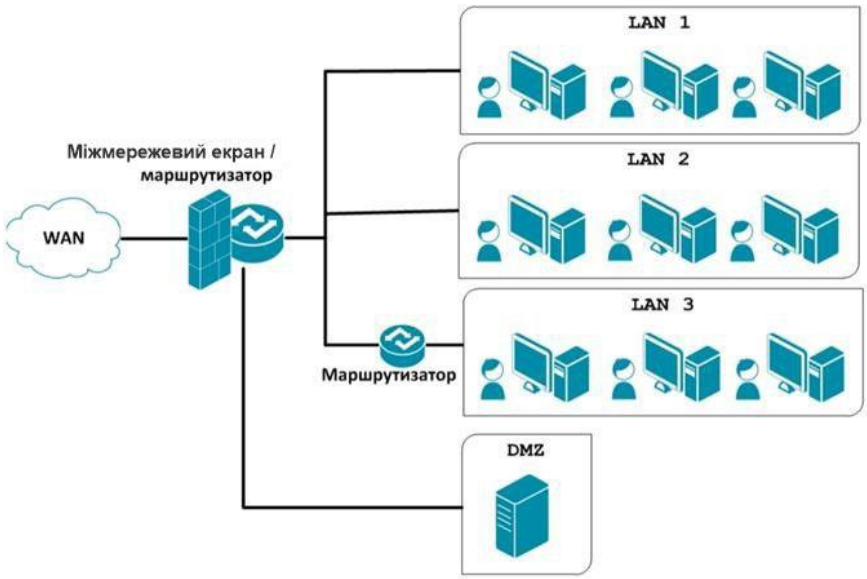



Рисунок 3.5 – Комп’ютерна мережа з виділеною DMZ

DMZ-мережі призначені для розташування систем і ресурсів, які не повинні бути розміщені у внутрішніх захищених мережах, але до яких необхідно отримати доступ або тільки ззовні, або тільки зсередини, або і ззовні, і зсередини (рис. 3.6). Причина в тому, що ніколи не можна гарантувати, що ці системи і

ресурси не можуть бути зламані. Але злом цих систем не повинен автоматично означати доступ до всіх внутрішніх систем.

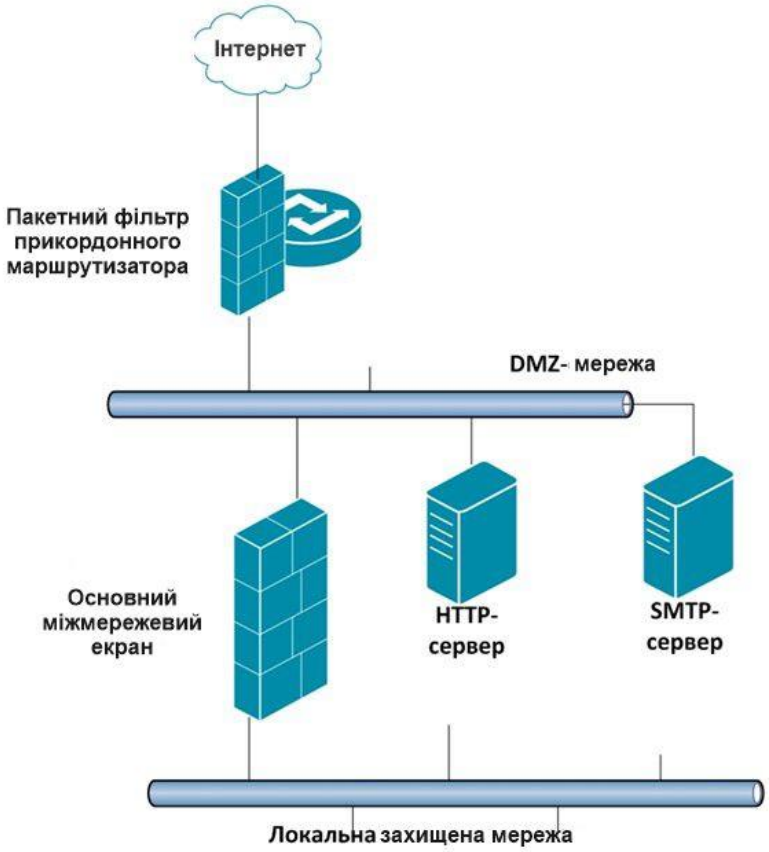



Рисунок 3.6 – Приклад оточення міжмережевого екрану з однією DMZ-мережею

Однією з конфігурацій DMZ-мережі є так звана «service leg» конфігурація брандмауера (рис. 3.7). У цій конфігурації міжмережевий екран має як мінімум три мережевих інтерфейсу. Один мережевий інтерфейс з'єднується з інтернетом, інший з'єднується з внутрішньою мережею, третій мережевий інтерфейс формує DMZ-мережу.


Рисунок 3.7 – Конфігурація Service Leg DMZ

У випадку оточення міжмережевого екрану двома DMZ-мережами (рис. 3.8), **прикордонний маршрутизатор** буде фільтрувати пакети і забезпечувати захист серверів, **перший міжмережевий екран** буде забезпечувати управління доступом і захист серверів внутрішньої DMZ в разі, якщо зовнішні сервера атаковані. Внутрішньо доступні сервери розміщуються у внутрішній DMZ, розташованої між основним і внутрішнім міжмережевими екранами; міжмережеві екрани будуть забезпечувати захист і управління доступом для внутрішніх серверів, захищених як від зовнішніх, так і від внутрішніх атак.

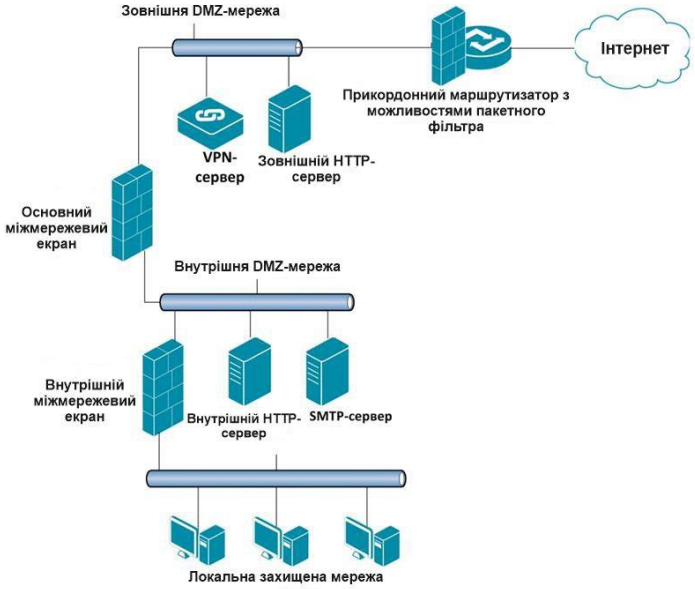



Рисунок 3.8 – Приклад оточення міжмережевого екрану з двома DMZ-мережами

VPN застосовується для забезпечення безпечних мережевих з'єднань з використанням мереж, які не є довіреними системами. Наприклад, технологія VPN все частіше створюється для надання віддаленого доступу користувача до мереж організації через Інтернет (рис. 3.9).

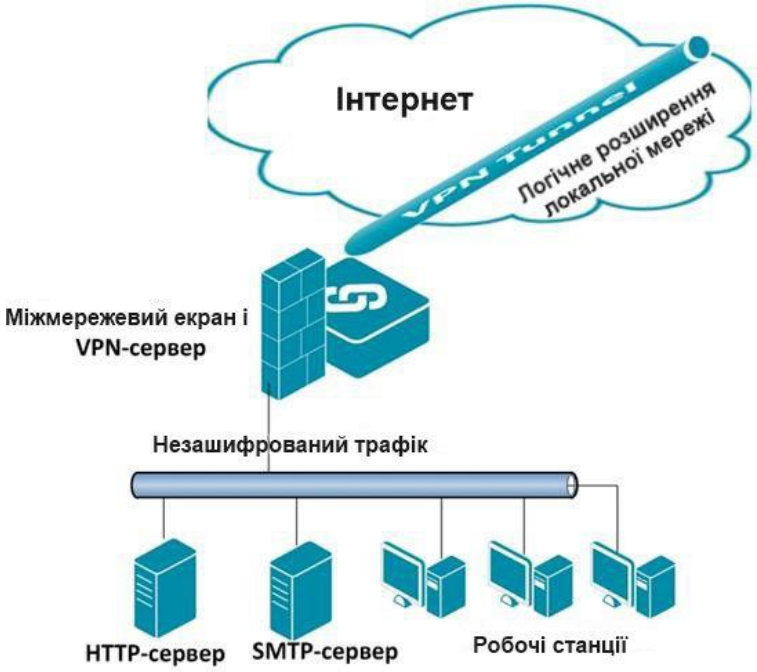



Рисунок 3.9 – Приклад поєднання брандмауера і кінцевої точки VPN

Екстранет може бути визначений як business-to-business інтранет. Ця мережа дозволяє забезпечити обмежений, керований доступ віддалених користувачів за допомогою тієї ж форми аутентифікації і шифрування, які є в VPN. Метою інтранет є представлення доступу до потенційно чутливої інформації віддаленим користувачам або організаціям, але при цьому забороняючи доступ всім іншим зовнішнім користувачам і системам (рис. 3.9).

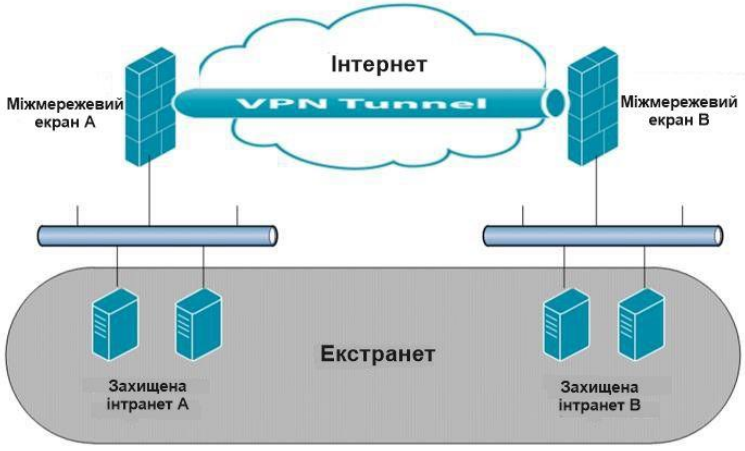



Рисунок 3.9 – VPN й екстранет, що зєднує дві інтранет мережі

**Принципи ефективної роботи МЕ:**

- визначити всі вимоги, які накладає зовнішнє оточення на функціонування брандмауера;
- створити політику брандмауера, в якій визначено, як слід обробляти вхідний і вихідний трафік;
- розробити набір правил брандмауера, які реалізують політику безпеки в організації та забезпечують максимальну продуктивність брандмауера. Проаналізувати продуктивність брандмауера;
- управляти архітектурою, політиками, ПЗ і іншими компонентами міжмережевого екрану слід протягом усього часу його функціонування.

3.2 Міжмережевий екран ОС Linux iptables

До появи iptables для забезпечення можливостей мережевого екрана в Linux використовувались проекти ipchains в Linux 2.2 и ipfwadm в Linux 2.0, що оснований на ipfw із системи BSD. Проекти ipchains и ipfwadm змінювали роботу стека протоколів ядра Linux, оскільки до появи netfilter в архітектурі ядра не існувало можливостей для підключення додаткових модулів управління пакетами. Iptables зберіг основну ідею ipfwadm – список правил, який складався із критеріїв і дій, що виконувались, якщо пакет відповідає критеріям. В ipchains була представлена нова концепція – можливість створювати нові ланцюги правил й перехід пакетів між ланцюгами, а в iptables концепція була розширена до чотирьох таблиць, які відрізняються від ланцюга правил по задачам – фільтрація, NAT і модифікація пакетів. Також iptables розширив можливості Linux в області визначення стану, дозволяючи створювати міжмережеві екрани, що працюють на сеансовому рівні.

Фреймворк netfilter, до складу якої входять iptables, дозволяє системному адміністратору визначити правила роботи з мережевими пакетами. Правила згруповані у ланцюги, кожен ланцюг - це упорядкований список правил. Ланцюги згруповані в таблиці - кожна таблиця пов'язана з різним видом обробки пакетів.

Кожне правило містить специфікацію, які пакети відповідають йому, і ціль, яка вказує, що робити з пакетом, якщо він відповідає цьому правилу. Кожен мережевий пакет, що прибуває на комп'ютер або залишає його з комп'ютера, проходить щонайменше один ланцюг, і кожне правило цього ланцюга намагається відповідати пакету. Якщо правило відповідає пакету, обхід зупиняється, а ціль правила диктує, що робити з пакетом.

Взагалі **netfilter –** це компонент ядра Linux, що забезпечує фільтрацію і модифікацію трафіку. Власне, саме він і є фаєрволом (рис. 3.10).

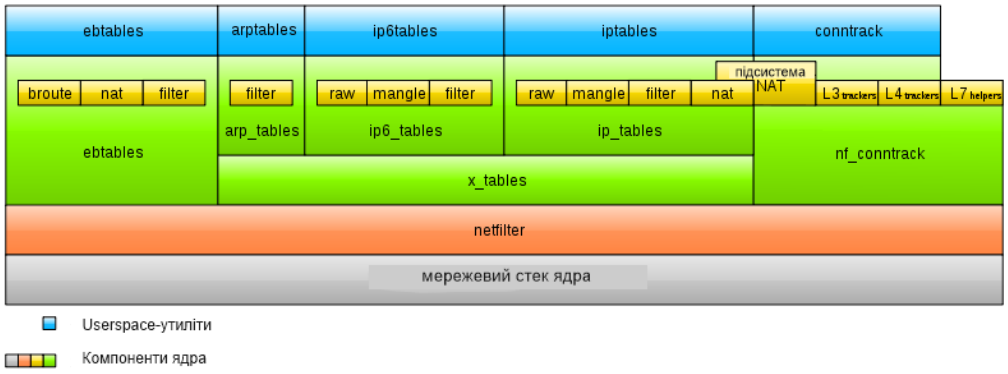



Рисунок 3.10 – Компоненти **netfilter**

До складу **netfilter** входять наступні модулі:

- ip_tables - фаєрвол для протоколу IPv4. Забезпечує фільтрацію пакетів, модифікацію їх заголовків і трансляцію мережевих адрес.
- ip6_tables - фаєрвол для протоколу IPv6. Забезпечує фільтрацію пакетів і модифікацію їх заголовків.
- arp_tables - фаєрвол для протоколів ARP і RARP. Забезпечує фільтрацію і модифікацію пакетів.
- x_tables - бекенд для ip_tables, ip6_tables і arp_tables. У цьому модулі визначені основні операції для роботи з фаєрволом «таблично-ланцюжкової» структури і їх компонентами.
- ebtables - Ethernet-фаєрвол (префікс eb від Ethernet Bridge). На відміну від трьох перерахованих вище фаєрволів, що працюють з протоколами мережевого і більш високих рівнів, ebtables працює на канальному рівні, виконуючи фільтрацію і модифікацію ethernet-кадрів, що проходять через мережеві мости, якщо такі є на хості.

**IPTABLES**

**iptables** - утиліта командного рядка, є стандартним інтерфейсом управління роботою брандмауера NETFILTER для Linux ядер версій 2.4 та 2.6. При отриманні пакету, він передається Netfilter для прийняття ним рішення: прийняти, обробити, або відкинути його на основі правил заданих за допомогою iptables. Таким чином, iptables це все, що потрібно для управління брандмауером.

iptables може відстежувати стан з'єднання і перенаправляти, змінювати або фільтрувати пакети, ґрунтуючись не тільки на даних з їх заголовків (джерело, одержувач) або вмісті пакетів, а й на підставі даних про з'єднання. Така можливість фаєрвола називається *stateful-фільтрацією*.

**Ключові поняття iptables**

**Критерій** - логічне вираження, що аналізує властивості пакета і/або з'єднання і визначає чи підпадає даний конкретний пакет під дію поточного правила. Критерій може бути загальним, неявним і явним.

**Дія (мета)** - опис дії, яку треба виконати з пакетом і/або з'єднанням в тому випадку, якщо вони підпадають під дію цього правила. Стандартні дії доступні у всіх ланцюжках:

- ACCEPT - підтвердити, передати на обробку наступним правилом;
- DROP - видалити, більше над пакетом ніякої обробки не проводити;
- QUEUE - передати на аналіз зовнішній програмі;
- RETURN - повернути на аналіз в попередній ланцюжок
- REJECT - пакет відхиляється. На відміну від DROP, де пакет просто відкидається, в даному випадку відправнику буде відправлено ICMP-повідомлення "Port unreachable" ( «Порт недоступний»). За допомогою опції - REJECT можна змінити тип ICMP повідомлення. Дія REJECT може бути використана тільки з ланцюжками INPUT, OUTPUT і FORWARD.

**Лічильник** - компонент правила, що забезпечує облік кількості пакетів, які потрапили під критерій даного правила. Також лічильник враховує сумарний обсяг таких пакетів в байтах.

**Правило** - складається з критерію, дії і лічильника. Якщо пакет відповідає критерію, до нього застосовується дія, і він враховується лічильником. Критерію може і не бути – тоді він неявно позначається критерієм «всі пакети». Вказувати дію теж не обов'язково - за відсутності дії правило буде працювати тільки як лічильник.

**Ланцюжок** - упорядкована послідовність правил. Ланцюжки можна розділити на призначені для користувача і базові.

**Базовий ланцюжок** - ланцюжок, що створюється за замовчуванням при ініціалізації таблиці. Кожен пакет, в залежності від того, чи призначений він самому хосту, згенерований ним або є транзитним, повинен пройти покладений йому набір базових ланцюжків різних таблиць. Крім того, базовий ланцюжок відрізняється від призначеного для користувача наявністю «дії за замовчуванням» (default policy). Ця дія застосовується до тих пакетів, що не були оброблені іншими правилами цього ланцюжка. Імена базових ланцюжків завжди записуються в верхньому регістрі (PREROUTING, INPUT, FORWARD, OUTPUT, POSTROUTING).

**Користувацький ланцюжок** - ланцюжок, створений користувачем. Може використовуватися тільки в межах своєї таблиці. Рекомендується не використовувати для таких ланцюжків імена у верхньому регістрі, щоб уникнути плутанини з базовими ланцюжками і вбудованими діями.

**Таблиця** - сукупність базових і призначених для користувача ланцюжків, об'єднаних загальним функціональним призначенням. Імена таблиць (як і модулів

критеріїв) записуються в нижньому регістрі, так як в принципі не можуть конфліктувати з іменами призначених для користувача ланцюжків. При виклику команди iptables таблиця вказується в форматі -t ім'я_таблиці. При відсутності явного вказівки, використовується таблиця filter.

**DNAT** - від англ. Destination Network Address Translation - Зміна Мережевої Адреси Одержувача. DNAT - це зміна адреси призначення в заголовку пакета. Найчастіше використовується в парі з SNAT.

**SNAT** - від англ. Source Network Address Translation - Зміна Мережевого Адреси Відправника. SNAT - це зміна вихідної адреси в заголовку пакета. Основне застосування - використання єдиного реального IP-адреси кількома комп'ютерами для виходу в Інтернет.

**Ланцюги та таблиці.**

На рисунку 3.11 представлено схему проходження пакетів в ланцюжках iptables.

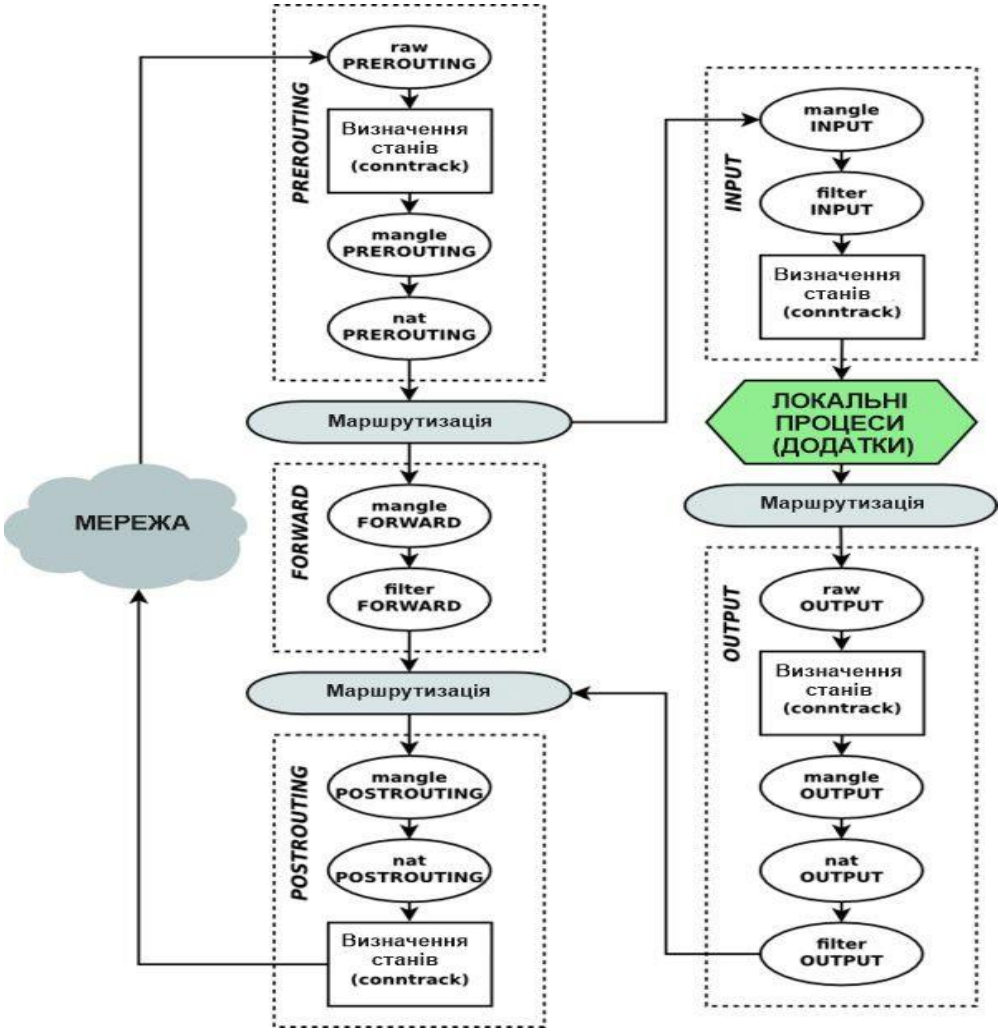



Рисунок 3.11 – Схема проходження пакетів в ланцюжках iptables

**Таблиця Mangle.**

Ця таблиця призначена, головним чином для внесення змін в заголовки пакетів (mangle - спотворювати, змінювати).

У цій таблиці не слід робити будь-якого роду фільтрацію, маскування або перетворення адрес (**DNAT, SNAT, MASQUERADE**).

У таблиці Mangle допускається (основні дії):

- Дія **TOS** виконує установку бітів поля Type of Service в пакеті. Це поле використовується для призначення мережевий політики обслуговування пакета, тобто задає бажаний варіант маршрутизації.
- Дія **TTL** використовується для установки значення поля TTL (Time To Live) пакета.
- Дія **MARK** встановлює спеціальну мітку на пакет, яка потім може бути перевірена іншими правилами в iptables або іншими програмами, наприклад iproute2. За допомогою "міток" можна управляти маршрутизацією пакетів, обмежувати трафік і т.п.

Таблиця mangle містить наступні ланцюжки:

- **PREROUTING** - дозволяє модифікувати пакет до прийняття рішення про маршрутизації.
- **INPUT** - дозволяє модифікувати пакет, призначений самому хосту.
- **FORWARD** - ланцюжок, що дозволяє модифікувати транзитні пакети.
- **OUTPUT** - дозволяє модифікувати пакети, які виходять від самого хоста.
- **POSTROUTING** - дає можливість модифікувати всі вихідні пакети, як згенеровані самим хостом, так і транзитні.

**Таблиця Nat**

Ця таблиця використовується для виконання перетворень мережевих адрес NAT (Network Address Translation). Тільки перший пакет з потоку проходить через ланцюжки цієї таблиці, трансляція адрес або маскування застосовуються до всіх наступних пакетів в потоці автоматично.

Дія **DNAT** (Destination Network Address Translation) здійснює перетворення адрес призначення в заголовках пакетів. Іншими словами, цією дією створюється перенаправлення пакетів на інші адреси, відмінні від зазначених у заголовках пакетів.

**SNAT** (Source Network Address Translation) використовується для зміни вихідних адрес пакетів. За допомогою цієї дії можна приховати структуру локальної мережі, а заодно і розділити єдиний зовнішній IP адреса між комп'ютерами локальної мережі для виходу в Інтернет. В цьому випадку брандмауер, за допомогою SNAT, автоматично виробляє пряме і зворотне перетворення адрес, тим самим даючи можливість виконувати підключення до

серверів в Інтернеті з комп'ютерів в локальній мережі.

Маскування (**MASQUERADE**) застосовується в тих же цілях, що і SNAT, але на відміну від останньої, MASQUERADE дає більш сильне навантаження на систему. Відбувається це тому, що кожен раз, коли потрібне виконання цієї дії - проводиться запит IP адреси для зазначеної в правилі мережевого інтерфейсу, в той час як для SNAT IP адреса вказується безпосередньо. Однак, завдяки такій відмінності, MASQUERADE може працювати в випадках з динамічною IP адресою, тобто коли ви підключаєтеся до Інтернет, скажімо через PPP, SLIP або DHCP.

**REDIRECT** - підміняє номер порту в TCP- або UDP-пакеті, а також підміняє адресу призначення на свою власну.

**SAME** - в залежності від ланцюжка (PREROUTING або POSTROUTING) може працювати як DNAT або SNAT. Однак, при вказівці (в параметрі --to-ip) одного або декількох діапазонів IP-адрес, визначає для кожного нового з'єднання підставляється адреса не випадково, а базуючись на IP-адресу клієнта. Таким чином, адреса для підміни залишається постійною для одного і того ж клієнта при повторних з'єднань (що не виконується для звичайних DNAT / SNAT). У деяких випадках це буває важливим.

**NETMAP** - дозволяє «прокинути» цілу мережу. Наприклад, для шлюзу, що стоїть між мережами 192.168.1.0/24 і 192.168.2.0/24 можна організувати цей прокид.

**MIRROR** - міняє місцями адресу джерела і призначення і висилає пакет назад. Створено виключно для демонстраційних цілей. Однак, може, наприклад, застосовуватися для захисту від сканування портів - в результаті атакуючий сканує свої власні порти. Однак наявність двох зустрічних MIRROR'ов на двох хостах може спричинити нескінченне блукання одних і тих же пакетів, що забиває канал. Тому необхідно застосовувати цю дію з обережністю.

Таблиця **nat** містить наступні ланцюжки:

- **PREROUTING** - в цей ланцюжок пакети потрапляють до прийняття рішення про маршрутизацію. По суті, термін «рішення про маршрутизації» має на увазі розподіл трафіку на вхідний (призначений самому хосту) і транзитний (що йде через цей хост на інші хости). Саме на даному етапі потрібно проводити операції пробросу (DNAT, REDIRECT, NETMAP).
- **OUTPUT** - через цей ланцюжок проходять пакети, що згенеровані процесами самого хоста. На даному етапі буде досягнуто потрібного результату операції проброса, так локально згенеровані пакети не проходять цілий ряд PREROUTING і не обробляються її правилами.
- **POSTROUTING** - через цей ланцюжок проходять всі вихідні пакети, тому саме в ньому доцільно проводити операції маскарадінгу (SNAT і MASQUERADE).

**Таблиця Filter:**

Таблиця **filter** використовується виключно для фільтрації пакетів. Для прикладу вона може DROP, LOG, ACCEPT чи REJECT пакети без проблем, як в інших таблицях. В цій таблиці є три ланцюжка.

Таблиця filter містить наступні ланцюжки:

- **INPUT -** цей ланцюжок обробляє трафік, що надходить безпосередньо самому хосту.
- **FORWARD -** дозволяє фільтрувати транзитний трафік.
- **OUTPUT -** цей ланцюжок дозволяє фільтрувати трафік, що виходить від самого хоста.

**Дії:**

**ACCEPT** - пропуск пакету. Пакет залишає поточну базовий ланцюжок і прямує далі по потоковій діаграмі.

**REJECT** - заблокувати пакет і повідомити його джерелу про відмову. За замовчуванням про відмову повідомляється відправкою відповідного ICMP-пакету «icmp-port-unreachable».

**DROP** - заблокувати пакет, та не повідомляти джерелу про відмову. Більш краща при фільтрації трафіку на інтерфейсах, підключених до Інтернету, так як знижує інформативність сканування портів хоста зловмисниками.

**Таблиця Raw**

Призначена для виконання дій з пакетами до їх обробки системою conntrack. В даній таблиці не будуть спрацьовувати критерії, яким для коректної роботи необхідний conntrack (це критерії conntrack, connmark, connlimit, connbytes).

**Ланцюжки**

- **PREROUTING –** вхідні пакети потрапляють в цей ланцюжок раніше, ніж в будь-який інший з ланцюжків iptables, і до обробки їх системою conntrack.
- **OUTPUT** - аналогічно для пакетів, що згенеровані самим хостом.

**Дії**

- **NOTRACK** - дозволяє запобігти обробку пакетів системою conntrack.
- **CT** -Дозволяє задати різні налаштування conntrack, відповідно до яких буде оброблятися з'єднання, відкрите даним пакетом.
- **RAWDNAT** - дозволяє виконувати «кидок» адрес і портів «сирим» методом - без використання системи conntrack, тобто без урахування станів з’єднань.

**Механізм Conntrack**

Основний критерій, який використовується для контролю стану з'єднання. Він надає ефективний набір інструментів, що дозволяє використовувати інформацію системи conntrack про стан з'єднання.

**--ctstate** маска

Маска містить перерахування через кому список можливих станів сполуки. Пакет вважається таким, що задовольняє критерію, якщо з'єднання, по якій він проходить, знаходиться в одному з перерахованих станів.

Можливі стани:

- **NEW** - з'єднання не відкрито, тобто пакет є першим в з'єднанні.
- **ESTABLISHED** - пакет належить до вже встановленого з'єднання. Зазвичай такі пакети приймаються без додаткової фільтрації, як і в випадку з RELATED.
- **RELATED** - пакет відкриває нове з'єднання, яке логічно пов'язане з уже встановленими, наприклад, відкриття каналу даних в пасивному режимі FTP.
- **INVALID** - ця ознака говорить про те, що пакет не може бути ідентифікований і тому не може мати певного статусу. Це може відбуватися з різних причин, наприклад при нестачі пам'яті або при отриманні ICMP-повідомлення про помилку, яке не відповідає будь-якому відомому з'єднанню.
- **UNTRACKED** - відстеження стану з'єднання для даного пакета було відключено. Зазвичай воно відключається за допомогою дії NOTRACK в таблиці raw.
- **DNAT** - показує, що до даного з'єднання застосована операція підміни адреси призначення.
- **SNAT** - показує, що до даного з'єднання застосована операція підміни адреси джерела.

Таблицю трасувальника можна знайти в файлі /proc/net/ip_conntrack. Тут міститься список всіх активних з’єднань. Якщо модуль ip_conntrack завантажений, то команда cat /proc/net/ip_conntrak повинна вивести інформацію про активні з’єднання.

**Побудови правил для iptables**.

Кожен рядок, який ви вставляєте в ту чи інший ланцюжок, повинні містити окреме правило. Кожне правило - це рядок, що містить в собі критерії які визначають, чи підпадає пакет під задане правило, і дію, яку потрібно зробити в разі виконання критерію. У загальному вигляді правила записуються приблизно так:

**iptables** [-t *table*] command [match] [target/jump]

*table*: FILTER, NAT, MANGLE (default FILTER)

*command*:

A - додати правило (в кінець);

D - видалити правило;

R – замінити;

I – вставити нове правило в ланцюг;

L – вивід списку правил в заданому ланцюгу;

F – скинути всі правила в заданому ланцюгу;

Z - скинути всі лічильники в заданому ланцюгу;

N – новий ланцюг;

X – видалити ланцюг;

P – задати політику для ланцюга;

E – задати нове ім'я для ланцюга;

*match* – критерії перевірки

*target/jump* – дія (DROP, REJECT, ACCEPT etc..)

Розділ **match** задає критерії перевірки, за якими визначається підпадає пакет під дію цього правила чи ні. Тут можна вказати найрізноманітніші критерії - IP-адреса джерела пакету або мережі, IP-адреса місця призначення, порт, протокол, мережевий інтерфейс і т.д.

**target** вказує, яка мета повинна бути виконано за умови виконання критеріїв в правилі. Тут можна змусити ядро передати пакет в інший ланцюжок правил, "скинути" пакет і забути про нього, видати на джерело повідомлення про помилку і т.п.

Групи критеріїв перевірки:

Перша - *загальні критерії* які можуть використовуватися в будь-яких правилах.

Друга -*TCP критерії* які застосовуються тільки до TCP пакетів.

Третя - *UDP критерії* які застосовуються тільки до UDP пакетів.

Четверта - *ICMP критерії* для роботи з ICMP пакетами.

П'ята - *спеціальні критерії*, такі як state, owner, limit та ін

**Дії та переходи.**

Дії і переходи повідомляють правилу, що необхідно виконати, якщо пакет відповідає заданому критерію. Найчастіше вживаються дії **ACCEPT** і **DROP**.

Опис переходів в правилах виглядає точно так само як і опис дій, тобто ставиться ключ -j і вказується назва ланцюжок правил, на який виконується перехід. На переходи накладається ряд обмежень, перше - ланцюжок, на який виконується перехід, повинен знаходитися в тій же таблиці, що і ланцюжок, з якої цей перехід виконується, друге - ланцюжок, що є метою переходу повинен бути створений до того як на нього буде виконуватися перехід . Наприклад, створимо ланцюжок tcp_packets в таблиці filter за допомогою команди

iptables -N tcp_packets

Тепер можна виконати перехід на цей ланцюжок наступним чином:

iptables -A INPUT -p tcp -j tcp_packets

**Основні файли конфігурації:**

/etc/sysconfig/iptables - файл конфігурації

Лог-файли

var/log/messages

/var/log/syslog
/var/log/iptables.log

3.3 Міжмережеві екран прикладного рівня. Проксі сервери.

Прикладні шлюзи, або міжмережеві екрани (МЕ) прикладного рівня, працюють як проксі-сервери протоколів прикладного рівня (HTTP, FTP, Telnet, тощо). Їхні функції – посередницькі. Такий МЕ містить в собі сервери прикладних протоколів. Крім можливості приховування внутрішньої структури захищеної мережі, такі МЕ дозволяють використовувати для розмежування доступу достатньо широкий спектр засобів автентифікації прикладного рівня, обмежуючи доступ на основі комбінації адрес, номерів портів, повноважень окремих користувачів, реального часу

Здійснення фільтрації на прикладному рівні означає, що МЕ переглядає всю інформацію всередині пакетів. Це надає можливості відфільтровувати окремі види команд або окремі типи інформації в прикладних протоколах. Наприклад, прикладний шлюз може забороняти використання команди PUT клієнтам FTP, або не пропускати вкладення заданих типів файлів в листах електронної пошти

Додаткові функції захисту прикладних шлюзів:

- ідентифікація й автентифікація користувачів при спробі встановити з’єднання через МЕ;
- перевірка справжності інформації, яку передають через МЕ;
- розмежування доступу до ресурсів мереж;
- фільтрація й перетворення потоку повідомлень;
- антивірусні й антиспамові перевірки;
- шифрування й дешифрування;
- аудит;
- реєстрація подій;
- реагування на події;
- аналіз зареєстрованої інформації;
- генерація звітів;
- кешування даних, що надходять із зовнішньої мережі.

**Переваги і недоліки прикладних шлюзів:**

Переваги:

- забезпечує найвищий рівень захисту локальної мережі завдяки реалізації функцій посередництва;
- захист на прикладному рівні дозволяє здійснювати перевірки, специфічні для окремих прикладних програм, що надає можливість нейтралізувати притаманні їм вразливості;
- в разі відмови прикладного шлюзу, трафік через нього буде повністю блоковано, і таким чином безпека локальної мережі порушена не буде, але,

звичайно, буде порушена доступність;

Недоліки

- значна складність самого МЕ, а також процедур його встановлення й конфігурування;
- значні вимоги до продуктивності та наявних ресурсів комп'ютерної платформи, на якій реалізовано МЕ;
- висока вартість;
- відсутність прозорості для користувачів;
- зниження перепускної здатності мережі при передачі трафіку через МЕ.

**Поняття та функції проксі-серверів.**

**Проксі-сервер** (від англ. proxy — «представник, уповноважений») — сервер (комп'ютерна система або програма) в комп'ютерних мережах, що дозволяє клієнтам виконувати непрямі (через посередництво проксі-сервера) запити до мережевих сервісів. Спочатку клієнт з'єднується з проксі-сервером і запитує який-небудь ресурс (наприклад, e-mail), розташований на іншому сервері. Потім проксі-сервер або підключається до вказаного сервера і отримує ресурс у нього, або повертає ресурс з власного кешу (у випадках, якщо проксі має свій кеш). У деяких випадках запит клієнта або відповідь сервера може бути змінена проксі-сервером з певною метою. Також проксі-сервер дозволяє захищати клієнтський комп'ютер від деяких мережевих атак і допомагає зберігати анонімність клієнта. Роботу проксі сервісу показано на (рис.3.12).

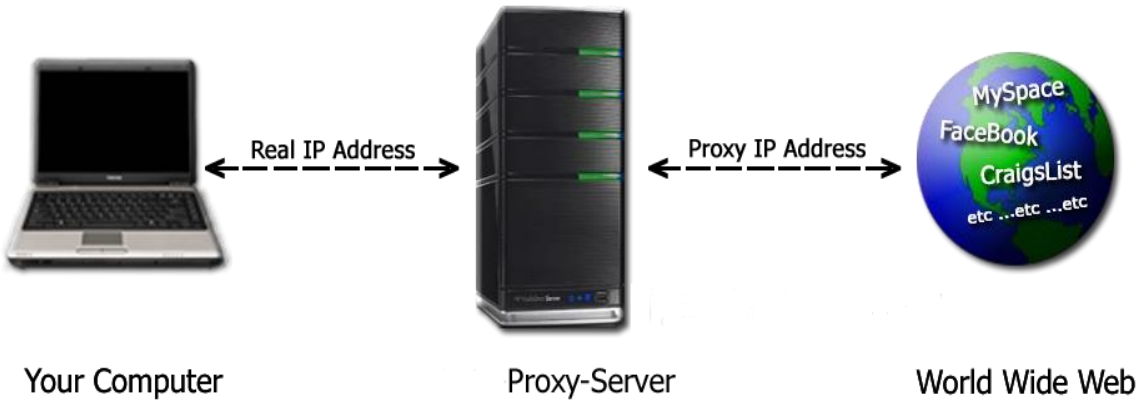



Рисунок 3.12 – Робота проксі-сервісу

**Найчастіше проксі-сервери застосовуються для наступних цілей:**

- **Забезпечення доступу з комп'ютерів локальної мережі в Інтернет**.
- **Кешування даних**: якщо часто відбуваються звернення до одних і тих ж зовнішніх ресурсів, то можна тримати їх копію на проксі-сервері і видавати по запиту, знижуючи тим самим навантаження на канал у зовнішню мережу і прискорюючи отримання клієнтом запитаної інформації.

- **Стиснення даних**: проксі-сервер завантажує інформацію з Інтернету і передає інформацію кінцевому користувачеві в стислому вигляді. Такі проксі-сервери використовуються в основному з метою економії зовнішнього трафіку.
- **Захист локальної мережі від зовнішнього доступу**: наприклад, можна налаштувати проксі-сервер так, що локальні комп'ютери будуть звертатися до зовнішніх ресурсів тільки через нього, а зовнішні комп'ютери не зможуть звертатися до локальних взагалі (вони побачать тільки проксі-сервер).
- **Обмеження доступу з локальної мережі до зовнішньої**: наприклад, можна заборонити доступ до певних веб-сайтів, обмежити використання Інтернету якимось локальним користувачам, встановлювати квоти на трафік або смугу пропускання, фільтрувати рекламу і віруси.
- **Анонімізація доступу до різних ресурсів**. Проксі-сервер може приховувати відомості про джерело запиту або користувача. У такому випадку цільовий сервер бачить лише інформацію про проксі-сервер, наприклад, IP-адресу, але не має можливості визначити дійсне джерело запиту. Існують також проксі-сервери, які передають цільовому серверу помилкову інформацію про справжнього користувача.
- **Багато проксі-серверів використовуються для кількох цілей одночасно**. Деякі проксі-сервери обмежують роботу декількома портами: 80 (HTTP), 443 (шифровані з'єднання HTTPS), 20,21 (FTP).

**Типи проксі-серверів:**

Проксі-сервер може працювати в наступних трьох основних режимах:

**Прозорий режим**

В цьому режимі (рис. 3.13) HTTP з'єднання здійснюване клієнтами перенаправляється на проксі-сервер без їх відома або явної конфігурації. У цьому режимі не потрібно налаштування клієнтів. Недоліки цього способу: необхідна конфігурація NAT і перенаправлення трафіку, аутентифікація клієнтів не працює, не перенаправляються FTP і HTTPS запити.

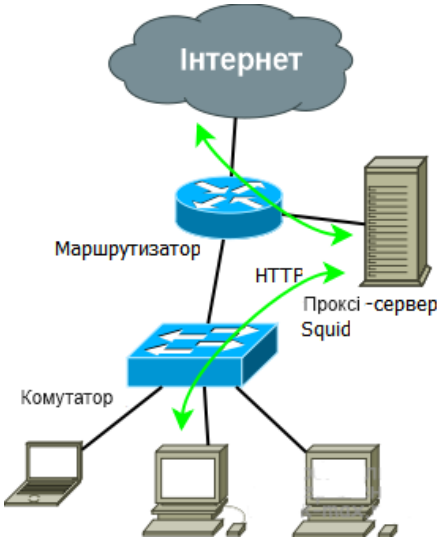



Рисунок 3.13 – Прозорий режим роботи проксі серверу

**Аутентифікуючий режим**

Для роботи в цьому режимі (рис. 3.14) клієнти повинні бути налаштовані для роботи з проксі-сервером (в налаштуваннях з'єднання повинна бути прописана адреса проксі-сервера). Може виконуватися аутентифікація і авторизація клієнтів через Kerberos, Ldap, NTLM, IP і Radius. Можливо побудова взаємодії з серверами Microsoft Active Directory шляхом аутентифікації клієнтів – членів домену, використовуючи протокол Kerberos, і подальшої авторизації членів груп домену використовуючи LDAP в прозорому режимі (користувач вводить свій пароль тільки при реєстрації в домені). Для авторизованих груп можливе застосування різних налаштувань контролю доступу та QoS (delay pools).

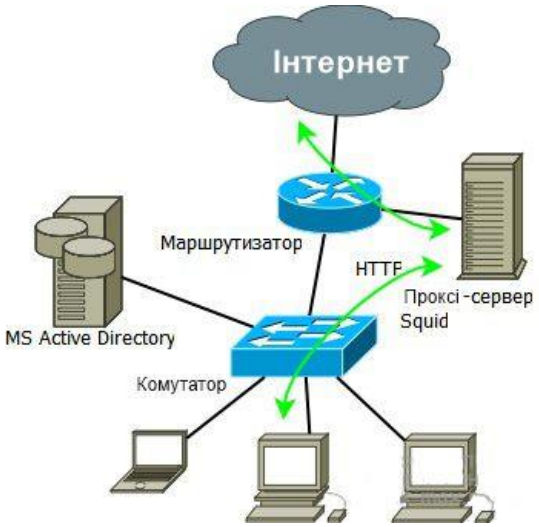



Рисунок 3.14 – Аутентифікуючий режим роботи проксі серверу

**Зворотний проксі-сервер** (рис. 3.15)

Проксі-сервер кешує вихідні дані. Зворотний проксі-сервер Squid отримує дані у HTTP сервера від імені клієнта і передає їх назад клієнту (наприклад, в Інтернет). Цей режим дозволяє здійснити:

- використання кешування, яке знижує навантаження на HTTP сервера;
- розподіл навантаження між HTTP серверами;
- маскування HTTP серверів і їх характеристик;
- запобігання веб атакам на сервери.

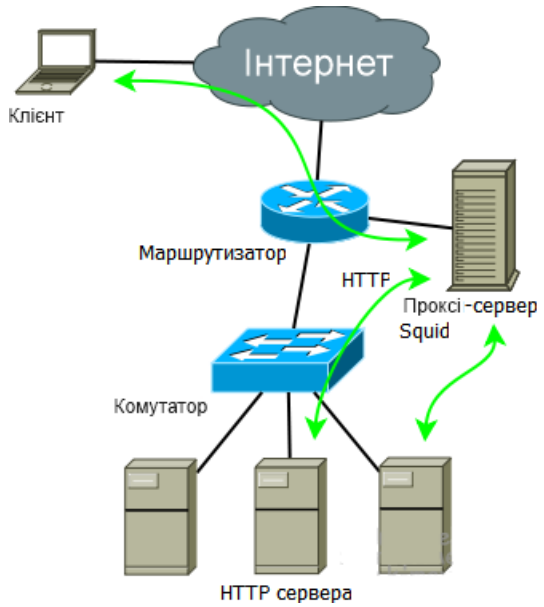


Рисунок 3.15 – Режим роботи зворотного проксі серверу

- **Транзитні проксі-сервери** (forward proxies).

Цей тип проксі серверу дозволяє користувачам із однієї зони безпеки виконувати запити до «наступної». З точки зору безпеки такий сервер приховує ім'я та адресу користувача що дає запит на ресурс. Він також може використовуватися для приховування інших атрибутів сеансу користувача. Типовим прикладом є корпоративні проксі-сервери, які обслуговують внутрішніх клієнтів шляхом дозволу їм на доступ до зовнішніх веб ресурсів.

- **Кешуючі проксі-сервери** (caching proxies)

Проксі-сервери, які сконфігуровані на повторне використання запитів та ресурсів, шляхом їх зберігання на сервері. Важливо правильно налаштувати даний тип сервера, щоб він кешував лише те що справді потрібно. Динамічний, регулярно обновлюваний контент не кращий вибір для кешування.

Серед зазначених типів проксі серверів існує кілька їх **видів**, кожен з яких має вузьку спеціалізацію, тобто підтримує роботу тільки з одним або декількома протоколами. Найпоширенішими на даний момент є http-, SOCKS- і NAT-проксі. Останні входять до стандартні компоненти сучасних операційних систем, таких як Linux і Windows. За своїми характеристиками програмні проксі-сервери NAT практично не відрізняються від апаратних (маршрутизаторів) і істотно поступаються в адмініструванні вузькоспеціалізованим проксі-серверів. Розглянемо найбільш популярні типи проксі-серверів:

HTTP проксі-сервер - один з найпоширеніших типів проксі-серверів в світі. Як видно з назви, він призначений для роботи з HTTP-протоколом і дозволяє працювати браузерам і іншим подібним програмам. Браузер запитує сторінку у проксі-сервера, який, в свою чергу, запитує її з власного кешу. Якщо сторінка не знайдена в кеші або має параметри, які забороняють її кешування, проксі-сервер

запитує її біля сусіднього проксі-сервера або безпосередньо звертається до сайту і після успішного завантаження сторінки видає її призначеному для користувача додатком. Оскільки далі буде розглядатися один з HTTP проксі-серверів - Squid, перерахуємо основні можливості цього типу проксі-серверів:

- дозволяють кешувати часто використовувані дані (веб-сторінки), завдяки чому скорочується зовнішній трафік і прискорюється завантаження сторінок кінцевим користувачем. Однак Інтернет стає все більш динамічним, і часто багато веб-сервери забороняють проксі-серверів кешувати дані або накладають обмеження на певні сторінки, тому приріст в економії трафіку не дуже істотний і може становити до 15%. Якщо питання про трафік стоїть гостро, то багато HTTP проксі-сервери підтримують ігнорування заголовків META в сторінках, тим самим дозволяючи кешувати динамічні дані;
- обмежують доступ не тільки до певних сайтів, але і до конкретних розділів сайту, за рахунок чого досягається велика гнучкість в адмініструванні користувачів. Крім того, доступ до сайтів певної групи можна дозволяти лише після додаткової аутентифікації;
- деякі HTTP проксі-сервери дозволяють обмежувати швидкість завантаження інформації, розставляючи пріоритети по типам файлів. Завдяки цьому можна блокувати високу швидкість завантаження, наприклад відеофайлів або музичних композицій;
- підтримують різання рекламних блоків (банерів) шляхом заміщення вихідної картинки або аплета своїм кодом, що скорочує додатковий трафік;
- можливо поділ конкретних сайтів на роботу через ті чи інші додаткові проксі-сервери або інтернет-канали, що дозволяє більш економно управляти трафіком і вибирати оптимальні канали зв'язку;
- одна з важливих особливостей HTTP проксі-сервера - можливість ведення докладної статистики по кожному користувачеві. Це дозволяє аналізувати використання трафіку окремими користувачами, а також виявляти найбільш часто застосовуються веб-сервери;
- дозволяють працювати не тільки з протоколом HTTP, але і з іншими подібними протоколом - FTP, а в разі необхідності - блокувати його.

HTTPS проксі-сервер - по суті справи, це точно такий же HTTP проксі-сервер, як і описаний вище. Основною відмінністю даного типу проксі-серверів є можливість шифрування переданих між клієнтом, проксі-сервером і кінцевим сервером даних, про що і повідомляє буква «s» в його назві. Проксі-сервер, що працює по протоколу HTTPS, лише організовує канал передачі між клієнтом і сервером і не дозволяє аналізувати проходить по ньому інформацію. Відповідно можливості HTTPS, в порівнянні з HTTP, істотно вужче. У той же час цей проксі-сервер може бути використаний в якості проксі-сервера для інших, відмінних від HTTPS протоколів - pop, smtp, imap і ряду інших. У будь-якому випадку проксі-сервер такого типу значно підвищує безпеку конфіденційної інформації, хоча і

має деякі недоліки. Зазвичай HTTPS проксі-сервер суміщають з HTTP проксі-сервером, що робить його більш універсальним і гнучким при налаштуванні клієнтів.

FTP проксі-сервер - це вид проксі-сервера призначений для роботи з протоколом FTP. Клієнт звертається до проксі-сервера, а він передає запит FTP-сервера. Існує два варіанти роботи клієнта FTP через проксі-сервер. Перший варіант, коли запити до FTP-сервера передаються через HTTP проксі-сервер, підтримує більшість HTTP проксі-серверів, оскільки він зручний для браузерів, але при цьому деякі функції роботи з FTP-сервером будуть недоступні. Другим варіантом якраз є робота через окремий FTP проксі-сервер, і в цьому випадку підтримуються всі опції цього протоколу - такого з'єднання вимагає більшість спеціалізованих FTP-клієнтів. Відзначимо, що проксі-сервери даного виду майже не застосовуються, оскільки звичайний HTTP проксі-сервер в більшості випадків задовольняє потреби звичайних користувачів, які лише завантажують інформацію з FTP-серверів, при цьому не використовуючи всі доступні FTP-протоколу функції.

SOCKS проксі-сервер - працює на основі спеціально розробленого протоколу SOCKS (скорочено від SOCKetS). Він дозволяє здійснювати аутентифікацію користувачів на серверній стороні, що підвищує гнучкість настройки подібних систем. Проксі-сервери SOCKS є універсальними і дозволяють користувачеві працювати через будь-який інший протокол з практично будь-яким видом сервісів в Інтернеті. Одна з особливостей проксі-серверів цього типу - можливість роботи від зовнішніх клієнтів з внутрішньомережевими серверами, розташованими за міжмережевими екранами. Такий підхід дозволяє широко використовувати цей вид проксі для забезпечення доступу клієнтів як з локальної мережі, так і в зворотному напрямку. Оскільки цей протокол є одним з найпопулярніших на даний момент, створені спеціальні програми, що надають можливість пропускати клієнтське програмне забезпечення в Інтернет через цей протокол навіть за відсутності підтримки його цим програмним забезпеченням.

Крім перерахованих вище основних видів проксі-серверів, які охоплюють більшість поширених протоколів передачі даних, існують вузькоспеціалізовані сервери, що забезпечують коректну роботу з тими чи іншими додатками. На додаток до всього вищесказаного відзначимо, що проксі-сервери, до яких може отримати доступ будь-який користувач Інтернету, називаються відкритими. Таких серверів досить багато по всьому світу, і орієнтовані вони в першу чергу на тих користувачів, які хочуть зберегти відвідування сторінок або серверів в таємниці від провайдера або від мережевого адміністратора. Однак найчастіше ці сервери можуть прослуховуватися, оскільки будь-який проксі-сервер (його трафік) без підтримки шифрування переданих даних може бути прослуханий на будь-якому з пристроїв, які знаходяться на шляху від клієнта до сервера.

Найпоширенішими прикладами проксі-серверів є:

- 3proxy (BSD, багатоплатформений);
- CoolProxy (proprietary, Windows);
- HandyCache (shareware, Windows) безкоштовний для домашнього використання;
- Kerio Control (proprietary, Windows, Linux);
- Microsoft Forefront Threat Management Gateway (proprietary, Windows);
- Nginx (веб-сервер, що має режим роботи як reverse proxy і часто для цього використовується);
- Squid (GPL, багатоплатформений);
- Traffic Inspector (proprietary, Windows);
- UserGate (proprietary, Windows);
- Інтернет Контроль Сервер (shareware, FreeBSD);
- TOR (shareware, багатоплатформений).

### 3.4 Проксі-сервер Squid

***Squid*** (кальмар) - кешуючий проксі - сервер для протоколів HTTP, FTP, HTTPS (при відповідних налаштуваннях). Ліцензія GNU GPL.

Однією з особливостей Squid є можливість працювати в режимі «зворотного проксі» («reverse proxy»), так само відомого як «прискорювач» («HTTP accelerator»). У цьому випадку замість кешування запитів декількох користувачів до безлічі сайтів, кешуються запити великої кількості користувачів до декількох сайтів. У цьому режимі прийнятий запит перевіряється на «динамічність» (потрібно кожен раз обробляти запит з нуля) і «вік» (актуальні ще дані). Якщо дані ще актуальні й не змінилися, то запит не передається серверу, а віддається з кешу Squid. Таким чином істотно знижується навантаження на сервери. «Зворотний проксі» здатний розподіляти запити між кількома серверами, балансуючи навантаження і/або забезпечуючи відмовостійкість, тобто фактично надає функціональність, аналогічну кластеру.

Squid підтримує роботу в режимі «прозорого проксі». В режимі прозорості не проксіруются FTP і HTTPS - запитів.

Проект **Squid** має свій офіційний веб-сайт, на якому можна отримати детальну інформацію про розробників, документацію, пожертвування, посилання на завантаження Squid та розділи підтримки: ***http://www.squid-cache.org***

Щодо конфігурації, то в Squid усі налаштування самого серверу на CentOS робляться в одному файлі, що знаходиться за шляхом: **/etc/squid/squid.conf**.

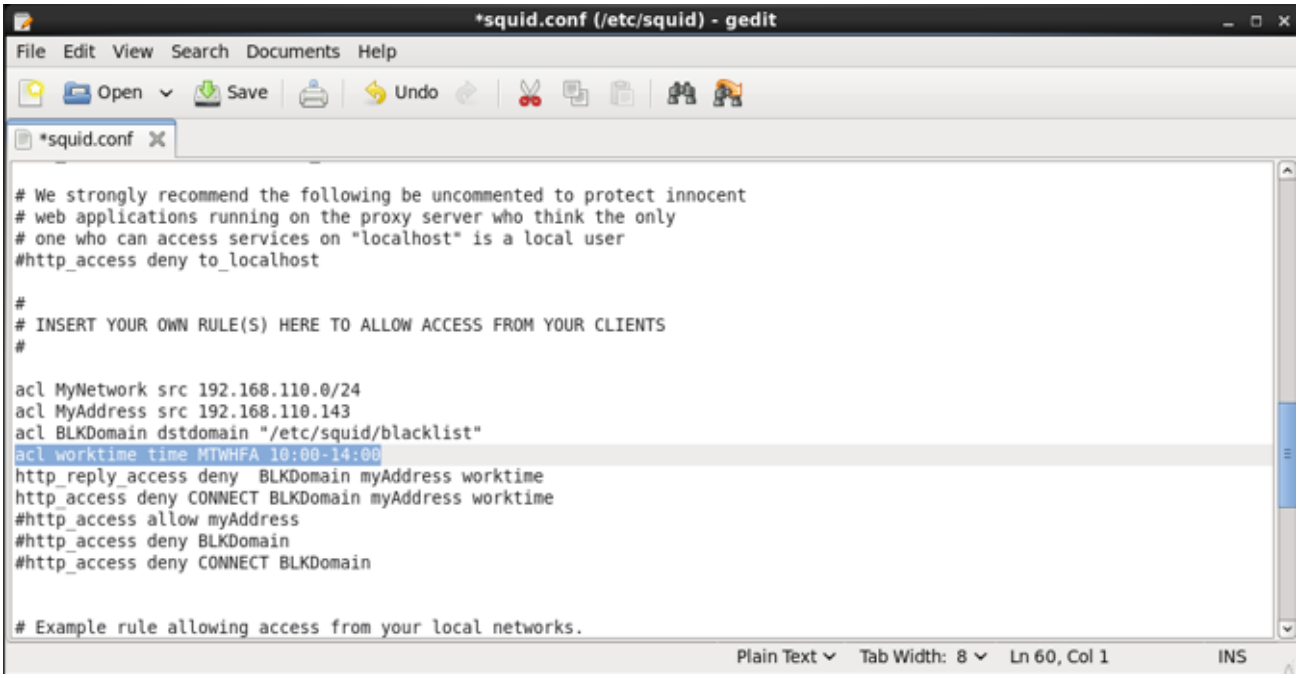



Рисунок 3.16 – Приклад налаштування конфігураційного файлу проксі серверу Squid

Структура файлу є дуже простою і складається з таких параметрів:
**http_port**,
**http_access**,
**acl** (списки контролю доступу).

**http_port** потрібен для того щоб проксі сервер Squid обслуговував тільки комп'ютери локальної мережі і бути невидимим для зовнішнього світу, щоб виключити можливість зловмисникам зовнішньої мережі скористатися внутрішнім каналом або трафіком.

**http_access** використовується для дозволу або заборони доступу до певних ресурсів, визначених адрес або до певних сайтів, за певними протоколами, портами і всього того, що безпосередньо вказано за допомогою **acl** (списків контролю доступу).

Система управління доступом в проксі сервері Squid є дуже гнучкою і великою. Вона складається з елементів зі значеннями і списків доступу з зазначенням **allow** (дозволу) або **deny** (забороною).

Формат запису acl наступний: **acl <ім'я> <критерій> <значення>**

Типи критеріїв:

- адреса (мережа) відправника запиту, цілі запиту (src ,dst);
- ім'я (доменне ім'я) відправника запиту, цілі запиту (srcdomain, dstdomain);
- частину URL запиту (url, url_regex);
- протокол (proto);

- порт (відправника, отримувача, самого Squid) (port);
- метод (PUT або GET) при передачі даних по HTTP;
- браузер користувача;
- авторизацію на проксі-сервері (auth);
- номер з'єднання;
- сертифікати користувача;
- параметри запиту;
- зовнішні обробники;
- час з'єднання (time).

**Приклад ACL**:

acl mynet src 192.168.1.0/24.

Ім'я – mynet, критерій – src (адреса мережі відправника), значення – 192.168.1.0/24 . Таким чином можна описати майже будь-які події, властивості користувачів, мереж, доменів та навіть додати залежність від часу.

Правила будуються наступним чином:

**http_access <allow/deny> <acl1,acl2 …,aclN>**.

allow — дозволити доступ, deny — заборонити доступ для з'єднань що підпадають під критерій заданий в списку доступу. Якщо в записі вказується декілька списків доступу, то воно виконується лише коли всі критерії співпадають, тобто умовно можна використати логічне «і» між цими критеріями. Такий прийом поширений при налаштуванні автентифікації, коли потрібно щоб виконувався критерій ідентифікації користувача і певні додаткові критерії. Це дуже потужний механізм, який дозволяє зробити досить гнучку систему доступу до інтернет ресурсів.

**Методи автентифікації в Squid**

Squid щодо автентифікації подібний Web-серверу Apache і підтримує ті ж способи автентифікації, що і Apache . На сьогоднішній час існують чотири основні типи автентифікації в HTTP протоколі:

1. **Basic** - найперша схема, найбільш часто використовувана
2. **NTLM** - це перша спроба Microsoft до SSO (Single-Sign-On) єдиного входу для ресурсів локальної мережі
3. **Digest** - вона ж дайджет аутентифікація. Трохи краща з боку захищеності ніж базова
4. **Negotiate (інша назва SPNEGO)** - вона ж покоління Kerberos SPN - друга спроба Microsoft реалізації SSO

**Загальні принципи організації аутентифікації в Squid**. Якщо бути точним, то сам Squid не виконує ніяких функцій автентифікації. Робота Squid по автентифікації полягає в декодуванні HTTP заголовку Authorization і передачі

декодованої інформації - так званому **хелперу**. Якщо надана інформація вірна, то доступ користувачеві надається, якщо ж не вірна або відсутня, то Squid повертає клієнту HTTP код 407. Тобто всю основну роботу по перевірці користувачів виконують сторонні модулі (вони ж – хелпери ). Розташування цих модулів залежить від дистрибутива і версії Squid, наприклад в CentOS їх розташування буде в **/usr/lib/squid**.

Коли Squid стартує, він запускає кілька підпроцесів, що виконують автентифікацію. Ці процеси відповідно до своєї схеми автентифікації перевіряють справжність користувача і повертають позитивний результат squid'у або негативний. Подібна техніка дозволяє використовувати велику кількість різних схем автентифікації.

Переглянути, які методи перевірки автентичності підтримує Ваш Squid, розміщення хелперів і багато інших параметрів, можна командою **squid3 -v**, висновок якої містить опції, з якими Squid був налаштований.

Взаємодія і опис налаштувань сторонніх модулів автентифікації проводиться за допомогою параметра **auth_param**. Клієнт буде по черзі намагатися використовувати схеми автентифікації в тому порядку, в якому вони задані в **squid.conf**.

Нова налаштована схема автентифікації вступить в силу тільки після перезапуску squid, при цьому, внесення змін у вже існуючі налаштовані схеми може проводиться без перезапуску демона (досить дати команду

**squid3 -k reconfigure або /etc/init.d/squid3 reload**).

Налаштований **зовнішній модуль перевірки автентичності** ще не означає, що ця перевірка буде працювати і пускати користувача в Інтернет. Для контролю доступу в інтернет, необхідно створити **acl**, який буде задіювати налаштований модуль автентифікації. Для цього, acl в поле тип_відбору повинен мати значення **proxy_auth, proxy_auth_regex** або **external** з використанням змінної % LOGIN. І, природно, для заданого acl необхідно мати дозвіл на доступ в параметрі http_access.

**Формат задання типу використовуваної автентифікації** (формат використання параметра auth_param) наступний:

**auth_param** *схема_аутентіфікациі параметри_схеми* [значення_параметрів]

де, *схема_аутентіфікациі* - одна зі схем (basic, digest, ...),

*параметри_схеми* - задають настройки зазначеної схеми і в різних схемах можуть відрізнятися,

значенія_параметров - в цьому полі описуються значення параметрів_схеми.

До кожної схеми автентифікації можуть використовуватися скільки завгодно різних зовнішніх модулів, аж до використання і самописних скриптів bash. Зазвичай при установці Squid встановлюються всі необхідні модулі згідно підтримуваних схем автентифікації.

**Basic**

Розглянемо найпростіший спосіб автентифікації, який існував ще з часів протоколу HTTP / 1.0. Basic метод автентифікації заснований на наданні клієнтом (веб-браузером або іншою програмою) - облікових даних (імені і пароля) при запиті будь-якої сторінки. При цьому, ім'я користувача, об'єднане двокрапкою з паролем кодується base64 і передається в http-запиті в поле Authorization. Наприклад ім'я користувача Artem і пароль 12345 у вигляді рядка Artem:12345 будуть передані в кодованому вигляді: TGdeuE3r5gOFdc3Q1. Кодування тексту в кодування base64 не є захистом. Фактично пароль передається у відкритому вигляді! Єдиною перевагою цієї схеми є те, що будь-який браузер, яким би він старим не був, підтримує даний вид автентифікації.

Для роботи **basic автентифікації в squid**, необхідно задати параметр, який вказує на описаний нами хелпер, який взаємодіє зі створеним нами файлом паролів. Крім того, необхідно описати acl, що містить список користувачів, які можуть отримати доступ до Інтернету (ці користувачі повинні бути присутніми в списку паролів), або замість користувачів вказати значення REQUIRED, яке «включить» в створений acl всіх користувачів, які пройшли автентифікацію. Ну і звичайно необхідно *дозволити доступ створеному acl за допомогою параметра http_access*.

У параметра **auth_param basic** є додаткові налаштування (наприклад, кількість одночасних з'єднань - auth_param basic concurrency n, час зберігання / кешування успішних авторизаций - auth_param basic credentialsttl 2 hours та ін.), З якими можна ознайомитися в squid.conf.

Крім зберігання паролів в файлі, при basic аутентифікації можна використовувати ще безліч джерел інформації про користувача / пароль, наприклад:

- LDAP - хелпер squid_ldap_auth
- PAM - хелпер pam_auth
- SASL - хелпер sasl_auth
- NIS - хелпер yp_auth
- MySQL - хелпер squid_db_auth
- RADIUS - хелпер squid_radius_auth
- та інші.

**Digest**

Цей метод більш захищений, ніж basic-аутентифікація, тому що використовує MD5-шифрування для відправки пароля через мережу. Схема роботи взаємодії аналогічна basic, за тим лише винятком, що на 2-му і наступних кроках у відповідь сервера і відповіді клієнта в поля автентифікації додаються деякі додаткові параметри і випадкові значення, які використовуються для

шифрування пароля. Сам пароль передається у вигляді хеша, який шифрується і дешифрується з використанням випадкового числа. Для створення файлу паролів використовується утиліта htdigest, яка знаходиться в пакеті apache2-utils. Використовується хелпер /usr/ lib/squid3/digest_pw_auth із зазначенням файлу паролів.

**NTLM**

NTLM протокол являється застарілим і майже не використовується. На сьогодні він являється небезпечним. Для перевірки справжності використовується хелпер /usr/lib/squid3/ntlm_auth. Даний вид автентифікації може працювати в старих версіях браузерів IE, Mozilla, Opera, Crome.

**Negotiate**

Negotiate метод працює на протоколі Kerberos. Автентифікація методом Negotiate реалізується на Windows Server. Перевіряти валідність користувача через Kerberos можна різними шляхами, наприклад можна використовувати SAMBA + Winbind, але це не є актуальним шляхом, тому дуже часто використовують інший спосіб - пакет krb5-user і файл krb5.conf. Пакет krb5-user та його залежності містить утиліти, розроблені Massachusetts Institute of Technology (MIT).

**Перенаправлення**

Squid має можливість перезаписувати запити URL. Squid можна налаштувати, щоб пропускати вхідні URL через процес «редіректора», який поверне новий URL, або порожній рядок, що означає відсутність змін. Редіректор – не є стандартною частиною пакета Squid. Редіректор дає адміністратору можливість контролювати переміщення користувачів.

SAMS (SQUID Account Management System) – програмний засіб для адміністрування доступу користувачів до проксі-серверу Squid.

На даний момент SAMS налаштовує роботу редіректорів:

- Редіректор SAMS - редіректор, працюючий напряму з базами SAMS.
- SquidGuard - потужний редіректор.
- Стандартний SQUID - простіший редіректор, описаний в документації SQUID.
- Редіректор SAMS.

Написаний спеціально для SAMS, напряму використовує інформацію, що міститься в базі даних. Дозволяє налаштувати різне перенаправлення запитів для різних користувачів. Він забезпечує:

- Обмеження доступу користувачів до Squid.
- Контроль часу доступу користувачів Squid.

- Обмеження доступу користувачів до заборонених ресурсів (або доступ тільки до дозволених).
- Перенаправлення запитів до банерів, лічильників.

Редіректор SquidGuard.

Потужний редіректор, з широким спектром можливостей. До складу редіректора входять списки «поганих» доменів (реклама, банери, порнографія).

Стандартний редіректор Squid

Цей редіректор описаний в документації. Написаний на perl. Створюється після подачі команди на реконфігурацію Squid, на основі списків перенаправлення запитів. Швидкий та легкий в використанні редіректор, але не розрізняє користувачів. При використанні цього редіректора, обмеження доступу реалізується списками доступу Squid.

**Обмеження максимальної швидкості з'єднання**

Обмеження максимальної швидкості з'єднання користувача в Squid реалізовано з допомогою механізму delay pool («пул затримки»). Механізм працює по принципу басейна в який втікає і витікає інформація (трафік). Окремі області пам'яті, які конфігуруються подібним чином, називаються bucket (укр. відро). У «відра» є параметри: ємність, швидкість наповнення. Якщо користувачі отримують інформацію на швидкості нижче, ніж швидкість наповнення, то відро завжди повне. Якщо користувач короткочасно підіймає швидкість більшу, то поки відро не стане пустим, він не обмежується по швидкості, а потім отримує обмеження. У разі наявності групових та індивідуальних «відер», вони включаються послідовно.

3.5 Системи виявлення атак.

Системи виявлення атак (СВА) – це програмні або програмно-апаратні системи, які автоматизують процес аналізу подій в інформаційно-комунікаційній системі з міркувань безпеки – англ. – Intrusion Detection Systems (IDS).

Крім виявлення атак (тих, що реально здійснюються, або тих, що є потенційно можливими), СВА здатні здійснювати реагування на атаки – від найпростіших звітів до активного втручання при визначенні проникнень. У наш час СВА вважаються необхідним елементом інфраструктури безпеки. Технологія виявлення атак базується на трьох складових:

- ознаках, що описують порушення політики безпеки;
- джерелах, у яких шукають ознаки порушень політики безпеки;
- методах аналізу інформації, яку одержують з різни джерел.

**Архітектура СВА:**

- сенсорна підсистема - збір подій, пов'язаних з безпекою системи, що захищається;

- підсистема аналізу - виявлення атак і підозрілих дій на основі даних сенсорів;
- сховище – забезпечує накопичення первинних подій і результатів аналізу;
- консоль управління - конфігурація СВА, спостерігання за станом системи, що захищається, і СВА, переглядання виявлених підсистемою аналізу інцидентів.

Для успішного функціонування СВА в мережі повинні бути створені такі канали зв'язку:

- канали, за допомогою яких СВА здійснює моніторинг хостів і мереж;
- канали між сенсорами і керуючим сервером, по яких передаються дані про виниклі події;
- канали, за якими виконуються відповідні дії СВА.

Можливий способи управління СВА:

- централізоване управління і прийняття рішення керування усім моніторингом, виявленням й звітністю здійснюється безпосередньо з єдиного «поста»);
- частково розподілене управління (моніторинг і визначення керуються з локального вузла, а ієрархічна звітність спрямовується в одне або більше центральних розташувань);
- повністю розподілене управління (моніторинг і визначення виконуються з використанням підходу, заснованому на агентах, коли рішення про відповідь приймаються в точках аналізу).

**Можливості СВА**

*Можливість блокування джерела атаки, перешкоджаючи її здійсненню.*

Найбільш ефективним є блокування атак у випадках, коли у системі є вже відомі, але ще не виправлені вразливості. Причини такої ситуації:

- у деяких системах не можуть бути виконані всі необхідні відновлення й модифікації;
- адміністратори іноді не мають досить часу або ресурсів для відстеження й встановлення всіх необхідних відновлень;
- користувачам можуть бути необхідні функціональність мережних сервісів і протоколів, які мають відомі уразливості;
- як користувачі, так і адміністратори роблять помилки при конфігуруванні й використанні систем.

*Можливість здійснювати реакцію на атаку, якщо системою зафіксований факт атаки і її джерело*. Це дозволяє змусити атакуючого відповідати за власну діяльність

*Можливість визначення преамбул атак*, які здебільшого є зондуванням мережі або деяким іншим тестуванням для виявлення вразливостей. За наявності СВА сканування буде виявленим, і доступ зловмисника до системи може бути

заблокованим. Навіть наявність простої реакції на зондування мережі означає підвищений рівень ризику для атакуючого і може змусити його відмовитись від подальших спроб проникнення в мережу

*Виконання документування існуючих загроз для мережі й систем.* Документована інформація про атаки може бути корисною при складанні звітів служби захисту інформації. Розуміння частоти й характеру атак дозволяє вжити адекватних заходів безпеки

*Забезпечення контролю якості розробки й адміністрування безпеки, особливо у великих і складних ІТС*. Функціонування СВА протягом тривалого часу надає інформацію про типові способи використання системи. Це може дозволити виявити вади у здійсненні керування безпекою, і виправити процедури керування.

*Одержання корисної інформації про проникнення*, що мали місце, з наданням поліпшеної діагностики для виявлення й коригування факторів, що сприяли компрометації системи. Навіть коли СВА не має можливості блокувати атаку, вона може зібрати про неї детальну і достовірну інформацію, яка може бути покладеною в основу відповідних правових і адміністративних заходів

Можливість визначити розташування джерела атак по відношенню до локальної мережі (зовнішні або внутрішні атаки). Це важливо при прийнятті рішень про розташування ресурсів у мережі

**Класифікації СВА за різними характеристиками.**

Відповідно **етапу фіксації здійснення атаки**:

- Типовим для СВА є *виявлення атак у реальному часі*, тобто в момент їх здійснення. Такі СВА є класичними.
- Існують системи, які здійснюють аналіз журналів реєстрації, і таким чином *виявляють атаки, які були здійснені раніше*.
- Іноді до СВА відносять засоби, які здійснюють *аналіз системи і попереджають про потенційну можливість здійснення атаки*. До таких засобів відносяться сканери вразливостей

Відповідно **до інформаційних джерел**:

- СВА рівня мережі (network-based IDS);
- СВА рівня вузлів (host-based IDS) - СВА рівня ОС, СВА рівня СКБД, СВА рівня прикладних програм

Відповідно до **методу аналізу**:

- виявлення зловживань (misuse detection);
- виявлення аномалій (anomaly detection).

Відповідно **до швидкості реакції** (затримка в часі між одержанням інформації із джерела та її аналізом і реакцією на неї):

- СВА пакетного режиму (interval-based IDS). Багато ранніх СВА рівня вузлів використовували таку схему роботи, тому що вони цілком залежали від

накопичення записів аудита в ОС. СВА пакетного режиму не виконують ніяких активних дій у відповідь на виявлені атаки.

- СВА реального часу (real-time IDS). СВА реального часу обробляють безперервний потік інформації від джерел. Це є переважною схемою роботи СВА рівня мережі, які одержують інформацію з потоку мережного трафіку. Виявлення проникнення, що здійснюється СВА реального часу, приводить до результатів досить швидко, і це дозволяє СВА виконувати певні дії у відповідь в автоматичному режимі.

Відповідно **до характеру відповіді** (набір дій, які система виконує після виявлення проникнень):

- Активні заходи. Автоматичне втручання в деяку іншу систему (наприклад, керування комутатором або мережним екраном)
- Пасивні заходи. Звіт СВА, зроблений для людей, які потім виконують деякі дії на основі даного звіту

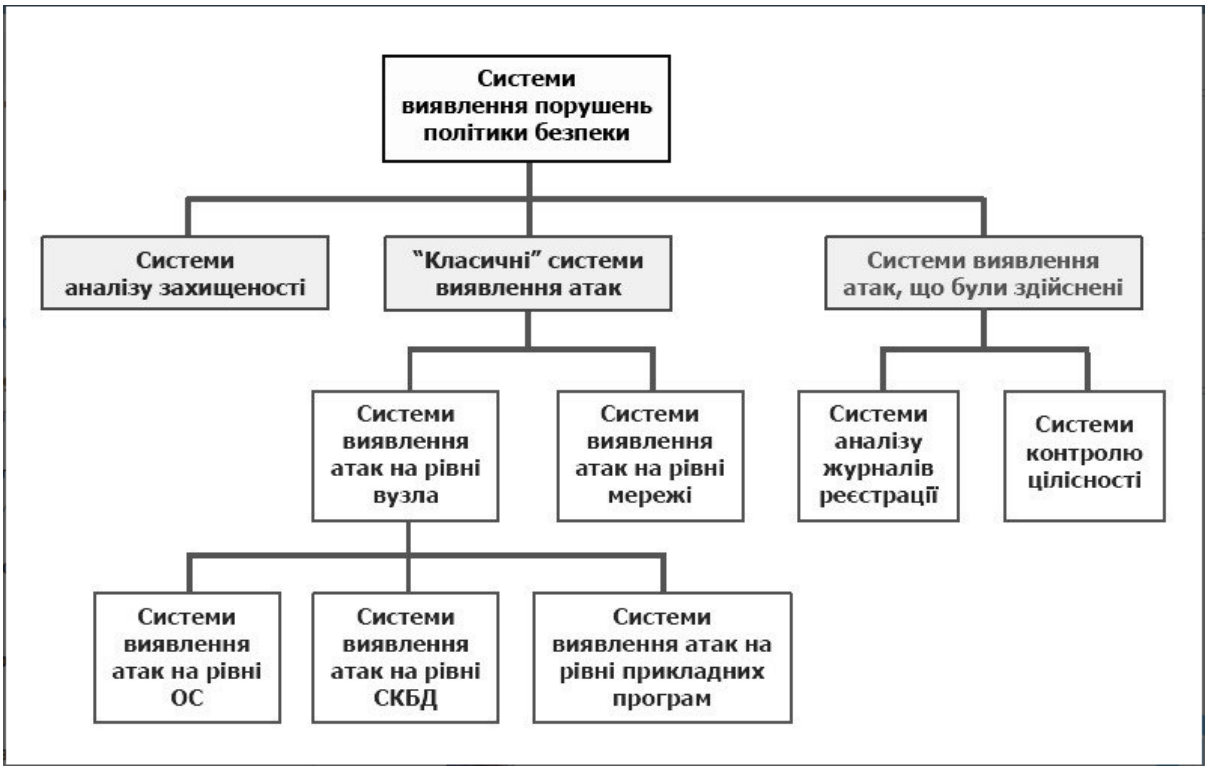



Рисунок 3.17 – Узагальнена класифікація СВА

**СВА рівня мережі.**

СВА рівня мережі визначають атаки, захоплюючи й аналізуючи мережні пакети. У сучасних системах часто використовується множина сенсорів, розташованих у різних точках мережі. Ці пристрої переглядають мережевий трафік, виконуючи його локальний аналіз та створюючи звіти про атаки для центральної керуючої консолі. СВА рівня мережі є найпоширенішими. До них

належить переважна більшість комерційних СВА

Переваги СВА рівня мережі:

- декілька оптимально розташованих СВА можуть переглядати велику мережу;
- розгортання СВА рівня мережі не впливає сильно на продуктивність мережі. Сенсори, як правило, є пасивними пристроями, які прослуховують мережний канал без впливу на нормальне функціонування мережі;
- СВА рівня мережі можуть бути зроблені практично невразливими до атак або навіть абсолютно невидимими для атакуючих.

Недоліки СВА рівня мережі:

- СВА рівня мережі, захоплюючи трафік, не гальмує роботу самої мережі, але може не встигати обробляти всі пакети у великій або зайнятій мережі.
- При підвищеному навантаженні в мережі СВА може пропустити атаку, не виявивши її. Деякі виробники намагаються вирішити дану проблему, повністю реалізуючи СВА апаратно.
- Необхідність швидко аналізувати пакети може призвести до того, що розробники СВА будуть обмежувати її можливості визначенням невеликої кількості атак або ж використовувати як можна менші обчислювальні ресурси, що знижує ефективність виявлення.
- У сучасних мережних технологіях намагаються уникати спільного середовища передачі даних, що ускладнює підключення сенсорів СВА рівня мережі. Для того, щоби СВА могла переглядати мережний трафік від декількох хостів, необхідне її підключення до маршрутизатора (комутатора) та таке налаштовування мережного обладнання, при якому на порт, що пов’язаний з СВА, буде потрапляти (дублюватись) увесь трафік сегмента. Це створює проблему через необхідність дуже великої перепускної спроможності цього порту та самої СВА.
- СВА рівня мережі не можуть аналізувати зашифровану інформацію. Ця проблема дуже актуальна при використанні VPN.
- Більшість СВА рівня мережі не здатні зробити висновок про те, чи була атака успішною, вони можуть тільки визначити, що атака була розпочата. Після того як СВА виявить атаку, адміністратор повинен вручну досліджувати кожний атакований хост для визначення, чи відбулося реальне проникнення.

**СВА рівня вузла.**

СВА рівня вузла мають справу з інформацією, зібраною усередині єдиного комп’ютера. Вони зазвичай використовують в якості інформаційних джерел журнали реєстрації подій, що створюються ОС та прикладними програмами. Деякі СВА рівня вузла розроблені для підтримки централізованої інфраструктури керування й одержання звітів СВА, що може допускати єдину консоль керування для відстеження багатьох хостів

Переваги СВА рівня вузла:

- СВА рівня вузла, з їхньою можливістю стежити за подіями локально по відношенню до хосту, можуть визначити атаки, які не можуть виявити СВА рівня мережі.
- На відміну від СВА рівня мережі, СВА рівня вузла можуть «бачити» наслідки початої атаки, тому що вони можуть мати безпосередній доступ до системної інформації, файлів даних і системних процесів, які є ціллю атаки
- СВА рівня вузла здатні аналізувати діяльність із великою вірогідністю й точністю, визначаючи тільки ті процеси й користувачів, які мають відношення до конкретної атаки.
- СВА рівня вузла часто можуть функціонувати в оточенні, у якому мережевий трафік зашифрований, коли джерела інформації рівня вузла створюються до того, як дані шифруються, і/або після того, як дані розшифровуються на хості призначення.
- На функціонування СВА рівня вузла не впливає топологія мережі.
- Коли СВА рівня вузла працюють із результатами аудита ОС, вони можуть надати допомогу у виявленні «троянських коней» або інших атак, які порушують цілісність ПЗ.

Недоліки СВА рівня вузла

- СВА рівня вузла більш складні у керуванні, тому що інформація повинна бути сконфігурована й повинна управлятися для кожного хоста, що переглядається.
- Джерела інформації (а іноді й частина засобів аналізу) для СВА рівня вузла розташовані на тому ж хості, що є метою атаки, тому, як складова частина атаки, СВА може бути атакована й відключена.
- СВА рівня вузла не завжди можуть визначити сканування мережі, або інші впливи, метою яких є вся мережа, тому що СВА спостерігає тільки за мережними пакетами, одержуваними конкретним хостом.
- СВА рівня вузла можуть бути блоковані деякими DoS-атаками.
- Коли СВА рівня вузла використовує результати аудита ОС як джерело інформації, об’єм інформації може бути величезним, що потребує додаткових ресурсів для зберігання в системі.
- СВА рівня вузла використовують обчислювальні ресурси хостів, за якими вони спостерігають, що впливає на продуктивність спостережуваної системи.

**СВА рівня прикладних програм.**

СВА рівня прикладних програм (application-based IDS) є специфічною підмножиною СВА рівня вузла, які аналізують події, пов’язані з конкретним прикладним ПЗ. Найбільш типовими джерелами інформації, що використовуються такими СВА, є журнали реєстрації транзакцій прикладного

ПЗ. Здатність взаємодіяти безпосередньо з прикладною програмою, з конкретним доменом, або використовувати знання, специфічні для певної прикладної програми, дозволяє СВА рівня прикладних програм визначати підозріле поводження авторизованих користувачів, що перевищує їхні повноваження. Такі порушення можуть виявитися тільки при аналізі взаємодії користувача з прикладною програмою.

Переваги СВА рівня прикладних програм:

- СВА рівня прикладних програм можуть аналізувати взаємодію між користувачем і програмою, що часто дозволяє відстежити неавторизовану діяльність конкретного користувача.
- СВА рівня прикладних програм, як правило, можуть працювати в зашифрованих оточеннях, тому що вони отримують інформацію у кінцевій точці інформаційного обміну, де інформація представлена вже в незашифрованому вигляді.

Недоліки СВА рівня прикладних програм:

- СВА рівня прикладних програм можуть бути більш уразливі, ніж СВА рівня ОС, до атак на записи реєстрації подій. Тому що журнали реєстрації прикладного ПЗ можуть бути захищені менш надійно, ніж результати аудита ОС.
- СВА рівня прикладних програм часто переглядають події на користувальницькому рівні абстракції, на якому у більшості випадків неможливо визначити «троянських коней» або інші подібні атаки, пов’язані з порушенням цілісності ПЗ.
- Для компенсації зазначених недоліків СВА рівня прикладних програм доцільно використовувати у комбінації з СВА рівня ОС та/або рівня мережі.

**Пасивні та активні СВА**

**Пасивна** СВА (Система виявлення вторгнень, IDS – Intrusion Detection System) при виявленні порушень безпеки записує інформацію про це порушення в додаток, а також надсилає інформацію про порушення на консоль та/або адміністратору системи за певним каналом зв'язку.

**Активна** СВА (Система Запобігання Вторгнень, IPS – Intrusion Prevention System) веде відповідні дії на порушення, скидаючи з'єднання або перенастроюючи міжмережевий екран для блокування трафіку від зловмисника. Відповідні дії можуть проводитися автоматично або по команді оператора.

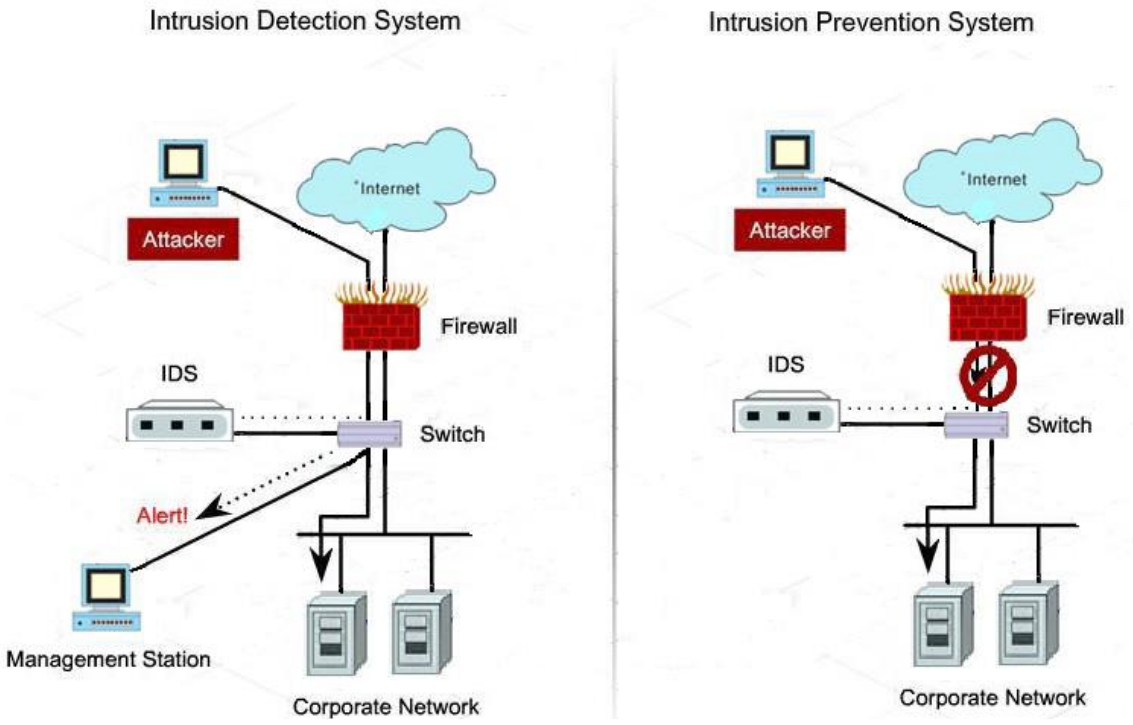



Рисунок 3.18 – Дія пасивної (IDS) та активної (IPS) СВА

**СВА і Брандмауер.**

Міжмережевий екран відрізняється від СВА тим, що обмежує надходження на хост або у підмережу певних видів трафіку для запобігання вторгнень і не відстежує вторгнення, що відбуваються всередині мережі. СВА, навпаки, пропускає трафік, аналізуючи його і сигналізуючи при виявленні підозрілої активності. Виявлення порушення безпеки проводиться зазвичай з використанням евристичних правил і аналізу сигнатур відомих комп'ютерних атак.

**Можливі варіанти розташування сенсорів СВА рівня мережі** (рис. 3.19).

**Позаду зовнішнього брандмауера в DMZ-мережі** (розташування 1).

Переваги:

- Бачить атаки, які виходять із зовнішнього світу, яким вдалося подолати першу лінію оборони мережевого периметра.
- Може аналізувати проблеми, які пов'язані з політикою або продуктивністю брандмауера, забезпечує першу лінію оборони.
- Бачить атаки, цілями яких є прикладні сервери (такі, як веб або ftp), що зазвичай розташовані в DMZ.
- Навіть якщо вхідна атака не розпізнається, СВА іноді може розпізнати вихідний трафік, який виникає в результаті компрометації сервера.

**Перед зовнішнім фаєрволом** (розташування 2).

Переваги:

- Документує кількість атак, що виходять з інтернету, метою яких є мережа.
- Документує типи атак, що виходять з інтернету, метою яких є мережа.

**На основний магістральної мережі** (розташування 3).

Переваги:

- Переглядає основний мережевий трафік, тим самим збільшується ймовірність розпізнавання атак.
- Визначає неавторизовану діяльність авторизованих користувачів всередині периметра безпеки організації.

**У критичних підмережах** (розташування 4).

Переваги:

- Визначає атаки, метою яких є критичні системи і ресурси.

Дозволяє фокусуватися на обмежених ресурсах найбільш значущих інформаційних цінностей, розташованих в мережі.

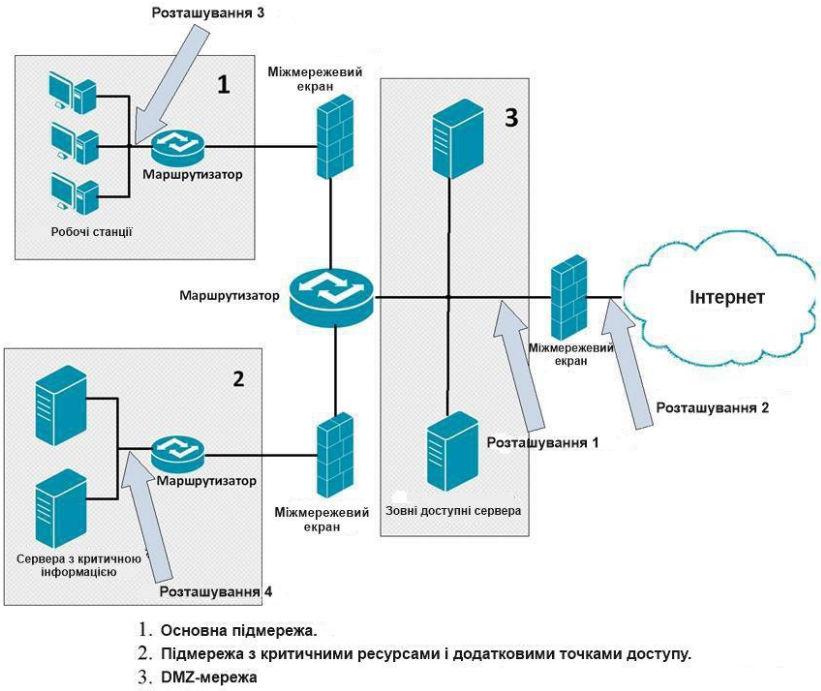



Рисунок 3.19 – Можливі варіанти розташування сенсорів СВА рівня мережі

3.6 IDS/IPS Suricata.

**Suricata -** це система виявлення та попередження мережевих атак з відкритим вихідним кодом, розробником якої є Open Information Security Foundation (OISF). Suricata здатна в режимі реального часу виявляти і запобігати вторгненням, здійснювати моніторинг мережевої безпеки, перевіряти мережевий

трафік, використовуючи спеціальні правила і сигнатури мови.

**Suricata** представляє собою комплексний засіб захисту від шкідливих програм, відомих загроз та правил, встановлених користувачем. Дана IDS/IPS виявляє та перевіряє аномалії в трафіку, здатна використовувати спеціалізовані набори правил. IDS/IPS Suricata може записувати в журнал логів HTTP запити, зберігати TLS сертифікати, витягувати файли з потоків і зберігати їх на диск. Також Suricata реєструє всі з'єднання HTTP на будь-який порт в файл для подальшого аналізу. Повна підтримка захоплення PCAP дозволяє легко аналізувати перехоплену інформацію, а завдяки Suricata TLS Parser можна реєструвати всі ключові обміни. Головна особливість Suricata - те, що, крім своїх унікальних напрацювань, система використовує практично всі напрацювання для Snort. Саме тому підходять всі набори правил Snort: Sourcefire VRT, OpenSource Emerging Threats (ETOpen) і комерційні Emerging Threats Pro.

**Особливості:**

- підтримка прискорення роботи через задіяння обчислень на стороні GPU;
- підтримує багатопоточність;
- підтримка автоматичного визначення протоколів;
- наявний декодувальник для протоколів HTTP, SSL/TLS, SMB, SMB2, DCERPC, SMTP, FTP і SSH;
- доступний модуль для ведення докладного журналу транзитних HTTP пересилань; підтримується витяг і перевірка переданих за протоколом HTTP файлів;
- підтримка розбору стисненого контенту;
- можливість ідентифікації за URI, Cookie, заголовкам, user-agent, тілу запиту/відповіді;
- підтримка різних інтерфейсів для перехоплення трафіку;
- висока продуктивність;
- механізм зіставлення по масці з великими наборами IP-адрес;
- підтримка виділення контенту за маскою і регулярними виразами;
- виділення файлів з трафіку, в тому числі їхня ідентифікація за іменем, типом або контрольною сумою MD5;
- можливість використання змінних в правилах;
- повна підтримка IPv6;
- підтримка протоколів тунелювання;
- підтримка декодування пакетів: IPv4, IPv6, TCP, UDP, SCTP, ICMPv4, ICMPv6, GRE, Ethernet, PPP, PPPoE, Raw, SLL, VLAN;
- режим ведення логу ключів і сертифікатів, які фігурують в рамках з'єднань TLS/SSL.

У Suricata використовується два режими IPS:

- NFQ;
- AF_PACKET.

NFQ IPS режим працює наступним чином:

1) Пакет потрапляє в Iptables.

2) Правило Iptables направляє його в чергу NFQUEUE за правилом «iptables -I INPUT -p tcp -j NFQUEUE».

3) З черги NFQUEUE пакети можуть оброблятися на рівні користувача, що і реалізує Suricata.

4) Suricata проганяє пакети по налаштованим правилам і в залежності від них може обрати одну із наступних дій: NF_ACCEPT, NF_DROP й NF_REPEAT

5) Пакети, які потрапляють в NF_REPEAT, можуть бути позначені в системі, і направлені знову на початок поточної таблиці Iptables.

**Структура конфігураційних файлів IDS/IPS Suricata**.

Файли конфігурацій та правил зазвичай розташовані за адресою /etc/suricata та /etc/suricata/rules.

Файл suricata.yaml розділений коментарями на розділи (кроки), які допомагають зорієнтуватися користувачу. При налаштуванні необхідно пройти такі кроки, як:

- проінформувати Suricata про мережу (вказати в якій мережі буде функціонувати IDS/IPS Suricata);
- вибрати правила, включити або вимкнути їх (вказати місце розташування правил, та зазначити правила, які будуть використовуватися. Правила, які тимчасово не будуть використовуватися, можна закоментувати використовуючи спеціальний символ #.);
- вибрати які виводи будуть використовуватися (визначаємо директорію, в яку будуть зберігатись лог-файли, інтервал оновлення та визначаються типи виводу відфільтрованої інформації);
- налаштувати загальні параметри запису подій (необхідно здійснити налаштування конфігурацій запису проаналізованих мережевих даних. Вибрати інтерфейс та за необхідності налаштувати інші конфігурації);
- налаштувати конфігурацію протоколів рівня додатків (відбувається конфігурація протоколів рівня додатків. Основним параметром конфігурації є включення, виключення його підтримки. Можна налаштувати такі протоколи, як HTTP, SSL, TLS, SMB, SMB2, DCERPC, SMTP, FTP, SSH, DNS, Modbus, ENIP/CIP, DNP3).

Загалом виділяють такі лог-файли:

- eve log (дозволяє сформувати висновок подій в форматі JSON для попереджень та моніторингу HTTP, DNS, SSH, TLS та інших протоколів.

Підтримка JSON істотно спрощує інтеграцію Suricata зі сторонніми додатками, такими як системи моніторингу);

- fast log (відповідає за моніторинг подій в режимі реального часу);
- stats log (відображає загальну статистику по подіям, перехопленим пакетам та по протоколам);
- suricata log (містить всю інформацію про активність системи IDS/IPS, її повідомлення та помилки).

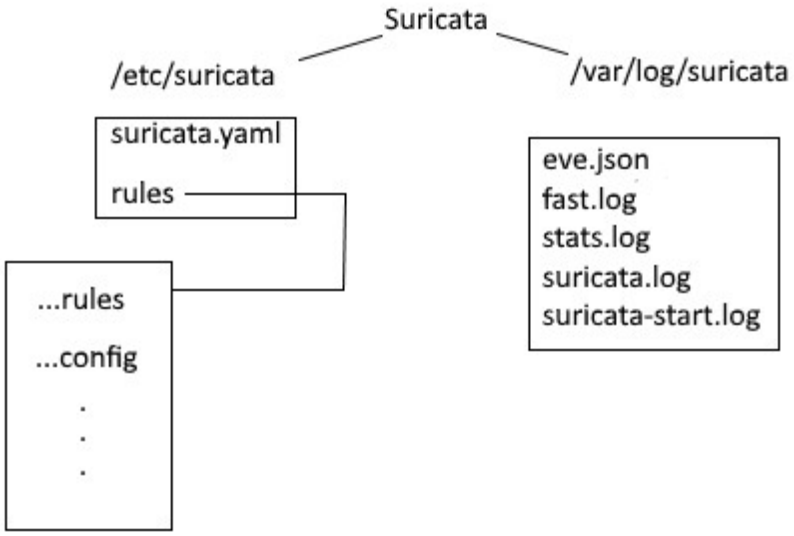



Рисунок 3.19 – Структура конфігураційних файлів IDS/IPS Suricata

**Структура правил IDS/IPS Suricata.**

Suricata використовує просту мову опису правил з широкими можливостями і достатньою гнучкістю. Нижче приведемо приклади, які допоможуть розібратися з правилами Suricata.

Більшість правил Suricata записується в один рядок. У колишніх версіях програми запис правила в один рядок був обов'язковим, але в сучасних варіантах програми допускається запис одного правила в кілька рядків, якщо всі рядки, за винятком останнього, завершуються символом \.

Кожне правило Suricata ділиться на дві частини - заголовки і опції. Заголовок правила включає дію, протокол, IP-адресу і маски для відправника і одержувача, а також номери портів відправника і одержувача. Опції правила включають повідомлення і інформацію про те, які частини пакета слід перевірити, щоб визначити чи слід застосовувати по відношенню до пакетів задану дію.

Нижче показаний приклад простого правила Suricata:

alert tcp any any -> 192.168.1.0/24 (content: «|00 01 86 a5|»; msg: «mountd access»;)

Опції правила вказуються в круглих дужках, а слова, за якими слідує

двокрапка, являються ключовими.

При дотриманні всіх зазначених у правилі умов по відношенню до пакету виконується задана дія (генерація сигналу в наведеному прикладі).

Заголовок правила в загальному випадку має вигляд:

<Дію> <протокол> <відправник> <порт> <напрямок> <одержувач> <порт> (опції)

Правило містить ключове слово, яке визначає **дію**, що виконується при збігу всіх заданих правилом умов. Дія завжди вказується першою і може бути одним з ключових слів. Основними ключовими словами є:

- alert - генерувати сигнал з використанням обраного методу і записати інформацію про пакети в журнальний файл;
- log - записати інформацію про пакети в журнальний файл;
- pass - пропустити (ігнорувати) пакет;
- activate - генерувати сигнал і активізувати інше динамічне правило;
- dynamic - правило не виконує ніяких дій до його активації за допомогою дії activate в іншому правилі, а при активації діє як log.

Можна також створювати власні типи дій, які називають типами правил – ruletype, і пов'язувати з ними модулі (plugin), використовуючи в подальшому такі правила як дії Suricata. У наведеному прикладі створюється тип правила для запису пакетів.

```
ruletype suspicious
{
type log
output log_tcpdump: suspicious.log
}
```

Наступне поле заголовка вказує **протокол**. Suricata в даний час розпізнає 4 протоколи - tcp, udp. icmp і ip.

Після протоколу в правилі зазначають **адреси і номера портів** для даного правила. Ключовим словом **any** будуть відповідати всі IP-адреси (0.0.0.0/0). Для запису адрес можна використовувати блоки CIDR, що описані в RFC1518. Блок CIDR показує префікс мережі і розмір маски, яка буде застосовуватися правилом до адрес у всіх пакетах для перевірки відповідності вказаному префіксу. Блок CIDR / 24 вказує мережу класу C, / 16 - класу B, а / 32 вказує адресу окремого хоста. Наприклад, запис 192.168. 1.0 / 24 буде відповідати адресами в діапазоні 192.168.1.0 - 192.168.1.255. Нотація CIDR забезпечує компактний і зручний спосіб представлення безперервних блоків мережевих адрес.

Наведеним нижче правилом будуть відповідати пакети, відправлені з будь якої (any) адреси в мережу класу C 192.168.1.0

alert tcp any any -> 192.168.1.0/24 111 (content: «|00 01 86 a5|»; msg: «mountd access»;)

Стосовно до адрес і блоків можна використовувати оператор заперечення !. При використанні цього оператора правилом будуть відповідати пакети, які не потрапляють в зазначений діапазон адрес.

Наступним елементом заголовку є номера **портів**. Їх можна задавати у вигляді конкретного значення, діапазону, списку або ключового слова any (будь-який порт). Для портів також підтримується оператор заперечення. Для завдання діапазону вказуються верхній і нижній межі, розділені двокрапкою (:). Граничні значення включаються в діапазон. Правилу

log udp any any -> 192.168.1.0/24 1:1024 log udp

будуть відповідати всі пакети UDP, адресовані в порти з 1 по 1024 хостів мережі класу C 192.168.1.0. Якщо одна з меж діапазону не задана замість неї використовується мінімальний (0) або максимальний (65535) номер порту.

Наступним елементом в заголовку є **оператор напрямку** ->, який показує напрямок передачі трафіку для даного правила. Адреси та порт зліва від цього оператора відносяться до відправника, а праворуч - до одержувача пакетів. Можна також створювати «двонаправлені» правила за допомогою оператора <>. У цьому випадку кожна з пар «адреса-порт» буде трактуватися програмою Suricata як відправник і одержувач. Такі правила зручні для аналізу пакетів в сеансових з'єднаннях (наприклад POP3). Використання оператору <- недопустимо в правилах Suricata.

Найважливішою частиною роботи Suricata є опції правил. Для розділення опцій в правилах Suricata використовується крапка з комою (;). Ключові слова опції відрізняються від аргументів двокрапкою (:).

Існує 4 основних категорії опцій правил:

- meta-data (інформація про правило, не надає впливу на детектування пакетів і виконувані по відношенню до них операції);
- payload (опція перегляду поля даних пакета (packet payload));
- non-payload (опція перегляду службових полів пакету);
- post-detection (опція, яка вказує, що потрібно зробити після виконання заданих для правила умов).

Розглянемо більш детально кожну категорію опцій правил.

Розпочнемо з опцій категорії **meta-data**.

Опція **msg** говорить машині запису в журнальний файл і генерації сигналів про необхідність включення текстового повідомлення в запис журнального файлу або дампу пакета. Ця опція представлена текстовим рядком з використанням \ як escape-символа для задання символів, що мають спеціальне значення в правилах Suricata (наприклад, символ ;).

Приклад формата msg:

msg: “<текст повідомлення>”

Ключове слово **reference** дозволяє включати в правила посилання на зовнішні системи ідентифікації атак. Модуль в даний час підтримує декілька таких систем, а також унікальні URL.

Основними системами є:

- bugtraq;
- eve;
- nessus;
- arachnids;
- mcafee;
- url.

Наступною опцією категорії meta-data є **sid**.

Ключове слово sid призначене для ідентифікації правил Suricata. Значення ідентифікаторів використовуються підключеними модулями виводу. Опцію слід використовувати разом з ключовим словом rev (див. нижче). У правилах глобального значення, розповсюджуваних з програмою Suricata використовуються значення sid в діапазоні від 100 до 1000000 Значення менше 100 зарезервовані, а значення, які перевищують 1 000 000, призначені для локального використання (ідентифікації ваших власних правил)

Ключове слово **rev** дозволяє вказати номер ревізії (версії) правила з даним ідентифікатором. Цю опцію слід використовувати спільно з sid.

Однією з найважливіших опцій категорії meta-data є **classtype**. Ключове слово classtype вказує категорію сигналу відповідно до класу атаки по частоті використання і важливості. Користувач може самостійно задати рівень пріоритету для кожного типу правил. Класифікація правил визначена в файлі classification.config з використанням простого синтаксису:

config classification: <ім'я класу>, <опис класу>, <пріоритет за замовчуванням>

Не менш важливою опцією meta-data є **priority**. Тег priority використовується для присвоювання правилам рівня пріоритету. Опція classtype присвоює правилу прийнятий за замовчуванням рівень пріоритету, який можна змінити за допомогою priority.

Приклад спільного використання опції classtype і priority показаний нижче.

alert tcp any any -> any 80 (msg: “EXPLOIT ntpdx overflow”; \

dsize: >128; classtype:attempted-admin; proirity:10;)

Наступна категорія опцій правил, яку розглянемо, буде категорію перевірки вмісту пакетів (**payload**).

Однією з найбільш вживаних опцій цієї категорія є **content**. Ключове слово content забезпечує підтримку однієї з найважливіших функцій Suricata - перевірку вмісту пакетів. Ця опція дозволяє користувачеві створювати правила для пошуку в пакетах певної інформації та виконання тих чи інших дій при її виявленні. Для перевірки вмісту пакетів використовується функція пошуку за шаблоном Boyer-

Moore. Якщо задана послідовність даних виявлена в полі вмісту пакета, перевірка вважається успішною і виконується інша частина правила. Слід пам'ятати, що при пошуку враховується регістр символів.

Аргумент опції може містити як текст, так і виконавчі дані (зазвичай вони вказуються між парою символів | і задаються послідовністю шістнадцятирічних представлень байтів). Нижче показаний приклад завдання рядка пошуку, який містить текст і бінарні дані.

alert tcp any any -> any 139 (content: “|5c00|P00|I|00|E|00 5c|”;)

Відзначимо, що в одному правилі може бути присутнім більше однієї опції content, що дозволяє знижувати рівень помилкових спрацювань за рахунок більш точного завдання шуканих послідовностей.

Якщо перед опцією стоїть знак заперечення (!), правилу будуть відповідати пакти, що не містять зазначених даних. Така можливість корисна для генерації сигналів з разі виявлення пакетів, що не містять заданої послідовності.

Більш детальна інформація щодо конфігурацій правил Suricata можна знайти на офіціальному сайті системи.

Узагальнена структура правила зображена на рисунку 3.20.

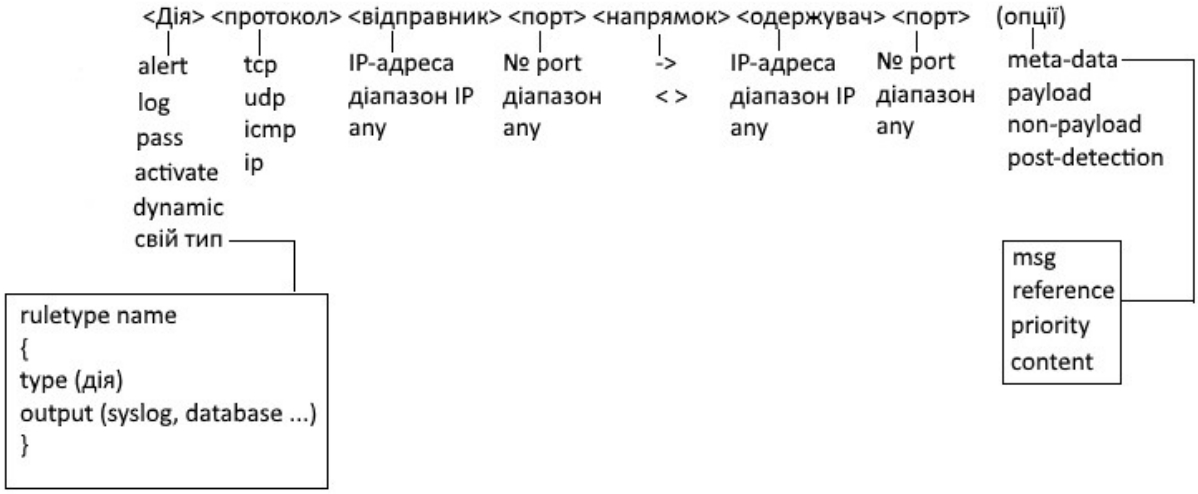



Рисунок 3.20 – Структура правил IDS/IPS Suricata

В цілому при створенні правил в IDS/IPS Suricata необхідно дотримуватися певних рекомендацій. Перерахуємо найважливіші з них:

- Suricata на першій фазі пошуку виконує перевірку відповідності заданим шаблонам (pattern). Чим довше шукана послідовність, тим точніше буде результат пошуку. Правила без опції content (або uricontent) уповільнюють роботу системи в цілому;
- намагайтеся писати правила так, щоб вони протидіяли використанню вразливостей, а не конкретним експлойтам. Наприклад, розумно шукати вразливі команди із занадто довгими аргументами, ніж shell-коди. Правила для

запобігання використання вразливостей менш уразливі з точки зору їх обходу шляхом простої зміни коду експлойта;

- враховуйте в правилах випадки нетипового використання протоколів;
- система пошуку по вмісту (content matching) в деяких випадках використовує рекурсію, тому неакуратно написані правила можуть призводити до того, що програма Suricata буде витрачати час на дублювання перевірок.

# 4. СУЧАСНІ ТА ПЕРСПЕКТИВНІ ТЕХНОЛОГІЇ ЗАХИСТУ ІНФОРМАЦІЇ В ІТС.

## 4.1 Антивірусні технології.

Шкідливе програмне забезпечення (ШПЗ) або шкідлива програма – це код, який виконує шкідливі дії; він може прийняти форму виконавчого файлу, скрипта, коду або будь-якого іншого програмного забезпечення (ПЗ). Зловмисники використовують ШПЗ для крадіжки конфіденційної інформації, для шпигування за зараженою системою або з метою взяти систему під контроль.

Нижче наведені деякі шкідливі дії, що виконуються ШПЗ:

- порушення роботи комп'ютера;
- крадіжка конфіденційної інформації;
- несанкціонований доступ до системи жертви;
- шпигунство;
- посилання спам-листів;
- участь в DDoS атаках;
- блокування файлів на комп'ютері та утримання їх з метою викупу;

Проаналізувавши різні джерела можна виділити таку класифікацію ШПЗ, яку представлено у вигляді схеми (рис. 4.1):

За середовищем інфікування:

- Файлове ШПЗ, воно поділяється на два типи:
  - пряме інфікування - ШПЗ при завантаженні починає шукати файли, які воно здатне заразити. Такі шкідливі програми розрізняються по принципу інфікування. Перезаписуючі файлові шкідливі програми - вони повністю замінюють код файлу на шкідливий. Товариські шкідливі програми - вони змінюють розширення файлу та додають до нього атрибут прихованого файлу й створюють свою копію з назвою прихованого файлу. При його виконанні спочатку виконується шкідливий код, а потім виконається прихований файл. Паразитичні шкідливі програми - вони беруть контроль над інфікованим файлом додаючи до початку або до кінця файлу шкідливий код;
  - Інфікування пам’яті - ШПЗ завантажується в пам’ять, не заражаючи файли напряму, а чекає, поки буде виконаний, файл, що підходить для інфікування, а потім інфікує його;
- Шкідливі програми завантажувача операційної системи (ОС) - шкідливі програми, що інфікують сектор диску для завантаження ОС, щоб отримати контроль над комп’ютером, ще до завантаження операційної системи;
- Файлово-завантажуючі шкідливі програми - ШПЗ, що інфікує як файли так и сектор диску для завантаження ОС;

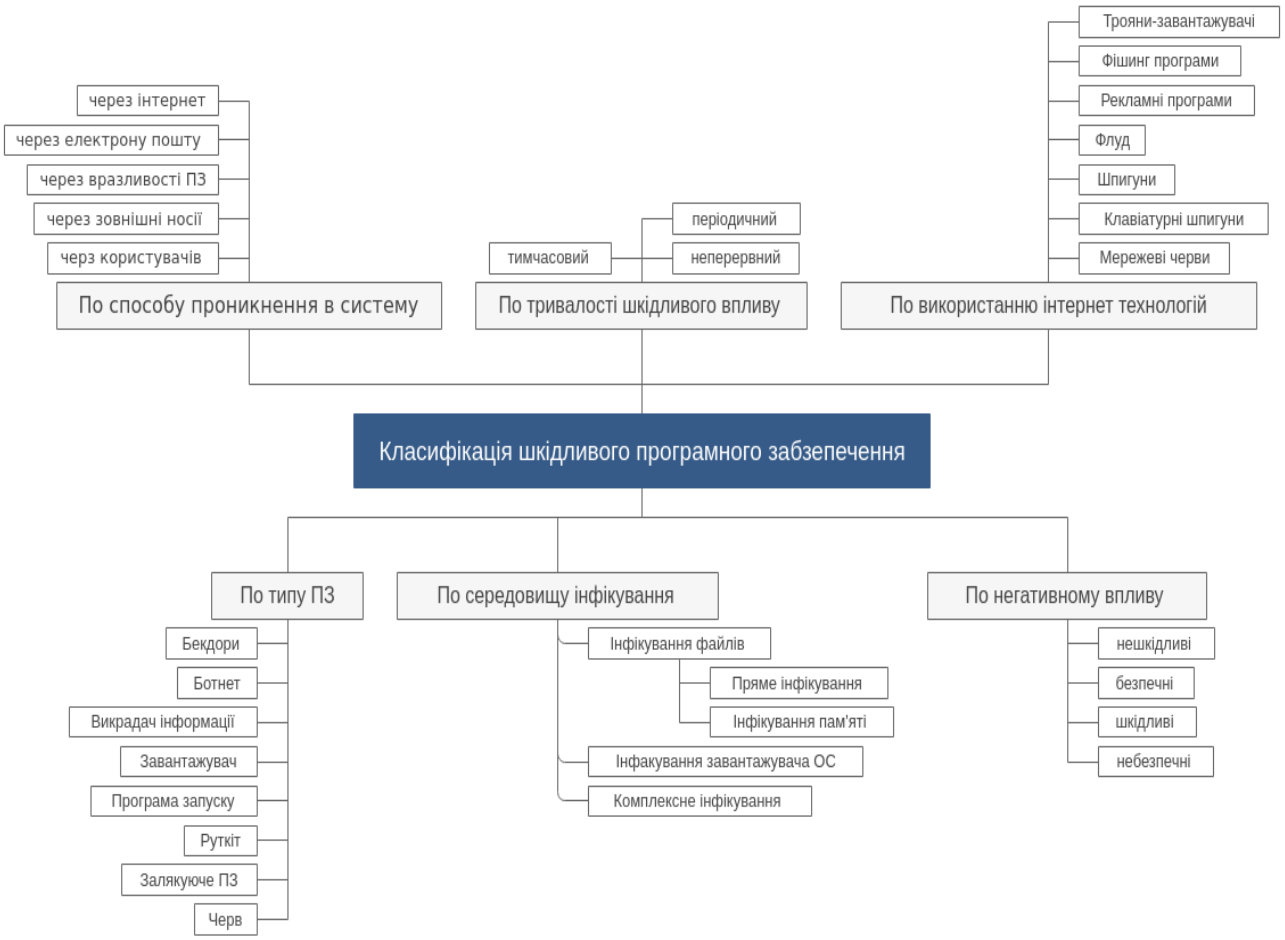



Рисунок 4.1 - Класифікація шкідливого програмного забезпечення

За негативним впливом на заражений комп’ютер:

- нешкідливі - ті що не впливають на продуктивність та функціонал зараженої системи;
- безпечні - зменшують кількість вільної пам’яті та їх діяльність супроводжується графічними та/або аудіо ефектами;
- шкідливі - їх діяльність може призвести до порушень роботи комп’ютера;
- небезпечні - такі шкідливі програми, можуть змінювати, модифікувати та знищувати інформацію, що міститься на комп’ютері;

За способом проникнення в систему:

- через Інтернет - глобальна інформаційна мережа - основне джерело розповсюдження ШПЗ;
- через електронну пошту - повідомлення, що надходять до користувачів можуть містити ШПЗ;
- через вразливості ПЗ - недолік або слабкість програмного засобу чи інформаційної системи (ІС), яку можна використати для розповсюдження шкідливих програм;

- через зовнішні носії - розповсюдження ШПЗ на зовнішніх носіях даних;
- через користувачів - користувачі самі можуть завантажити шкідливу програму, шукаючи потрібне їм ПЗ на неперевірених джерелах або використовуючи піратським ПЗ;

За тривалістю шкідливого впливу:

- неперервний;
- періодичний;
- тимчасовий;

За використанням мережевих технологій:

- Трояни-завантажувачі - ШПЗ, що несанкціоновано завантажують інформацію з мережі;
- Фішинг програми - ШПЗ, що надсилає листи електронної пошти, з метою отримання від користувача конфіденційної інформації;
- Рекламні програми - ШПЗ, що без оповіщення користувача включені в заражену програму з метою показу реклами;
- Флуд - ШПЗ, що несанкціоновано використовує атаковану систему в розподіленій атаці на відмову в обслуговуванні;
- Шпигуни - шкідливі програми, основна мета яких слідкування за користувачами та збір конфіденційної інформації;
- Клавіатурні шпигуни - ШПЗ, що призначені для несанкціонованого запису інформації, що користувач вводить до комп'ютера через клавіатуру та надсилання її зловмиснику;
- Мережеві черви - шкідливі програми, що самостійно розповсюджуються комп'ютерною мережею і використовують вразливості електронно-обчислювальних машин (ЕОМ) для проникнення до їх системи та подальшого розповсюдження;

За принципом роботи ШПЗ:

- Бекдор - шкідливий код, який встановлюється на комп'ютер, щоб відкрити доступ зловмиснику. Бекдори зазвичай дозволяють під'єднатися до комп'ютера з мінімальною аутентифікацією або зовсім без неї та виконати команди в локальній системі;
- Ботнет - шкідливий код, який відкриває зловмисникам доступ до системи, чим схожий на бекдор, проте всі комп'ютери, заражені одним ботнетом, отримують одні і ті ж інструкції від єдиного керівного сервера;
- Завантажувач - ШПЗ, єдиною метою якого є завантаження іншого шкідливого коду. Зловмисники зазвичай встановлюють завантажувачі при початковому доступі до системи. Це ШПЗ завантажить і встановить інше ШПЗ;
- Викрадач інформації - ШПЗ, яке збирає інформацію на комп'ютері жертви і, як правило, відправляє її зловмисникам. Як приклад можна привести програми, що захоплюють хеші паролів, перехоплювачі і

кейлогери. Це ШПЗ, зазвичай, використовуються для отримання доступу до облікових записів інтернет-додатків, таких як електронна пошта або інтернет-банкінг;

- Програма запуску - ШПЗ, за допомогою якої запускається інший шкідливий код. Зазвичай в таких програмах використовуються нетрадиційні методики запуску, що дозволяють непомітно отримати доступ до системи або підвищити привілеї;
- Руткіт - ШПЗ, що приховує існування іншого коду. Руткіти зазвичай застосовуються в поєднанні з іншими шкідливими програми, такими як бекдори, що дозволяє відкрити зловмисникам доступ до системи і ускладнити виявлення коду;
- ШПЗ, що залякує - ШПЗ, створене для залякування атакованого користувача і схилення його до певних дії. Зазвичай має графічний інтерфейс, схожий з антивірусом або іншою програмою, що має забезпечувати безпеку ІС. Вона повідомляє користувачеві про наявність в його системі шкідливого коду і переконує його в тому, що єдиним виходом із ситуації є покупка певного «ПЗ», хоча насправді це лише видалить ШПЗ, що залякує;
- Програма для розсилки спаму - ШПЗ, яке заражає комп'ютер користувача і потім з його допомогою розсилає спам. Цей тип програм генерує дохід для зловмисників, дозволяючи їм продавати послуги з розсилки спаму;
- Черв - шкідливий код, який здатний копіювати себе і заражати інші комп'ютери.

ШПЗ часто охоплює кілька категорій. Наприклад, програма може одночасно містити кейлогер, збирати паролі і бути хробаком для розсилки спаму. Тому не варто зациклюватися на класифікації шкідливих програм по їх функціональності.

ШПЗ можна також класифікувати на основі того, чи є атака зловмисника масовою або цілеспрямованою. Масове ШПЗ, таке як ШПЗ, що залякує, створено для зараження якомога більшого числа комп'ютерів. Вони мають найбільше поширення, але поступаються в складності ШПЗ, що було розроблено для цілеспрямованої атаки. Тому масове ШПЗ простіше виявити і нейтралізувати, адже системи безпеки розраховані саме на них.

Цілеспрямоване ШПЗ, наприклад, може бути єдиний в своєму роді бекдор, який адаптується під конкретну ІС. Воно представляє велику загрозу для ІС тому, що зустрічається рідко, а значить, системи безпеки мають менше шансів впоратися з ним або хоча б виявити його. Без докладного аналізу такого цілеспрямованого ШПЗ усунути його і захистити мережу практично неможливо. Даний вид шкідливих програм зазвичай відрізняється підвищеною складністю і вимагає ретельного аналізу.

**Методи боротьби з ШПЗ:**

- **виявлення** ШПЗ;
- **блокування** роботи ШПЗ;
- **усунення** наслідків дії ШПЗ.

Виявлення ШПЗ бажано здійснювати на стадії його впровадження або, принаймні, до початку здійснення деструктивних функцій ШПЗ.

Усунення наслідків дії ШПЗ ведеться у двох напрямах:

- видалення ШПЗ;
- відновлення (при необхідності) файлів, областей пам'яті.

Відновлення системи залежить від типу ШПЗ, а також від елементу часу виявлення ШПЗ по відношенню до початку деструктивних дій.

**Методи виявлення ШПЗ:**

- сканування;
- виявлення змін;
- евристичний аналіз;
- використовування резидентних сторожів;
- вакцинація програм;
- апаратурно-програмний захист від вірусів.

**Сканування** – один з найпростіших методів виявлення ШПЗ. Сканування здійснюється програмою-сканером, яка проглядає файли у пошуках пізнавальної частини ШПЗ – сигнатури. Програма фіксує наявність вже відомого ШПЗ, за винятком поліморфного ШПЗ, які застосовують шифрування тіла вірусу, змінюючи при цьому кожного разу і сигнатуру. Програми-сканери можуть зберігати не сигнатури відомого ШПЗ, а його контрольну суму. Програми-сканери часто можуть видаляти знайдене ШПЗ. Такі програми називаються поліфагами. Метод сканування застосовується для виявлення ШПЗ, сигнатури якого вже виділені і є постійними. Для ефективного використовування методу необхідне регулярне оновлення відомостей про нове ШПЗ.

Метод **виявлення змін** базується на використовуванні програм-ревізорів. Ці програми визначають і запам'ятовують характеристики всіх областей на дисках, в яких звичайно розміщується ШПЗ. При періодичному виконанні програм-ревізорів порівнюються характеристики, що зберігаються, і характеристики, одержувані при контролі областей дисків. За наслідками ревізії програма видає відомості про можливу наявність вірусів. Звичайно програми-ревізори запам'ятовують в спеціальних файлах образи головного завантажувального запису, завантажувальних секторів логічних дисків, характеристики всіх контрольованих файлів, каталогів і номера дефектних кластерів. Можуть контролюватися також обсяг встановленої оперативної пам'яті, кількість підключених до комп'ютера дисків і їх параметри. Головною перевагою методу є можливість виявлення ШПЗ всіх типів, а також нового

невідомого ШПЗ. Досконалі програми-ревізори знаходять навіть стелс-віруси. Є у цього методу і недоліки. За допомогою програм-ревізорів неможливо визначити ШПЗ у файлах, які поступають в систему вже зараженими. ШПЗ буде знайдене тільки після розмноження в системі. Програми-ревізори непридатні для виявлення зараження ресурсів, оскільки документи і таблиці дуже часто змінюються.

**Евристичний аналіз** порівняно недавно почав використовуватися для виявлення ШПЗ. Як і метод виявлення змін, даний метод дозволяє визначати невідоме ШПЗ, але не потребує попереднього збору, обробки і зберігання інформації про файлову систему. Суть евристичного аналізу полягає в перевірці можливих середовищ існування ШПЗ і виявлення в ньому команд (груп команд), характерних для ШПЗ. Такими командами можуть бути команди створення резидентних модулів в оперативній пам'яті, команди прямого звернення до дисків, минувши ОС. Евристичні аналізатори при виявленні «підозрілих» команд у файлах або завантажувальних секторах видають повідомлення про можливе зараження. Після отримання таких повідомлень необхідно ретельно перевірити ймовірно заражені файли і завантажувальні сектори всіма наявними антивірусними засобами.

Метод використовування **резидентних сторожів** заснований на застосуванні програм, які постійно знаходяться в ОП ЕОМ і відстежують всі дії решти програм. У разі виконання якою-небудь програмою підозрілих дій (звернення для запису в завантажувальні сектори, переміщення в ОП резидентних модулів, спроби перехоплення переривань і т. п.) резидентний сторож видає повідомлення користувачу. Програма-сторож може завантажувати на виконання інші антивірусні програми для перевірки «підозрілих» програм, а також для контролю всіх файлів (із змінних дисків, флешек, по мережі), що поступають ззовні. Істотним недоліком даного методу є значний відсоток помилкових повідомлень, що заважає роботі користувача, викликає роздратування і бажання відмовитися від використовування резидентних сторожів.

Під **вакцинацією програм** розуміється створення спеціального модуля для контролю її цілісності. Як характеристика цілісності файлу, зазвичай використовується контрольна сума. При зараженні вакцинованого файлу, модуль контролю знаходить зміну контрольної суми і повідомляє про це користувача. Метод дозволяє знаходити все ШПЗ, у тому числі і незнайоме, за винятком стелс-вірусів.

**Методи усунення наслідків зараження ШПЗ**

У процесі усунення наслідків зараження ШПЗ здійснюється видалення ШПЗ, а також відновлення файлів і областей пам'яті, в яких знаходилося ШПЗ. Існує два методи видалення наслідків дії ШПЗ антивірусними програмами.

***Перший метод*** припускає відновлення системи *після дії відомого ШПЗ*. Розробник програми-фага, що видаляє ШПЗ, *повинен знати структуру ШПЗ* і його характеристики розміщення в середовищі існування.

***Другий метод*** дозволяє відновлювати файли і завантажувальні сектори, заражені *невідомим ШПЗ*. Для відновлення файлів програма відновлення повинна *завчасно створити і зберігати інформацію про файли, одержану в умовах відсутності ШПЗ*. Маючи інформацію про незаражений файл і використовуючи відомості про загальні принципи роботи ШПЗ, здійснюється відновлення файлів. Якщо ШПЗ піддало файл необоротним змінам, то відновлення можливе тільки з використанням резервної копії або з дистрибутива. При їх відсутності існує тільки один вихід – знищити файл і відновити його самостійно. Якщо антивірусна програма не може відновити головний завантажувальний запис або завантажувальні сектори, то можна спробувати це зробити самому. У разі невдачі слід відформатувати диск і встановити ОС.

**Технологія виявлення ШПЗ**

*Технологія емуляції* дозволяє запускати додаток в середовищі емуляції, емулюючи поведінку операційної системи або центрального процесора. При виконанні програми в режимі емуляції додаток не зможе завдати шкоди системі користувача, а шкідливий вплив буде виявлятися емулятором. Незважаючи на гадану ефективність даного підходу, він також не позбавлений недоліків - емуляція займає надто багато часу і ресурсів комп'ютера користувача, що негативно позначається на швидкодії при виконанні повсякденних операцій, також, сучасні шкідливі програми здатні виявляти виконання в імітованого середовища і припиняти своє виконання в ній.

*Технологія віртуалізації робочого оточення* працює за допомогою системного драйвера, який перехоплює всі запити на запис на жорсткий диск і замість виконання запису на реальний жорсткий диск, виконує запис в спеціальну дискову область - буфер. Таким чином, навіть в тому випадку, якщо користувач запустить шкідливе програмне забезпечення, воно проживе не далі ніж до очищення буфера, який за замовчуванням виконується при виключенні комп'ютера. Однак, слід розуміти, що технологія віртуалізації робочого оточення не зможе захистити від шкідливих програм, основною метою яких є крадіжка конфіденційної інформації, тому що доступ на читання до жорсткого диска не заборонений.

*Технологія забезпечення безпеки на основі політик безпеки*.

*Політики, що поширюється до появи оновлення антивірусних баз або патчів* (для того, щоб «заткнути дірку» повільної роботи антивірусних лабораторій, на формування політики також потрібен час (не завжди менше, ніж на аналіз вірусу для додавання процедур детектування в антивірусну базу), ще одним недоліком даного підходу є частота зміни політик безпеки)

*Політики, які роблять неможливою експлуатацію тих чи інших вразливостей в програмному забезпеченні*. *Обмеження доступу в корпоративну мережу для комп'ютерів, які не відповідають політиці безпеки компанії* (наприклад: відсутні необхідні оновлення операційної системи, немає останніх оновлень антивірусних баз і т.д.). Для приведення комп'ютера у відповідність політиці виділяється доступ тільки на спеціальний сервер оновлень. Після установки всіх необхідних оновлень і виконання інших дій, необхідних політикою безпеки, комп'ютер отримує доступ в корпоративну мережу.

*Технологія забезпечення безпеки на основі IPS* (Intrusion Prevention System) Системи запобігання вторгнень передбачають можливість закриття найбільш часто використовуваних шкідливими програмами вразливостей комп'ютера перед новою загрозою, ще до виходу оновлення антивірусних баз: блокування портів, тобто можливості попадання інфекції на комп'ютер і її подальшого розмноження; створення політик для обмеження доступу до тек або окремих файлів; виявлення джерела інфекції в мережі і блокування подальших комунікацій з ним. Дана технологія відмінно працює проти атак хакерів і без файлових черв'яків і вірусів, але проти поштових черв'яків, класичних вірусів і троянських програм IPS неефективна.

*Технологія забезпечення безпеки від переповнення буфера*. Ідея технології - не допустити переповнення буфера для найбільш поширених програм, сервісів Windows, включаючи Word, Excel, Internet Explorer, Outlook і SQL Server. При більшості сучасних атак задіються різні вразливості, що використовують переповнення буфера. Запобігання переповнення буфера також можна віднести до проактивного захисту, тому що ця технологія просто виключає використання такої вразливості будь-яким шкідливим кодом або атакою.

**Система антивірусного захисту** виконує функції із захисту серверів та робочих станцій від атак комп’ютерних вірусів та перекриває найбільш імовірні варіанти проникнення вірусів до мережі. Система антивірусного захисту забезпечує:

- можливість невпинного захисту об’єктів інформаційно-обчислювальної мережі відповідно до встановлених вимог та політики безпеки;
- можливість автоматизованого блокування проникнення комп’ютерних вірусів з усіх можливих джерел;
- принципи мінімальної достатності, автоматичного функціонування та централізованого захищеного віддаленого керування антивірусним програмним забезпеченням і має доповнюватися адміністративними заходами. Також система антивірусного захисту не повинна істотно знижувати продуктивність об’єктів інформаційно-обчислювальної мережі;
- роботу з повним набором існуючих операційних систем для серверів та робочих станцій інформаційно-обчислювальної мережі;

- “лікування” усіх відомих комп’ютерних вірусів із занесенням інформації про це у відповідні протоколи роботи для забезпечування моніторингу роботи. В разі підозри на зараження невідомими комп’ютерними вірусами та неможливості лікування забезпечується блокування доступу користувачів до цієї інформації;
- можливість оперативного повідомлення відповідальних осіб щодо виникнення критичних або особливих ситуацій у мережі, тобто: виявлення вірусу, збої у системі оновлень, змінення налаштовувань системи антивірусного захисту. Також передбачено генерацію керувальних SNMP-послідовностей з метою візуалізації контрольованих подій у системі мережного керування;
- автоматизоване централізоване оновлення антивірусного програмного забезпечення;
- гнучке масштабування за появи нових об’єктів антивірусного захисту;
- ліцензійність та підтримування постачальниками застосовуваного антивірусного програмного забезпечення.

Структурно систему антивірусного захисту зорганізовано за ієрархічним рівневим принципом. Ефективний захист від комп’ютерних вірусів забезпечується комплексною системою антивірусного захисту, котра складається з чотирьох антивірусних рівнів і передбачає встановлення антивірусного програмного забезпечення на об’єкти, які є найбільш піддавані вірусному зараженню.

**На першому рівні антивірусного захисту** захищається інформаційно-обчислювальна мережа шляхом встановлення первинного антивірусного шлюзу та перевіряння усіх операцій взаємодії з глобальними мережами і, зокрема, Інтернетом. Контролюються як вхідний, так і вихідний трафіки протоколів HTTP та FTP. В разі виявлення антивірусним шлюзом комп’ютерного вірусу у трафіка, інфікована інформація блокується й чиняться дії, передбачувані шлюзовим антивірусним засобом.

**На другому рівні антивірусного захисту** захищаються сервери електронної пошти шляхом встановлення на них антивірусного програмного забезпечення, яке в режимі реального часу перевіряє вхідні та вихідні поштові повідомлення. В разі виявлення зараженого листа можуть надсилатися необхідні повідомлення про вірусну загрозу передбачуваним адресатам: відправникові, одержувачеві та адміністраторові. Якщо поштове повідомлення може бути вилікуване, воно надходить до місця призначення, а в противному разі – воно заблоковується до з’ясування обставин. Листи електронної пошти, з яких антивірусна програма не спромоглася вилучити комп’ютерного віруса, можуть бути знищені або спрямовані до Ізолятора.

**На третьому рівні антивірусного захисту** захищаються файлові сервери інформаційно-обчислювальної мережі: файлові, баз даних, та, за потреби, технологічні сервери й сервери розробників програмного забезпечення.

Антивірусне програмне забезпечення, встановлюване на файлові сервери, має виявляти спроби запису заражених та підозрюваних файлів на ці сервери. Інформація перевіряється в режимі реального часу в фоновому режимі або за запитом адміністратора. Є можливість заблоковувати спроби запису заражених, підозрюваних файлів на сервери. Кожна така спроба фіксується в протоколах та звітах антивірусної програми із зазначенням імені зараженого файла, шляху до нього. Інформація стосовно таких спроб також може бути негайно надіслана адміністраторові антивірусної безпеки та на робочу станцію, з якої вчинено спробу запису заражених та/або підозрюваних файлів. Ведення протоколів антивірусного захисту здійснюється централізовано, завдяки наявності утиліти мережного керування. На технологічні комп'ютери, які за режимом не мають доступу до джерел зараження комп'ютерними вірусами, антивірусне програмне забезпечення також встановлюється, але обслуговується вручну групою антивірусного захисту.

**На четвертому рівні антивірусного захисту** захищаються автоматизовані робочі місця (АРМ) користувачів та комп'ютери, яких з певних причин не підімкнено до мережі. АРМ мають у своєму складі антивірусний монітор та антивірусний сканер, який здійснює перевіряння за запитом користувача або адміністратора антивірусного захисту.

4.2 Технології виявлення та блокування кібератак на кінцевих пристроях (EDR, UBA, UEBA).

Gartner визначає **Endpoint Threat Detectionand Response** (EDR) як інструмент для детектування і розслідування підозрілих активностей (і їх слідів) на кінцевих точках. Таким чином, цей клас рішень може бути віднесений до продуктів сімейства Advanced Threat Protection. Архітектурно схема роботи рішення виглядає таким чином: агент на кінцевій точці відстежує події на рівні системи і мережі, і або відправляє інформацію про них на сервер або хмару для подальшого аналізу, або локально проводить аналіз, дозволяючи оперативно відреагувати на виявлені загрози. Технології машинного навчання і поведінкового аналізу, а також інтеграція с потоками Threat Intelligence, дозволяють виявляти невідомі загрози і надавати розширені інструменти для розслідування інцидентів і побудови звітів.

Сучасні рішення класу EDR дозволяють:

- забезпечувати моніторинг кінцевих точок в режимі реального часу і представляти наочну візуалізацію активностей всіх робочих станцій і серверів в корпоративній інфраструктурі з єдиної консолі;
- ефективно виявляти і пріоритезувати інциденти інформаційної безпеки в міру їх виникнення на кінцевих точках;
- записувати і зберігати інформацію по активностей на кінцевих точках для подальшого розслідування комплексних інцидентів;

- надавати необхідну інформацію фахівцям ІБ для оперативного розслідування інцидентів;
- реагувати на інциденти, забезпечуючи їх стримування, а також допомагають у відновленні робочих станцій у вихідне до інциденту стан;
- підтримувати можливість взаємодії з рішеннями класу EPP.


Рисунок 4.2 - Основні функціональні блоки EDR

**Наочність**

Рішення EDR дозволяє проводити спостереження за всіма діями кінцевих точок, наприклад, встановлення нового програмного забезпечення, скачування файлів, підвищення рівня привілеїв облікових записів, а також зміни в запущених процесах, в мережевій активність, в поведінці користувачів та ін.

**Виявлення**

Вкрай важливо якнайшвидше виявити атаку, що розвивається, до того, як вона завдасть серйозної шкоди організації. Рішення EDR дозволяють виявити складні атаки на їх ранніх стадіях, використовуючи комбінацію різних механізмів пошуку і методів аналізу невідомих загроз, в тому числі побудованих на базі машинного навчання і поведінкового аналізу.

Методи виявлення:

- IoC-сканування;
- антивірусний захист;

- виявлення аномальної поведінки;
- пісочниця;
- розвідка загроз;
- ретроспективний аналіз;
- кореляція.

**Розслідування.**

Швидкий доступ до всіх даних з кінцевих точок і до інформації про активність дозволяє аналітикам виконувати комплексне розслідування і глибокий аналіз джерел загроз. EDR надає докладні відомості про будь-які виявлені загрози, наприклад, як загроза проникла в інфраструктуру організації; список порушених робочих станцій і серверів; інформація про зміни в порушених атакою кінцевих точках; застосунки, що запускаються; створені в ході атаки файли; завантажені файли та ін.

**Реагування.**

Широкий набір інструментів дозволяє фахівцям з інформаційної безпеки переглядати події, оцінювати масштаб порушень і визначати всі порушені кінцеві точки. Рішення EDR дозволяють забезпечувати стримування складних інцидентів, усуваючи загрози на окремих кінцевих точках, і їх наслідки без впливу на роботу користувачів. Під час реагування здійснюється:

- зупинка запущених шкідливих процесів;
- карантин файлів;
- ізолювання заражених робочих станцій;
- проведення необхідних віддалених дій з файлами та реєстром;
- блокування підключень кінцевих точок до зловмисних доменів, URL- і IP-адресами;
- збір криміналістичних артефактів (образів оперативної пам'яті, образів жорсткого диска);
- відновлення робочих місць в стан що був до зараження.

**User [and Entity] Behavioral Analytics** (UEBA / UBA) - клас систем, що дозволяють на основі масивів даних про користувачів та IT-сутності (кінцевих станціях, серверах, комутаторах, тощо) за допомогою алгоритмів машинного навчання і статистичного аналізу будувати моделі поведінки користувачів і визначати відхилення від цих моделей, як в режимі реального часу, так і ретроспективно.

Як джерела даних для UEBA-систем можуть виступати файли журналів серверних і мережевих компонентів, журнали систем безпеки, локальні журнали з кінцевих станцій, дані з систем аутентифікації і навіть зміст листування в соціальних мережах, месенджерах і поштових повідомленнях.

Системи UEBA представлені як у вигляді окремих програмних рішень, так і у вигляді розширень для вже існуючих систем: SIEM (Security Information and

Event Management), DLP (Data Loss Prevention), EDR (Endpoint Detection and Response) та ін.

Системи UEBA, архітектурно, вирішують 4 **основні завдання**:

- Збір даних з різних джерел, здійснення як простого статистичного, так і розширеного аналізу, з використанням методів машинного навчання, в режимі реального часу і/або з певною періодичністю.
- Швидка ідентифікація атак і інших порушень, більшість з яких не визначаються класичними засобами ІБ.
- Пріоритизація подій, консолідованих з різних джерел (SIEM, DLP, AD, тощо), для більш оперативного реагування з боку адміністраторів ІБ.
- Більш ефективна реакція на події за рахунок надання адміністраторам ІБ розширеної інформації про інцидент, що включає всі об'єкти, які були залучені в аномальну активність.

В ядро будь-якої UEBA-системи включаються технології по роботі з великими масивами даних. І якщо у випадку з розширеннями до відомих SIEM-систем (IBM QRadar UBA, HPE ArcSight UBA, LogRhythm AI Engine) такі технології доступні з коробки, то самостійні рішення повинні або використовувати сторонні розробки (наприклад, Exabeam використовує Elastic Stack), або свої власні (Splunk UBA, Microsoft ATA).

UEBA здатні доповнювати широкий клас ІБ-систем і вирішувати завдання, які ці системи не можуть вирішити в принципі:

- Визначення скомпрометованих акаунтів користувачів і кінцевих станцій.
- Визначення та запобігання інсайдерських загроз.
- Моніторинг співробітників і їх прав доступу.

### 4.3 Технології протидії витокам інформації (DLP)

Системи захисту від витоків конфіденційної інформації (DLP) - це системи, що дозволяють в режимі реального часу проводити моніторинг і блокування вхідних і вихідних повідомлень співробітників, відправлення файлів на зовнішні носії, мережеві сховища інформації та веб-ресурси, а також контроль голосових і текстових повідомлень, переданих по протоколу SIP, з метою запобігання витоку конфіденційної інформації.

Порушення конфіденційності (як випадкове, так і навмисне) може спричинити втрату довіри клієнтів і партнерів, удар по репутації компанії, зниження її конкурентоспроможності та прямі фінансові збитки. Доступ з робочого місця до популярних сервісів веб-пошти, соціальних мереж і месенджерів збільшує ризик витоків конфіденційної інформації. При цьому блокування цих ресурсів провокує співробітників шукати нові обхідні шляхи комунікації, підвищуючи свою технічну грамотність.

Ефективним засобом контролю інформаційних ресурсів є комплекс організаційних заходів, розроблених регламентів і технічні засоби, що дозволяють контролювати і забезпечувати їх дотримання. До них відносяться системи класу DLP (Data Leak Prevention, захист від витоків інформації). Під «витоком» в даному випадку розуміється порушення конфіденційності важливої інформації.

DLP-системи будуються на аналізі потоків даних, які перетинають периметр інформаційної системи, що захищається. При детектуванні в цьому потоці конфіденційної інформації спрацьовує активна компонента системи, і передача повідомлень (пакета, потоку, сесії) блокується.

Системи DLP це технології, що дозволяють запобігти витоку конфіденційної інформації. У перебігу останніх декількох років використовувалася велика термінологія: Information Leakage Protection (ILP), Information Leak Protection (ILP), Information Leakage Detection & Prevention (ILDP), Content Monitoring and Filtering (CMF), Extrusion Prevention System (EPS) та ін. Але остаточним і найбільш точним терміном прийнято вважати Data Leak Prevention (DLP, запропонований агентством Forrester в 2005 г.).

Функціонал DLP-систем включає в себе моніторинг інформації, що передається співробітниками в інтернет, і контроль даних, що зберігаються на внутрішніх загальнодоступних ресурсах компанії, робочих станціях користувачів і копіюються на зовнішні носії. На сьогоднішній день DLP-рішення, як правило, мають у своєму розпорядженні широкі можливості для контролю інформації, засновані на контентному аналізі і більш складних алгоритмах (наприклад, цифрових відбитках, пошуку ідентифікаторів та ін.).

Системи захисту від витоків конфіденційної інформації виконують такі завдання:

- Зберігання копій конфіденційної інформації, що передається засобами електронної пошти, месенджерів і IP-телефонії, що відправляється на зовнішні носії та мережеві корпоративні та веб-ресурси, з метою подальшого розслідування інцидентів інформаційної безпеки в організації.
- Блокування передачі конфіденційної та іншої небажаної інформації за межі компанії.
- Блокування передачі невідповідної інформації у внутрішній мережі компанії.
- Блокування можливості використання співробітниками ресурсів організації в особистих цілях.
- Пошук місць розташування несанкціонованих копій конфіденційної інформації (пошук конфіденційних даних).

Крім цього, DLP-системи можуть використовуватися в якості систем контролю дій співробітників - контролювати їх присутність на робочому місці, відстежувати їх лояльність і благонадійність.

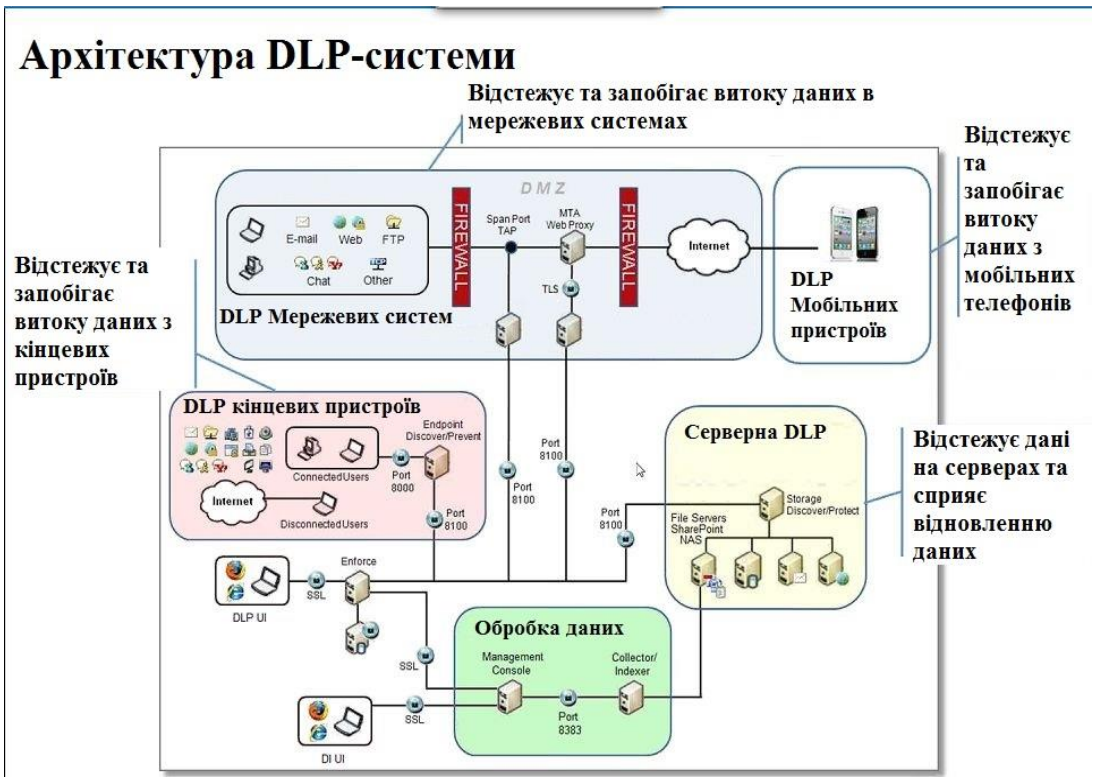



Рисунок 4.3 – Архітектура DLP систем

**Шлюзові DLP-системи** засновані на використанні шлюзів - централізованих серверів обробки мережевого трафіку, що на них надсилається. Область використання таких рішень обмежена самим принципом їх роботи. Іншими словами, шлюзові системи дозволяють захищатися лише від витоків інформації через протоколи, використовувані в традиційних інтернет-сервісах: HTTP, FTP, POP3, SMTP та ін. При цьому контролювати те, що відбувається на кінцевих точках корпоративної мережі з їх допомогою неможливо

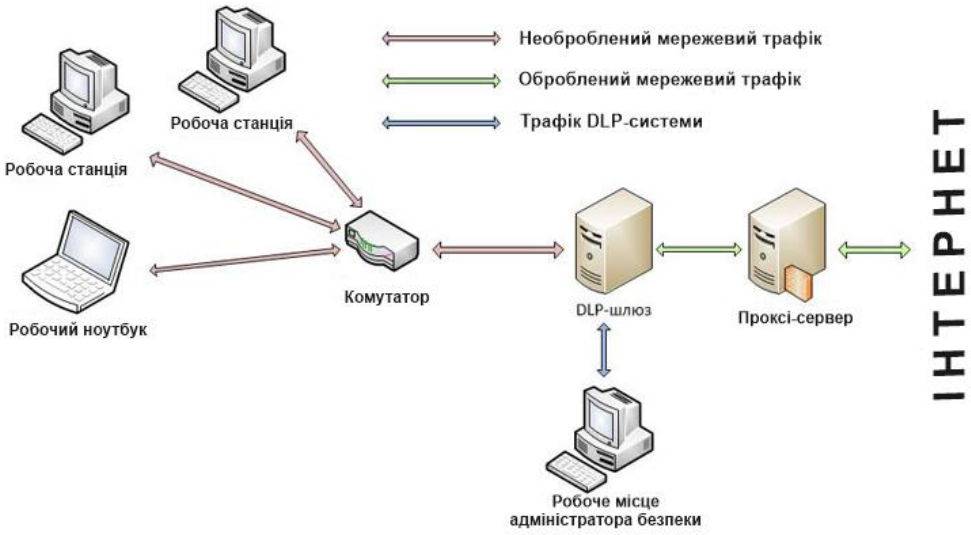



Рисунок 4.4 – Шлюзова DLP-системи, що працює в режимі блокування

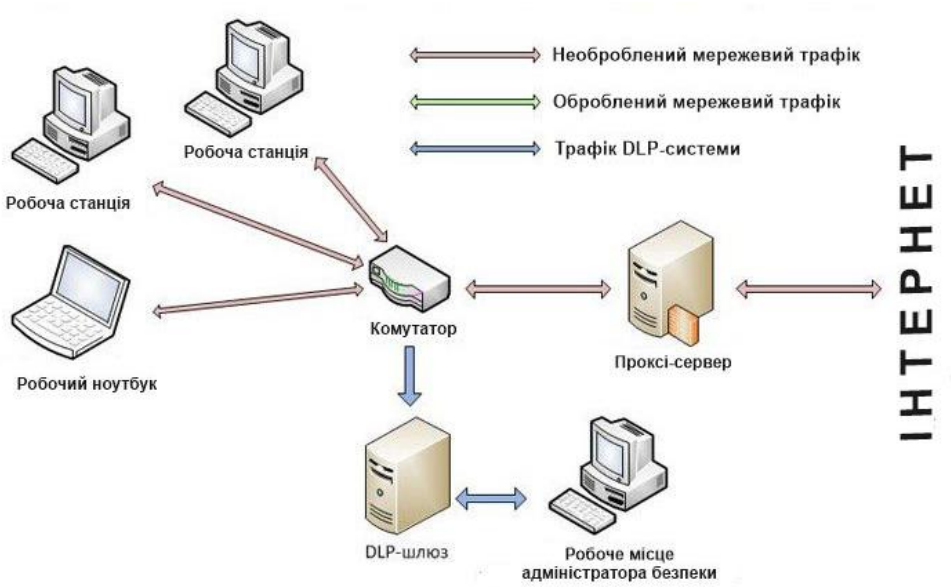



Рисунок 4.5 – Шлюзова DLP-системи, що працює в режимі моніторингу

**Хостові DLP-системи** засновані на використанні спеціальних агентів, які встановлюються на кінцевих точках корпоративної мережі. Ці програми грають відразу дві ролі. З одного боку вони контролюють діяльність користувачів комп'ютерів, не дозволяючи їм виходити за рамки встановленої політики безпеки (наприклад, забороняючи копіювати будь-які файли на "флешки"). А, з іншого, реєструють всі дії операторів і передають їх в централізоване сховище, дозволяючи співробітникам відділу інформаційної безпеки отримати повну картину того, що відбувається. Використання програм-агентів обмежує сферу застосування хостових DLP-систем: вони здатні бачити лише локальні або мережеві пристрої, підключені безпосередньо до тих комп'ютерів, на яких вони працюють.

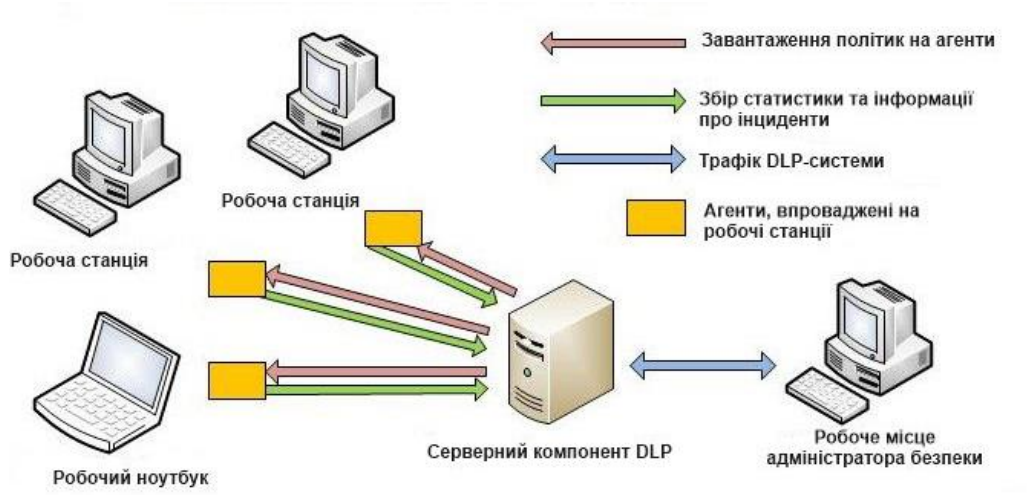



Рисунок 4.6 – Хостова DLP-система

4.4 Використання технологій обману в захисті ІТС (Deception Technology).

Призначення технології обману полягає в тому, що вона допомагає досягти будь–яких або всіх наступних чотирьох цілей безпеки:

- Увага – увагу противника може бути перенаправлено від реальних активів до фальшивих.
- Енергія – цінний час і енергію зловмисника можна витратити на фейкові цілі.
- Упевненість у собі – невизначеність може бути створена навколо правдивості виявленої вразливості.
- Аналіз – може бути створена основа для аналізу безпеки в реальному часі поведінки супротивника.

По–перше, DT може відвернути увагу противника від реальних цілей і ресурсів, збільшуючи його свободу дій. По–друге, технологія обману може привести до того, що противник вибере курс дій, який працює на користь захисника. По–третє, технологія обману може допомогти захиснику несподівано виявити противника. Нарешті, DT може захистити реальні ресурси від руйнування. Так як технологія принципово психологічна, можна вважати оманливу поведінку одного суб'єкта (зловмисника) як вплив поведінки іншого (цілі). Мета технології обману – запобігти кіберзлочинцю, який встиг проникнути в мережу, завдати будь–якої суттєвої шкоди.

DT працює принципово інакше, ніж звичайні системи захисту комп’ютерних мереж. Звичайні засоби захисту мають тенденцію працювати безпосередньо з діями зловмисника, наприклад, щоб запобігти їх або виявити. DT маніпулює мисленням зловмисника, змушуючи його діяти так, щоб це було вигідно захиснику. Будучи принципово іншим, DT може бути сильною там, де звичайні засоби захисту слабкі.

Deception technologies – сукупність технік імітації IT-інфраструктури та дезінформації зловмисників, які використовуються з метою:

- порушення засобів автоматизації зловмисника та затримки його дій або ж порушення прогресу атаки;
- виявлення та уповільнення просування атак зловмисників, що в результаті дозволяють зупиняти атаки до нанесення значного збитку;
- звернення уваги зловмисника на себе, щоб відвернути його від реальних ресурсів, при цьому зібрати інформацію про місцезнаходження зловмисника, його інструменти і методи атак.

Обман в цьому контексті використовується в якості техніки для оборонних або підривних цілей і не має протизаконного характеру.

На рисунку 4.7 представлено класифікацію існуючих типів Deception Technology за чотирма основними та незалежними критеріями (класами): за видом, за типом атакованих ресурсів, за рівнем взаємодії та за спеціалізацією.

Найпростішою формою Deception Technology є Honeypots, так звані приманки. Honeypot («Пастка») (з англ. – горщик з медом) – ресурс, що являє собою приманку для зловмисників. Метою якої є дослідження атак та несанкціонованого доступу до систем, що згодом дозволить вивчити стратегію зловмисника та визначити перелік засобів, за допомогою яких можуть бути завдані удари реально наявним об'єктам безпеки.

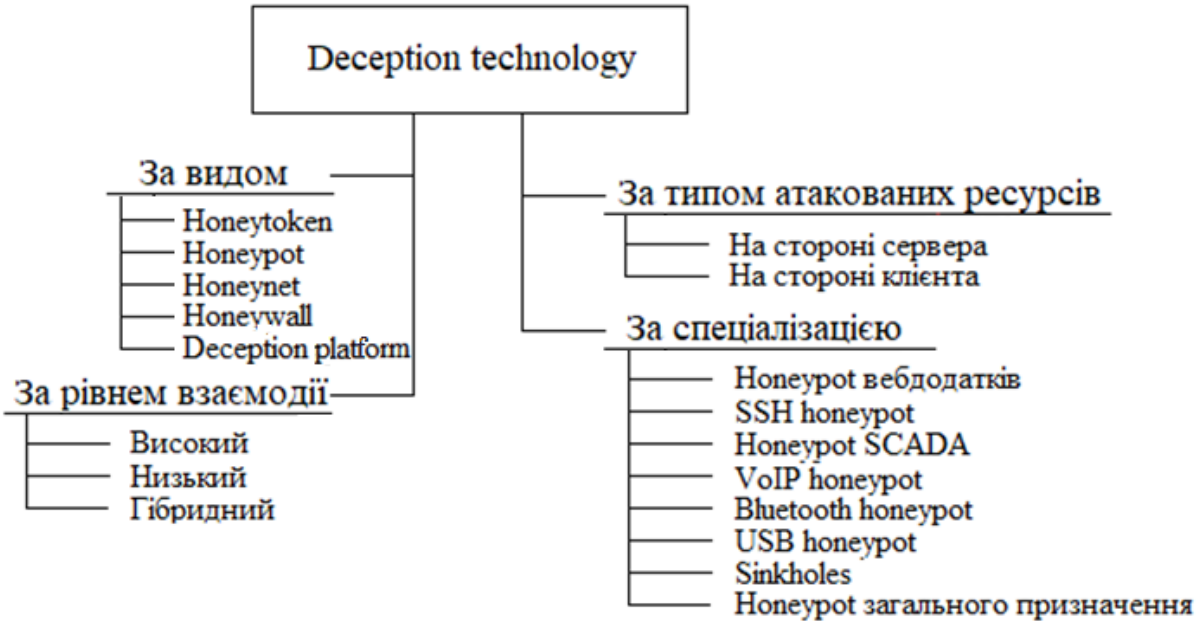



Рисунок 4.7 – Класифікація існуючих типів Deception Technology

Перший критерій (клас) – за типом атакованих ресурсів – описує, чи використовуються ресурси honeypot у режимі сервера чи клієнта. Honeypot на стороні сервера використовує мережеві сервіси, такі як SSH або NetBIOS, прослуховуючи їх стандартні порти і відстежуючи будь–які вхідні підключення віддаленими клієнтами. Навпаки, на стороні клієнта honeypot буде використовуватися набір клієнтських додатків, таких як веббраузер, які підключаються до віддалених служб і відстежують всю згенеровану активність.

- Honeypots на стороні сервера

Honeypots, призначені для виявлення і вивчення атак на мережеві сервіси, називаються серверними. Honeypots цього типу діють як сервер – вони надають відкритий порт, кілька портів або цілі програми та пасивно прослуховують вхідні з'єднання, встановлені віддаленими (ймовірно, шкідливими) клієнтами. Часто такі типи honeypot виявляють загрози, які використовують сканування як засіб виявлення потенційних жертв для компрометації – наприклад, сканування черв'яків або ботів – але їх також можна використовувати для виявлення спроб ручного зламування комп'ютерів. Honeypot на стороні сервера вважається «традиційною» приманкою.

- Honeypots на стороні клієнта

Honeypots на стороні клієнта, призначені для виявлення атак на клієнтські програми. Клієнтська програма – це частина програмного забезпечення, яка встановлює з'єднання з сервером і взаємодіє з ним. Найбільш популярним і найбільш цільовим типом клієнтських додатків є веббраузери разом з пов'язаними розширеннями і плагінами.

Клієнтські honeypot сильно відрізняються по своїй роботі від серверних. Вони активно встановлюють з'єднання з сервісами, щоб виявити шкідливу поведінку серверу або контенту. Найбільш популярними клієнтами є ті, які виявляють атаки на веббраузери та їх плагіни, поширювані через вебсторінки. Деякі з них також мають можливість переглядати різні форми вкладень, і були також спроби створити honeypot для миттєвих повідомлень.

Другий критерій (клас) – за рівнем взаємодії – визначає, чи є honeypot реальним ресурсом (з високим рівнем взаємодії) або тільки емуляцією (низький рівень взаємодії). Змішаний тип honeypot, який поєднує в собі обидві функції, називається гібридним honeypot. Різниця між ними полягає в тому, що приманки з низькою взаємодією відтворюють програмними або апаратними засобами або їх комбінацією роботи інших програм або пристроїв.

- Honeypots з низьким рівнем взаємодії

Honeypots з низьким рівнем взаємодії – це інструменти, які працюють, емулюючи свої ресурси: сервіси (в разі honeypots на стороні сервера) або клієнтські програми (у разі honeypots на стороні клієнту). Емуляція в цьому контексті означає, що ресурси, імітовані ресурсом honeypot, мають обмежену функціональність у порівнянні з реальними. Взаємодія з атакуючим в деякій мірі обмежена точністю емуляції. Ресурси honeypot повинні бути максимально схожими на їх реальні еквіваленти. Цей ступінь точності сильно впливає на процес взаємодії між honeypot і атакуючим. Недостатня точність може привести до того, що атаки будуть припинені достроково, навіть до того, як будуть здійснені реальні шкідливі дії. Це також значно полегшує виявлення honeypot.

Основною перевагою honeypot з низьким рівнем взаємодії є те, що їх легше розгортати і обслуговувати. Користувач має повний контроль над атакою і процесом зараження. Можна визначити поточну стадію атаки, яка представляє собою цінну інформацію. Емуляція також знижує ризик злому системи. З іншого боку, honeypot з низьким рівнем взаємодії мають ряд недоліків. Недоліком є їх низька точність емуляції. У певних випадках емульовані ресурси зазвичай ведуть себе не так, як реальні, незалежно від того, наскільки ретельна була спроба їх створення. Це може привести до того, що атака або зараження припиняться до остаточної фази або honeypot виявляється. Інша проблема полягає в тому, що неможливо розпізнати ще не відомі вразливості (так звані вразливості 0–дня). Вся діяльність, особливо на ранніх стадіях атак, повинна бути закодована в логіку honeypot, оскільки інструмент атакуючого очікує певної послідовності дій.

- Honeypot з високим рівнем взаємодії

Honeypot з високим рівнем взаємодії – це інструменти, які забезпечують реальні операційні системи та ресурси (клієнтські програми чи послуги). Факт використання «реальних» систем і ресурсів означає, що вони не емулюються. Проте, можна використовувати віртуальне середовище для таких цілей, і це насправді звичайна практика. У цій концепції сценарії взаємодії з зловмисником практично не обмежені, тому компроміс або процес зараження повинні бути повністю завершені у всіх випадках.

Реальна поведінка як операційної системи, так і ресурсів під час атаки є основною перевагою honeypot з високим рівнем взаємодії. Цей тип honeypot здатний виявляти атаки на вразливості 0–дня. Проте, область виявлення обмежена тільки певними (версіями) додатків, встановлених в середовищі honeypot.

Обсяг даних, що збираються honeypot з високим рівнем взаємодії, може бути більшим, ніж при використанні інструментів з низьким рівнем взаємодії. З іншого боку, через складність середовища honeypot виникають проблеми з визначенням того, які елементи поведінки системи/програми є підозрілими або шкідливими, а які – безпечними.

Іншим недоліком honeypots з високим рівнем взаємодії є обмежений контроль над етапами атаки. Ризик скомпрометувати реальні системи і, як наслідок, втратити контроль над honeypot, вище, ніж у аналога з низьким рівнем взаємодії. Інша проблема полягає в тому, що honeypots з високим рівнем взаємодії вимагають більше ресурсів у порівнянні з honeypots з низьким рівнем взаємодії через їх складності. Це впливає на масштабованість і продуктивність. Крім того, розгортання і використання honeypots з високим рівнем взаємодії, включаючи налаштування і управління, вимагає значних зусиль.

- Гібридні honeypot

Гібридні honeypot поєднують в собі інструменти як з низьким, так і з високим рівнем взаємодії, щоб отримати переваги обох.

Третій критерій (клас) – за спеціалізацією. Цей критерій визначає, яка служба або метод атаки/виявлення є основною областю дії honeypot.

Існує вісім можливих значень цього підкласу:

1. Honeypot вебдодатків – інструменти, призначені для виявлення атак на вебдодатки;
2. SSH honeypot – інструменти, орієнтовані на атаки Secure Shell (SSH);
3. Honeypot SCADA – інструменти, адаптовані промислові системи управління;
4. VoIP honeypot – інструменти виявлення загроз в інтернет–телефонії (Voice over IP);
5. Bluetooth honeypot – інструменти, призначені для виявлення атак, що поширюються по технології Bluetooth;

6. USB honeypot – інструменти, призначені для виявлення атак за допомогою USB–пристроїв;

7. Sinkholes – інструменти, що використовують «техніку провалів» для виявлення та моніторингу інфекцій в мережі;

8. Honeypot загального призначення – інструменти, призначені для виявлення більш ніж однієї техніки атаки або декількох служб.

Четвертий критерій (клас) – за видом. Варто звернути увагу на різницю між складовими частинами DT – honeypot, honeynet, honeytoken та honeywall.

Так як призначення honeypot було описано вище, наступною складовою частиною є honeynet. Мережа взаємопов'язаних honeypot, яка координує свої зусилля для забезпечення активного захисту мережі, називаються honeynet. Вона може бути відокремлена від робочої мережі. Керуючий трафік, що потрапляє в honeynet, повинен фіксуватися honeypot цієї мережі. При проектуванні honeynet (див. рис. 4.8) можна налаштувати honeypot з низьким рівнем взаємодії, тобто honeyd, який буде перенаправляти запити на певні порти на honeypot з високим рівнем взаємодії (або інший з низьким рівнем взаємодії). Таким чином, служби на визначених портах будуть добре емулюватися, в той час як на інших портах деякі мережеві пакети будуть як і раніше реєструватися.

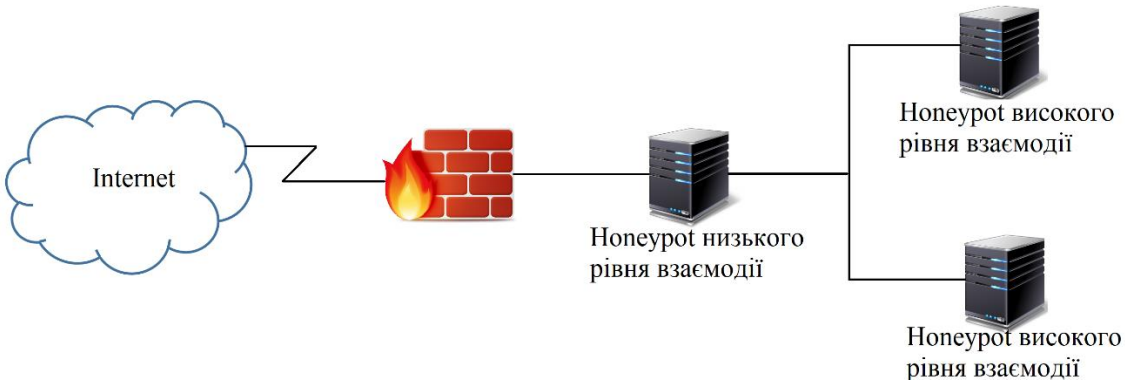



Рисунок 4.8 – Honeypot з низьким рівнем взаємодії, що перенаправляє обраний трафік на рішення з високим рівнем взаємодії

Може здатися гарною ідеєю виділити всі доступні невикористовувані IP–адреси і порти для honeynet. Тим не менш, це не завжди так. Велика кількість IP–адрес, що відповідають на всі порти (або, на їх частину) однаковим чином, може викликати підозри у зловмисника і фактично полегшити ідентифікацію honeynet. З цієї причини може бути доцільно зробити «прогалини» в адресному просторі, тобто не використовувати кілька IP–адрес поспіль. Важливо звернути увагу на те, що в більшості випадків краще мати всього декілька адрес в декількох різних мережах, відокремлених один від одного як фізично, так і логічно, ніж безліч IP–адрес в одній мережі.

Для успішної реалізації honeynet потрібно правильно розгорнути архітектуру honeynet. Ключем до архітектури honeynet є Honeywall – ще одна

складова частина DT, яка слугує для аналізу трафіку який йде з honeypot або ж на нього. Його можна використовувати для моніторингу і контролю всіх вхідних і вихідних з'єднань з honeynet. Honeywall дозволяє витягувати необроблений трафік в форматі PCAP, аналізує мережеві потоки. Крім того, для спрощення обслуговування системи надається набір командного рядка і графічні інструменти.

Основна мета вебінтерфейсу honeywall – дозволити користувачеві вивчати події, що відбуваються в honeynet. Модуль аналізу даних здатний перехоплювати дані, що надходять з honeypots, і зіставляти їх з мережевим трафіком. На основі цієї інформації модуль візуалізації може уявити деревовидний графік процесів, які вони викликали на атакованих машинах.

Весь трафік, що проходить через honeywall, розбивається на окремі потоки і відображається у вигляді списку з інформацією про: час події, виявлену операційну систему зловмисника, IP–адреси, які беруть участь в обміні даними, обсяг трафіку і кількість пакетів, обмін між кінцевими точками, і, нарешті, служба (номер порту і протокол), який піддається атаці. Оператор може виконати подальший аналіз потоків, вивчивши результати.

Проаналізувавши основні складові системи обману, можна зобразити рисунок, який показує місце й різницю між honeypot, honeynet, honeytoken та honeywall (див. рис. 4.9).

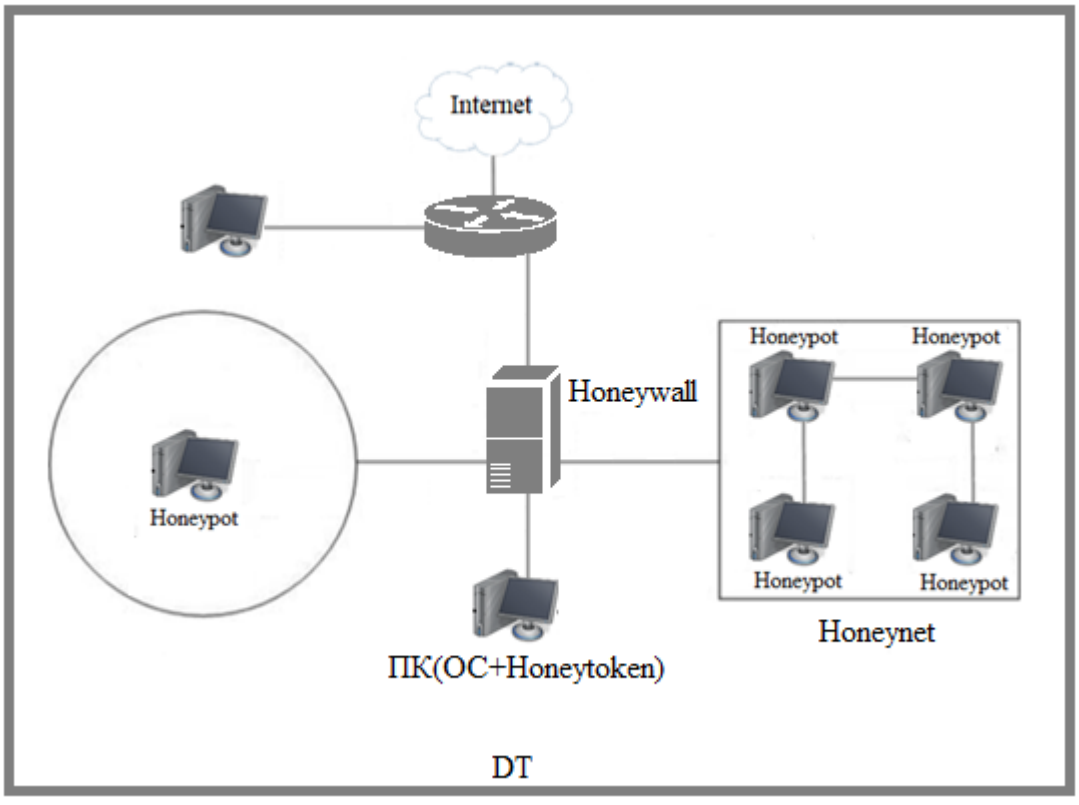



Рисунок 4.9 – Honeypot, honeynet, honeytoken та honeywall в Мережі

Deception Technology відноситься до ряду більш досконалих продуктів ніж honeypot і honeynet, які забезпечують більшу автоматизацію як для виявлення, так і для реалізації захисту на основі даних, які вони збирають. Важливо відзначити, що існують різні рівні deception technology. Деякі з них є чимось більшим, ніж просто honeypot, а інші імітують повноцінні мережі, які містять реальні дані і пристрої. Для захисту ресурсів також застосовують платформи deception – honeypots та «крихти» (які ведуть до honeypot) по всій мережі, щоб зробити все середовище пасткою, перетворюючи все навколишнє середовище в віртуальне мінне поле для активного залучення і дезінформації зловмисника, а також розкриття його присутності. Це дозволяє виявляти атаки на початку циклу, збирати відомості про загрози для конкретної компанії і виявляти зловмисника, перш ніж він зможе завдати якої-небудь суттєвої шкоди.

Концепція Deception Technology полягає в тому, щоб забезпечити захист ресурсів за допомогою використання honeypot, так як honeypot, безумовно, є її компонентом. Виходячи із значення Honeypot, їх поняття та класифікації, варто проаналізувати, яку роль вони відіграють в інформаційній безпеці та яке їх місце в структурній моделі ITC. Різні варіанти розміщення honeypot дадуть повні відомості про тактику нападника. Розміщення honeypot (див. рис. 4.10) всередині локальної мережі дасть уявлення про атаки зсередини мережі, а розміщення на загальнодоступних серверах цієї мережі або в DMZ – про атаки на незахищені мережеві служби, такі як: поштові сервіси, SMB, ftp–сервери і т.д.

- **Honeypot в локальній мережі:**

Honeypot може бути встановлений всередині локальної мережі (після firewall) тобто на комп'ютерах локальної мережі та серверах. Якщо не здійснюється віддалене адміністрування мережі, то слід перенаправляти весь вхідний ssh–трафік на honeypot. Приймаючи мережевий трафік, honeypot повинен протоколювати всі події, причому на низькому рівні. Honeypot – це не проста програма, яка записує логи сервера. Якщо зловмисник зміг отримати доступ до сервера, стерти всі логи йому не важко. В ідеалі всі події в системі повинні записуватися на рівні ядра.

- **Honeypot в DMZ**:

В DMZ розташовуються загальнодоступні сервери. Наприклад, це може бути вебсервер або поштовий сервер. Оскільки наявність таких серверів часто привертає увагу спамерів та зловмисників, необхідно забезпечити їхній інформаційний захист. Встановлення honeypot на сервері DMZ є одним з рішень цієї проблеми. За відсутності виділеного вебсерверу, можна емулювати вебслужби за допомогою програмних honeypot. Вони дозволяють в точності відтворити неіснуючий насправді вебсервер та заманити зловмисника. Емуляція поштових та інших мережевих служб обмежується лише вибором конкретного програмного рішення.

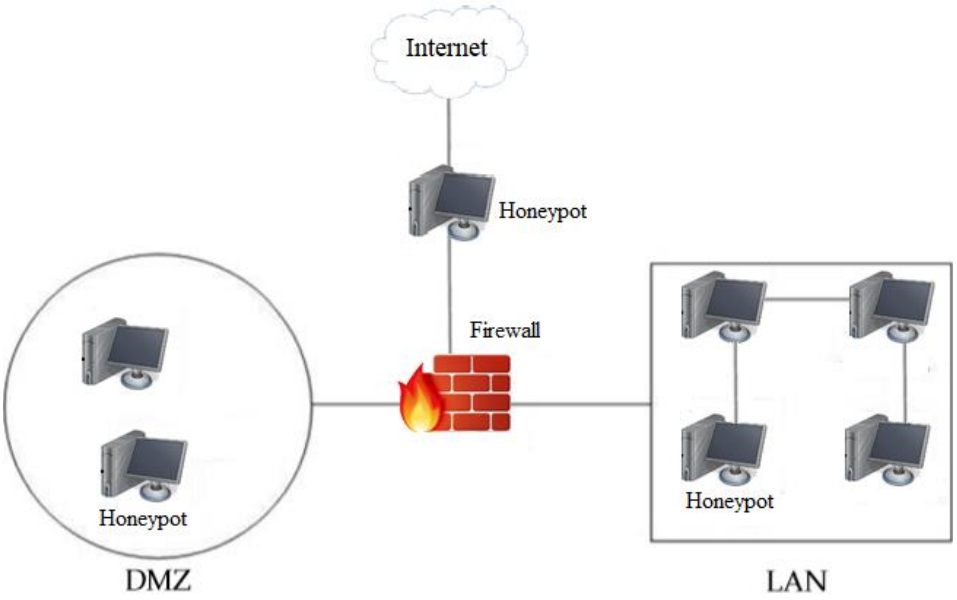



Рисунок 4.10 – Honeypot в локальній мережі та в DMZ

Існує кілька стратегій розгортання honeypots. Вони варіюються від установки одного honeypots до створення цілої мережі honeypots – honeynet. Стратегії залежать від розміщення honeypots, типу запитуваних даних, а також від обсягу ресурсів, які людина готова інвестувати в ці зусилля. Збір інформації – одна з основних функцій honeypots. Залежно від того, де, як і який honeypot буде розгорнуто, можуть бути зібрані різні типи інформації.

- **Типове розгортання в Інтернеті:**

Найбільш поширеним розгортанням для honeypots є конфігурація, що стоїть перед Інтернетом (див. рис. 4.11). Цей сценарій зазвичай використовується в тому випадку, коли ціль дослідження є збір зразків шкідливого ПЗ, подальший аналіз налаштувань honeynet. Також відстежується активність мережевих черв'яків або просто вивчається поведінка зловмисника. Це все може включати спостереження за шкідливою діяльністю в інтернеті (фоновий шум), а також вивчати нові вразливості.

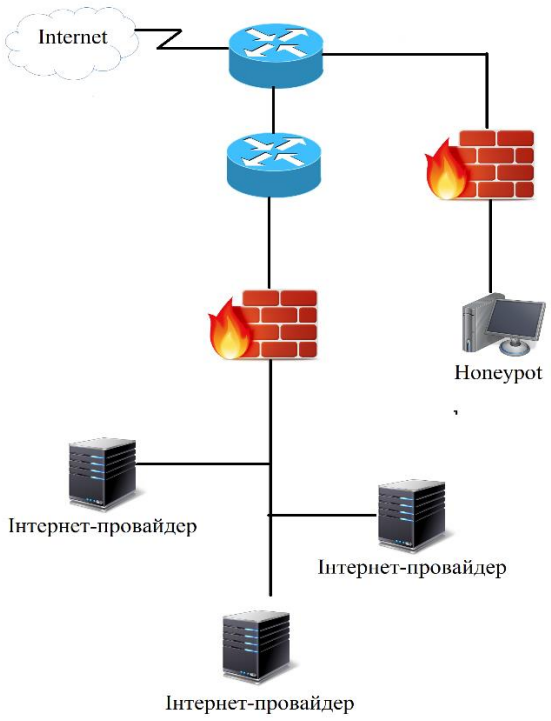



Рисунок 4.11 - Архітектура типового розгортання в Інтернеті

- **Внутрішнє розгортання**

Крім того, в сегментах виробничої мережі можна розмістити honeypots для виявлення зламаних систем і отримання інформації про внутрішню загрозу. В такому випадку, Honeypots повинен бути розміщений в іншому сегменті LAN(див. рис. 4.12) і перед цим йому потрібно призначити ip адрес. Необхідно дотримуватися обережності, щоб законний трафік не виявився на honeypot, так як це може викликати помилкові позитивні результати. Так як honeypot не має будь–якої виробничої цінності, будь–яка взаємоді з ним (заборона помилок конфігурації) означатиме зловмисну діяльність. Крім використання в якості датчика, він слугує для вивчення того, що відбувається після того, як мережева інфраструктура наражається на ризик нападника.

Особливо в контексті цілеспрямованих атак або так званих розширених постійних загроз (APT), такий підхід може бути дуже корисним, оскільки ці зловмисники переміщуються по мережі, використовуючи законні облікові дані, коли це можливо, і використовують шкідливі тільки в певних випадках. Звичайно, такий підхід не дуже простий, так як необхідно дозволити зловмиснику як і раніше мати доступ до мережі, honeypot повинен бути дуже реалістичним, щоб його не виявили відразу.

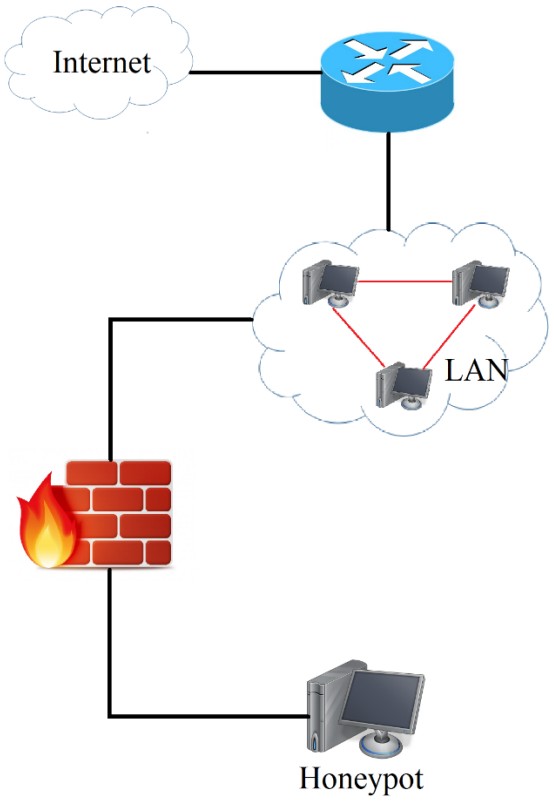



Рисунок 4.12 – Архітектура внутрішнього розгортання

Оскільки Honeypot неможливо повністю заховати в мережі, треба знати про ризик, пов'язаний з його розгортанням. Для того, щоб досягнути цілі, зловмисник не повинен легко ідентифікувати honeypot. При компрометації користь honeypot різко зменшується (якщо, звичайно, основною метою не було тривале спостереження за діяльністю нападника на honeypot). Зловмисник може обійти мережу honeypot або навіть ввести помилкові дані, що може істотно перешкодити аналізу даних або зробити його абсолютно неможливим.

Honeypot встановлюється в мережі з єдиною метою – бути атакованим, так як він розроблений з вже навмисними вразливостями. Основними компонентами архітектури Deception Technology є (див. рис. 4.13):

1. Система Honeypot – це не справжня система виробництва, а жертва для зловмисників. Це забезпечує атакуючим доступ до файлів honeypot і підроблених системних ресурсів. Автоматичні відповіді на дії зловмисника налаштовані так, щоб показати honeypot як справжню систему.
2. Honeywall – надають журнали про те, як зловмисник намагається проникнути в Honeypot. Honeywall налаштований так, щоб реєструвати всі пакети, що надходять в систему Honeypot.
3. Модуль моніторингу – це модуль оцінки загроз, який відстежує мережеві і/або системні дії на предмет шкідливих дій або порушень політики і створює звіти для станції управління. Перегляд порядку, послідовності, відміток

часу і типу пакетів, які використовуються зловмисником для отримання доступу до Honeypot, і натискання клавіш, доступ до системи, змінені файли і т. д., допомагають визначити інструменти, методологію, яка використовується зловмисниками, і їх наміри (вандалізм, крадіжка даних, пошук в точці віддаленого запуску і т. д.). IDS може виконувати роботу блоку моніторингу.

4. Модуль сповіщення – Honeypot повинен мати можливість генерувати сповіщення по електронній пошті, щоб відправляти адміністратору повідомлення про трафік, що йде на honeypot чи з нього, щоб він міг переглядати дії зловмисника, поки вони відбуваються.

5. Блок реєстрації – цей блок забезпечує ефективне сховище для всіх журналів міжмережевого екрану і системи, а також трафіку, що проходить між firewall і системою honeypot.

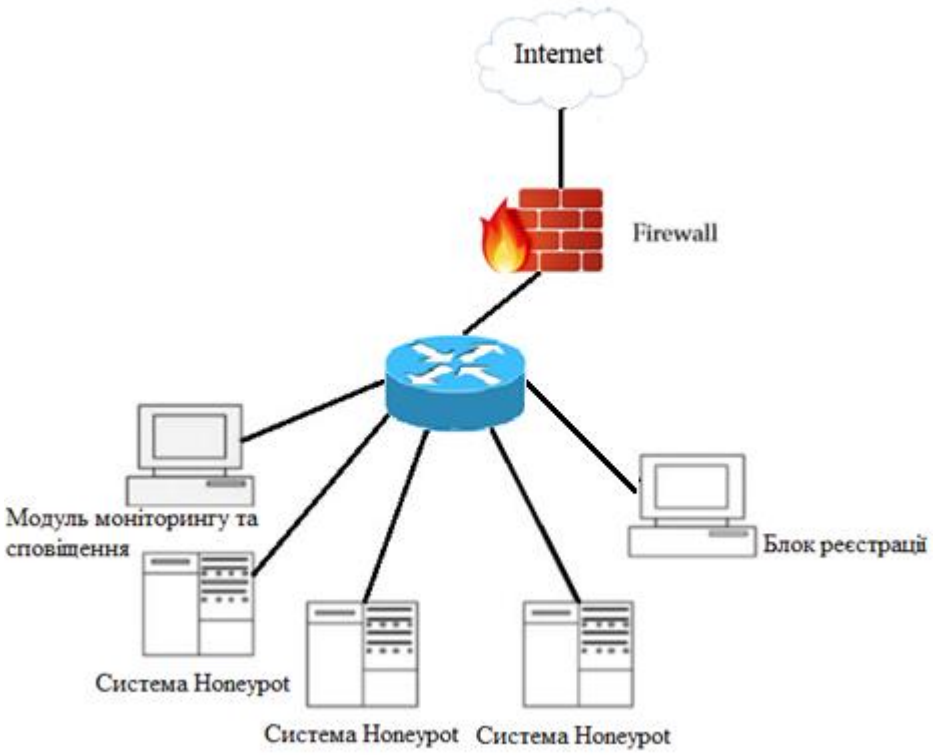



Рисунок 4.13 – Загальна архітектура DT

Стратегія DT полягає в безпечному відключенні зловмисників від виробничих систем та отриманні інформації про зловмисників шляхом реєстрації їх дій. На Рисунку 4.14 показаний приклад того, як DT може бути розгорнута в мережі з метою захисту. У цьому прикладі Honeypot–A моделює систему без будь–якого firewall або системи виявлення вторгнень (IDS); Honeypot–B імітує вразливу службу типу Ftp або Telnet, щоб привернути увагу зловмисника;

Honeypot–C і Honeypot–D імітують інші системи в мережі організації, щоб відвернути зловмисника від реальних систем.

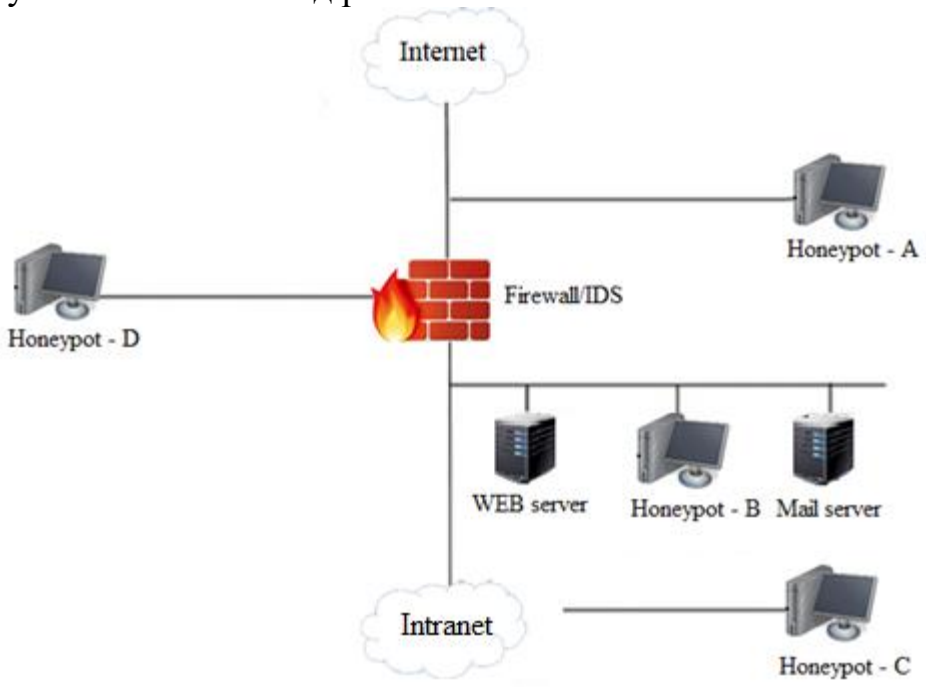



Рисунок 4.14 –Приклад розгортання DT в мережі

4.5 Технології механізму безпечного виконання програм (Sandbox).

**Пісочниця (англ. Sandbox)** — механізм для безпечного виконання програм. Пісочниці часто використовують для запуску непротестованого коду, неперевіреного коду з ненадійних джерел, а також для запуску та виявлення вірусів.

**Пісочниця** — це, зазвичай, жорстко контрольований набір ресурсів (місце на жорсткому диску та в оперативній пам'яті) для виконання гостьової програми. Доступ до мережі, системних ресурсів операційної системи, пряме зчитування інформації з пристроїв введення найчастіше або частково емулюються, або сильно обмежуються. Пісочниці є прикладом віртуалізації.

Підвищена безпека виконання коду в пісочниці передбачає захист системи від великих навантажень — саме тому деякі види пісочниць використовують для запуску невідлагодженого або шкідливого коду.

**Види пісочинць:**

**Аплети**, які виконуються у віртуальній машині або інтерпретаторі, що дозволяє запускати Java-код із будь-яких веб-сайтів без загрози операційній системі.

**Ізоляція на основі повної віртуалізації**. Використання будь-якої віртуальної машини в якості захисного шару над гостьовою операційною системою, де встановлений браузер і інші потенційно небезпечні програми, через які користувач може заразитися, дає досить високий рівень захисту основної робочої системи.

**Ізоляція на основі часткової віртуалізації файлової системи і реєстру.** Зовсім необов'язково тягати з собою движок віртуальної машини, можна підпихати процесам в пісочниці дублікати об'єктів файлової системи і реєстру, поміщаючи в пісочницю додатки на робочій машині користувача. Спроба модифікації даних об'єктів призведе до зміни лише їх копій всередині пісочниці, реальні дані не постраждають. Контроль прав не дає можливості атакувати основну систему зсередини пісочниці через інтерфейси операційної системи. Недоліки такого підходу також очевидні - обмін даними між віртуальним і реальним оточенням утруднений, необхідне постійне очищення контейнерів віртуалізації для повернення пісочниці до початкового, незараженого стану. Також, можливі пробої або обхід такого виду пісочниць і вихід шкідливих програмних кодів в основну, незахищену систему.

SandboxIE, BufferZone, ZoneAlarm ForceField, ізольоване середовище Kaspersky Internet Security, Comodo Internet Security sandbox, Avast Internet Security sandbox.

**Ізоляція на основі правил**. Всі спроби зміни об'єктів файлової системи і реєстру не віртуалізуються, але розглядаються з точки зору набору внутрішніх правил засобів захисту. Чим повніше і точніше такий набір, тим більший захист від зараження основної системи надає програма. Тобто, цей підхід є якийсь компроміс між зручністю обміну даними між процесами всередині пісочниці і реальною системою і рівнем захисту від шкідливих модифікацій. Контроль прав не дає можливості атакувати основну систему зсередини пісочниці через інтерфейси операційної системи.

До **плюсів** такого підходу відноситься, також, відсутність необхідності постійного відкату файлової системи і реєстру до початкового стану.

**Недоліки** подібного підходу-програмна складність реалізації максимально точного і повноцінного набору правил, лише частковий відкат змін всередині пісочниці. Так само, як і будь-яка пісочниця, що працює на базі робочої системи, можливий пробій або обхід захищеної середовища і вихід шкідливих кодів в основну, незахищену середу виконання.

DefenseWall, Windows Software Restriction Policy, Limited User Account + ACL.

**Системи, засновані на «можливості»** (capability-based security), також дозволяють обмежувати ресурси програм, в залежності від призначених їм «можливостей».

**Режими використання антивірусних пісочниць.**

1. Режим постійного захисту. При старті процесу, який може бути загрозою для основної системи, він автоматично поміщається в пісочницю.

2. Режим ручного захисту. Користувач самостійно приймає рішення про запуск того або іншого додатка всередині пісочниці.

**Онлайн-пісочниці**

У мережі є ряд проектів з інформаційної безпеки, які реалізують свої рішення в якості окремо працюючих віртуалізованих систем для виконання коду з подальшим аналізом застосунків різних модифікацій (наприклад, https://cuckoo.cert.ee/). Як правило, ці проекти мають онлайн-версії таких систем з безкоштовним використанням. Можливо завантажити підозрілий файл і через деякий час отримати повну інформацію про те, що він робить, будучи запущеним в системі.

**Офлайн-пісочниці**

Використання офлайн пісочниць можливий при умові коли:

- Доступ до інтернет поганий.
- Он-лайн пісочниці в даний момент перевантажені, а виконання аналізу критично по часу.
- Виконання в онлайн пісочницях блокується досліджуваним файлом.
- Необхідна більш тонка настройка режиму виконання файлу при аналізі, наприклад - збільшення часу затримки з моменту запуску.

**Cuckoo Sandbox (**https://cuckoosandbox.org/**) -** система для автоматичного дослідження шкідливого ПО, експлойтів, шкідливих скриптів, документів, архівів і посилань. Система здатна перевіряти документи pdf, doc, xls, rtf, скрипти Python, JS, DLL бібліотеки, бінарники, jar і багато іншого.

У спеціально підготовленій віртуальній системі встановлюється Python 2.7, додається на автозавантаження агент Cuckoo, який буде взаємодіяти з пісочницею, спеціальним чином налаштовуються інтерфейси мережі, для перехоплення і подальшого аналізу мережевого трафіку. Після всіх маніпуляцій робиться знімок файлової системи, він же Snapshot. Пісочниця завантажує тестований файл, визначає його тип і відповідно до типу файлу виробляє необхідні маніпуляції, всі зміни всередині пісочниці фіксуються в звіт. Після роботи система відновлює снапшот і віртуальна система повертається до свого початкового стану.

**Cuckoo Sandbox здатна на**:

- Моніторинг викликів win32 API функцій.
- Дамп мережевої активності.
- Дамп і аналіз пам'яті.
- Створення скріншотів в ході виконання аналізу.
- Збереження копій всіх створених файлів і завантажених в процесі перевірки.
- Трасування інструкцій, виконуваних шкідливим процесом.
- Створення зручного звіту в json, mmdef, maec, html-форматах.
- Абсолютна ізольованість середовища, в якій здійснюється запуск шкідливих програм.

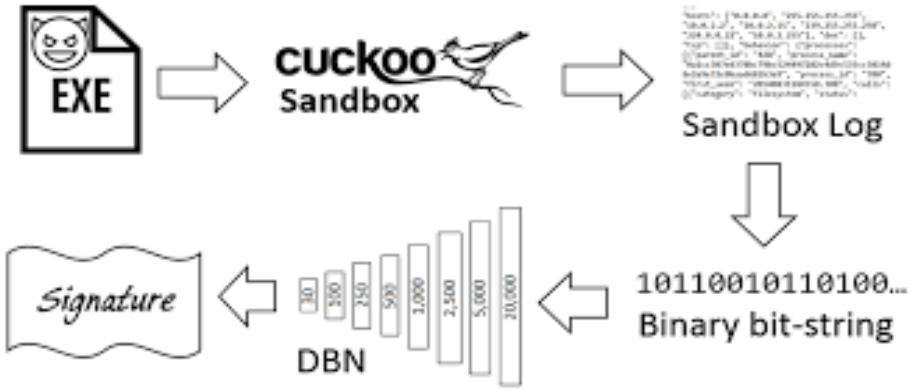



Рисунок 4.15 – Порядок аналізу файлів за допомогою Cuckoo Sandbox

4.6 Системи управління інформацією та подіями безпеки SIEM

Моніторинг систем та мереж відіграє вирішальну роль в захисті від кібератак. Проте природа кіберзлочинності швидко змінюється, через що деякі атаки часто залишаються непоміченими. Об'єднання даних з різних джерел та кореляція між різними подіями, а також здатність зберігати ці дані протягом тривалого періоду часу, стало критично необхідним для забезпечення моніторингу систем та мереж. Також зростання числа кібератак потребує від організацій набору засобів контролю безпеки, які повинні забезпечувати постійний моніторинг. Дані задачі можуть вирішити SIEM системи. Програмні продукти та послуги SIEM об'єднують в собі декілька різних технологій:

- LMS «Log Management System» - система яка збирає та зберігає файли журналів (операційних систем, додатків та інше) з декількох хостів та систем в одному місці, забезпечуючи централізований доступ до журналів, замість доступу до них з кожної системи окремо.
- SLM/SEM «Security Log/ Event Management» - системи, засновані на активному моніторингу та аналізі включаючи візуалізацію даних та оповіщення.
- SIM «Security Information Management» - система, призначена для збору та управління даними, які відносяться до даних безпеки, з декількох джерел.
- SEC «Security Event Correlation» - система визначення переліку подій, які потребують вивчення.

**SIEM** (Security Information and Event Management) - технологія яка в собі містить всі вище-перелічені функції. Ця система автоматично збирає та обробляє інформацію з розподілених джерел, зберігає її в одному централізованому місці, знаходить співвідношення між різними подіями та виробляє сповіщення та звіти виходячи з цієї інформації. **Основними можливостями технології SIEM** є:

- **Агрегація даних**: Управління журналами об'єднаних даних зібраних з багатьох джерел, включаючи мережу, сервери, бази даних, додатки, представляючи можливість консолідувати дані, які відслідковуються, щоб уникнути пропуску важливих подій.

- **Кореляція**: забезпечує пошук загальних атрибутів та пов’язання події в осмислені зв’язки. Ця технологія надає можливість робити різні методи кореляції для інтеграції різних джерел, щоб перетворити дані в корисну інформацію
- **Сповіщення**: яке надає можливість автоматизованого аналізу взаємопов’язаних подій та створення сповіщень, щоб повідомити адміністратора про проблеми. Оповіщення може бути відправлені на панель інструментів або сторонніми каналами, наприклад по електронній пошті.
- **Панелі моніторингу**: інструменти які можуть збирати дані про події і перетворювати їх в інформаційні діаграми, щоб допомогти побачити шаблони або ідентифікувати дії, які не утворюють стандартний шаблон.
- **Сумісність**: застосування додатків для автоматизації збору даних, формування звітності для адаптації агрегуючих даних до існуючих процесів управління інформаційною безпекою та аудиту.
- **Зберігання**: використання довгострокового зберігання історичних даних для полегшення кореляції даних з плином часу і для забезпечення зберігання, необхідного для відповідності вимогам.
- **Криміналістичний аналіз**: можливість пошуку по журналам на різних вузлах і періодах часу на основі певних критеріїв.

SIEM системи більш чітко та детально бачать що відбувається в мережі, ніж це може забезпечити будь-яка інша система управління безпекою або джерело інформації. При правильному використанні система класу SIEM забезпечує наступні **переваги**:

- Зниження ризиків виникнення загроз інформаційної безпеки за рахунок оперативного виявлення і реагування.
- Скорочення витрат та підвищення продуктивності роботи фахівців інформаційної безпеки.
- Автоматизацію процесу оцінки відповідності вимогам вітчизняних і міжнародних стандартів (FISMA, PCI DSS, ISO 27001)
- Контроль стану IT-інфраструктури.
- Оцінку ефективності наявних засобів захисту за рахунок виявлення причин виникнення кіберінцидентів.
- Централізоване зберігання інформації про події та інциденти інформаційної безпеки, з можливістю їх подальшого аналізу.

Для сприяння ефективного та всебічного функціонування SIEM необхідно приділяти увагу її розробці, тобто її архітектурним технологіям та процесам. Проте навіть найкращий моніторинг та методологія можуть не гарантувати виявлення фактичної APT атаки (просунутої багатокрокової атаки). Замість цього використовують більш всебічний аналіз та кореляцію, щоб виявити поведінку, характерну для атак, пов’язаних з APT, бокового переміщення та ексфільтрації даних.

**Кореляція** – визначення залежності між будь-якими двома випадковими величинами або біваріантними даним. Існують сигнатурні (rule based) та без сигнатурні методи кореляції. Сигнатурні – ті, в яких необхідно добавити деякі правила визначення інциденту. Без сигнатурні – беруть інформацію про інциденти від вендорів. Існує велика кількість методів кореляції, проте використовують лише наступні:

- Statistical – важкий без сигнатурний метод кореляції подій, заснований на вимірюванні двох або більше змінних та обчисленні степені статистичного зв'язку між ними.
- RBR Rule-based (pattern based) (HP ECS, IMPACT, RuleCore) – метод, в якому взаємозв'язок між подіями визначається аналітиками в зарання заданих специфічних правилах.
- CBR Codebook (case) based (SMARTS). Кореляція відбувається по прохідним векторам з попередньо заданої матриці подій.
- MBR model based reasoning (дуже великий MMTR) – метод заснований на абстракції об'єктів та нагляду за ними в рамках моделі.
- Graph based. Кореляція полягає в пошуку залежності між системними компонентами в графічному представлені (network devices, host, services) та побудові графіку на їх основі. Якщо залежність виявляється, граф використовується для пошуку основної причини виникнення проблеми.
- Neural network based – ідеологічний метод. Нейронна мережа навчається для пошуку аномалій в потоці подій.

Кореляція подій пропонує повний контекст та логічний аналіз через послідовність пов'язаних подій. В результаті аналітики можуть прийняти рішення про подальші дії.

Використання методів кореляції в реагуванні на кібератаки включає в собі наступні переваги:

- Видимість загрози в реальному часі. Активна кореляція та аналіз подій можуть допомогти виявляти загрози в режимі реального часу.
- Чіткість мережевої безпеки. Мережа може контролюватись в будь який час.
- Безперервні звіти про відповідність. Методи кореляції можуть використовуватись для забезпечення постійного моніторинг всіх структур. Потім можливо створювати звіти для чіткого опису загроз та подій, пов'язаних з безпекою, а також кроків, необхідних для запобігання потенційних ризиків.
- Зменшення експлуатаційних витрат. Інструменти кореляції подій автоматизують такий процес, як аналіз великих робочих процесів, щоб зменшити кількість відповідних попереджень. В результаті витрачається менше часу на розбір в ситуації, та більше на виправлення безпосередніх загроз.
- Покращує управління часом. Потребується менше ресурсів завдяки використання інструментів кореляції в Security information and event management

(SIEM).

Кореляційний механізм є невід'ємною частиною функціональних можливостей технологій SIEM, що дозволяє акцентувати увагу тільки на критичних та дійсно важливих загроз, працювати не з подіями, а з інцидентами, своєчасно виявляти аномалії та ризики. Що робить їх невід'ємним інструментом оперативного виявлення та реагування.

Одною з основних задач архітектури SIEM є підтримка та управління змінами конфігурації системи, службами каталогів, оглядом та аудитом журналів, як службових так і привілейованих користувачів з включенням реакцій на інциденти. Крім того, додатки, пов'язані з управлінням ідентифікацією та доступом, повинні регулярно оновлюватись для посилення безпеки системи та усунення зовнішніх загроз. Більш того, архітектура SIEM повинна забезпечувати можливість представлення, аналізу та збору інформації з мережевих та захисних пристроїв. Також слід зазначити особливості виявлення аномалій, виявлення поліморфного коду та атак «нульового дня». Автоматичний синтаксичний аналіз і нормалізація журналу можуть встановити шаблони, які збираються за допомогою візуалізації SIEM з використанням подій безпеки.

Залежно від того, як розроблена SIEM, процес виявлення потенційних загроз може бути як в пакетному, так і в реальному часі, виходячи з того, наскільки важливою є загроза безпеки. SIEM визначає процес, згідно з яким будь-які фірмові інструменти, на яких вона встановлена, можуть мати стандартний набір архітектури проектування. В загальному випадку робочий процес SIEM представлений на рисунку 4.16.


Рисунок 4.16 – Робочий процес SIEM

Основними кроками (рівнями) роботи SIEM є:

- Рівень збору даних (1): перший крок в будь-якій системі моніторингу та оповіщення – отримати дані, які можливо проаналізувати та повідомити, чи є в них будь-яка інформація для запуску оповіщення про загрозу. Основна практика полягає в розгортанні додатків для будь-якої фільтрації та пересилки журналів в будь-яку конкретну систему зберігання. Джерелами даних для SIEM можуть бути брандмауери, IDS/ IPS системи, веб-фільтри, контролери доменів, системи збору журналів тощо. SIEM також можуть бути інтегровані в хмарні сервіси для отримання даних журналу інфраструктури, розгорнутій в хмарі. Події можуть бути відфільтровані на рівні збору даних, так щоб зберігались тільки необхідні дані.
- Рівень зберігання(5): відповідає за збір та фільтрацію подій на основі

визначених правил, а потім зберігає дійсні події в цільовій системі зберігання. Сьогодні найбільш розподіленими і високопродуктивними системами зберігання є такі як HDFS, S3, Kafka та інші.

- Рівень аналізу кореляції та безпеки (2,3): основним кроком у SIEM є створення правил та політик для визначення того, які події відносяться до загроз безпеки. Враховуючи наявні можливості для машинного навчання, можна мати модель для автоматичного виявлення та категоризації подій на події низької, середньої та високої загрози, а потім передавати їх на інший рівень для здійснення відповідних дій.
- Рівень дій та відповідності (4): цей рівень відіграє важливу роль в SIEM, так як він приймає відповідні дії та відправляє оповіщення групі оперативного реагування, щоб сповістити про інцидент безпеки. Система дійсно може бути спроектована так, щоб мати можливість блокувати будь-яку подальшу обробку запитів про кореляцію подій і посилати попередження команді безпеки для розгляду інциденту в пріоритетному порядку. Ці інциденти можуть потім використовуватись для визначення документу відповідності, який може використовуватись, щоб уникнути подібних повідомлень про інциденти в подальшому.

В теорії **принцип роботи SIEM** рішення заключається у наступному: система збирає інформацію, аналізує «на льоту» (генеруючи попередження), складає аналізовані події в бази даних, перевіряє поведінку на підставі попередніх спостережень.

На рис. 4.17 представлена узагальнена схема роботи SIEM системи.

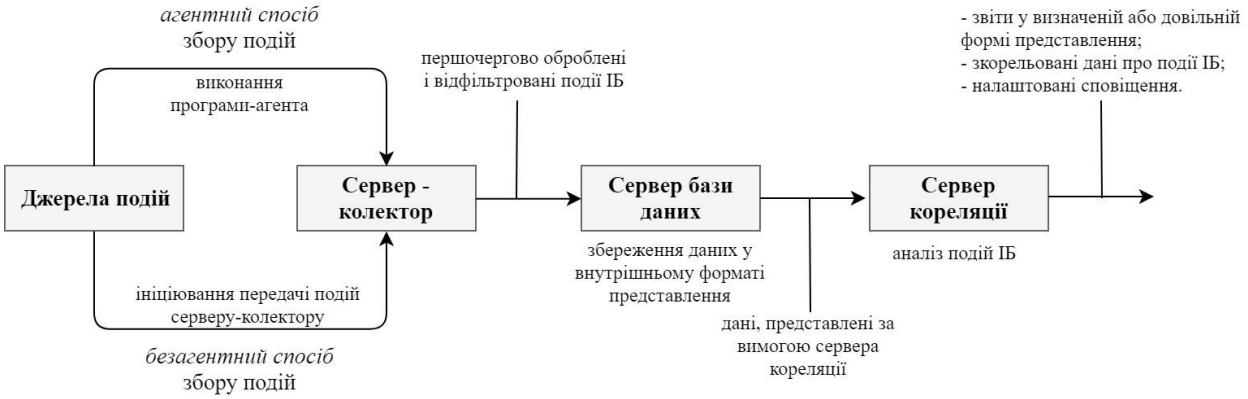



Рисунок 4.17 – Схема роботи SIEM системи

На практиці схема реалізується за допомогою відповідних компонентів:

- агенти (збір даних з різних джерел);
- сервери-колектори (акумуляція інформації, що надійшла від агентів або безпосередньо з джерел);
- сервер баз даних (зберігання інформації);

- сервер кореляції (аналіз інформації).

Збір даних може здійснюватися за допомогою спеціальних агентів, які представляють собою програму, що локально збирає журнали подій і по можливості передає їх на сервер. Для «вичитки» того чи іншого джерела даних агент використовує колектори - бібліотеки для розуміння конкретного журналу подій або системи. Колектори грають важливу роль, так як різні джерела можуть назвати одну і ту ж подію по-своєму. Наприклад, Firewall одного виробника може записувати в звіт deny, іншого discard, третього drop, хоча подія одна і теж. Колектори допомагають привести всі ці події до узагальненого вигляду. Якщо для джерела немає відповідного колектора, події можна спробувати відправляти як SYSLOG (за умови, що джерело вміє це робити). Однак і тут можна зіткнутися з «проблемою синонімів» і необхідністю писати додатковий обробник для приведення даних в єдиний формат.

Інформацію можна збирати віддалено за допомогою протоколів NetBIOS, RPC, TFTP, FTP. Однак в цьому випадку може виникнути проблема з навантаженням на мережу, так як частина систем дозволяє передавати тільки журнал цілком, а не «свіжі» записи.

SIEM системи використовують інформацію з таких джерел:

- системи аутентифікації і системи контролю і управління доступом (Authentification, Access Control);
- антивірусні засоби;
- міжмережеві екрани;
- системи виявлення / запобігання вторгнень (IDS/IPS);
- системи веб-фільтрації;
- активні мережеві пристрої;
- системні журнали подій ІБ серверів і робочих станцій користувачів;
- журнали аудиту систем управління базами даних;
- ключові корпоративні ресурси: поштові сервери, файлообмінні сервери, CRM- і ERP-системи;
- інші бізнес-додатки відповідно до вимог ІБ компаній і стандартів.

Основні методи і способи виявлення інцидентів з допомогою SIEM рішень:

- візуальний (інформаційні панелі і події);
- правила кореляції в режимі реального часу;
- звіти;
- аналітика.

Отримавши інформацію, система може її проаналізувати. В основі аналізу лежить математика і статистика. Але відправною точкою служать вручну задані правила. Наприклад, одноразова подія «login failed» нічого не означає, в той час як три і більше таких події від одного облікового запису вже можуть свідчити про спробу підбору пароля.

У найпростішому випадку в SIEM системах правила представлені в форматі RBR (Rule Based Reasoning) і містять набір умов, тригери, лічильники, сценарій дій. Наприклад, враховувати параметри віддаленості двох останніх точок використання банківської карти за невеликий інтервал часу: якщо о 17:00 її використовували для оплати кави в Києві, а через 10 хвилин намагаються зняти денний ліміт в Гонконзі, то на обличчя – спроба шахрайства.

Можна зробити висновок, що SIEM системи здатні виявляти:

- мережеві атаки;
- спроби несанкціонованого доступу до конфіденційної інформації;
- шахрайство;
- помилки і збої в роботі інформаційних систем;
- вразливості;
- помилки конфігурацій в засобах захисту та інформаційних системах;
- цільові атаки (APT).

4.7 Стек ELK як система централізованого зберігання журналів

**Logstash** – відповідає за обробка вхідних логів;

**Elasticsearch** – відповідає за зберігання логів;

**Kibana** – представляє собою веб-інтерфейс;

**Scirius Community Edition** – веб-додаток для управління набором правил Suricata.

**Logstash** - це потужний інструмент для збору, систематизації і аналізу логів, який допомагає отримати загальне уявлення про середовище, а також вчасно виявити проблеми. Один із способів підвищити ефективність інсталювання Logstash - це використовувати фільтри для збору логів додатків і структурування даних, завдяки чому дані можна легко запросити і проаналізувати.

**Elasticsearch** представляє собою пошуковий двигун з JSON REST API, який використовує Lucene і написаний на Java. Подібні двигуни використовуються при складному пошуку по базі документів. Наприклад, пошук з урахуванням морфології мови або пошук по географічним координатами.

**Kibana** представляє собою плагін для Elasticsearch з відкритим вихідним кодом, основною задачею якого є візуалізації даних. Він забезпечує можливості візуалізації поверх контенту індексується на кластері Elasticsearch. Користувачі можуть створювати гістограми, діаграми розсіювання, або кругові діаграми та карти при великих обсягах даних.

**Узагальнена схема роботи стека** зображена на рисунку 4.18.


Рисунок 4.18 – Узагальнена схема роботи стека ELK

**Beats** - спеціальний клієнт, що забезпечує збір і відправку даних в Logstash.

Beats дозволяє:

- Проводити попередню фільтрацію;
- Відправляти дані з джерела;
- Доставка і централізація в Elasticsearch;
- Доставка у Logstash для перетворення та розбору;
- Доставка у Elastic Cloud;
- Libbeat: API для побудови користувацьких beats;
- 30+ спільнот Beats.

Крім цього, також доступні:

- Filebeat (збір даних з журналів);
- Winlogbeat (збір подій Windows Event Logs);
- Metricbeat (метрики);
- Packetbeat (мережева інформація);
- Heartbeat (uptime).

Всі клієнти доступні для платформ Linux, Windows і Mac OSX.

Всі налаштування проводяться в файлі конфігурації **/etc/filebeat/filebeat.yml**, після установки вже є готовий шаблон із мінімальними настройками. Там же лежить файл filebeat.full.yml (в новій версії filebeat.reference.yml), в якому прописані всі установки, доступні для Filebeat.

У файлі для Filebeat потрібно вказати, **що брати і куди відправляти**. При цьому вказуються тип документа (input_type, log_type і document_type, він потім буде доступний в пошуку) і специфічні параметри підключення. За замовчуванням Filebeat збирає всі файли в /var/log/*.log, тобто всі файли в каталозі /var/log, що закінчуються на .log.

```
filebeat.prospectors:
- type: log
      paths:
        - /var/log/messages
        - /var/log/*.log
output:
```

```
logstash:
  hosts: ['logstash.host:5044']
```

**Logstash**

Приймає потік логів від додатків і устаткування, обробляє їх запрограмованим чином і передає в ElasticSearch.

Дозволяє:

- Введення даних всіх форм, розмірів та джерел;
- Розбір та динамічне перетворення даних;
- Транспорт даних на будь-який вихід;
- Захист та шифрування введених даних;
- Створення власного конвеєру;
- Більше 200 плагінів.

Основна конфігурація Logstash відбувається в командному рядку в директорії **/etc/logstash/conf.d/**. Саме тут описується, що приймати, як це обробити і зберегти.

На рисунку 4.19 показано порядок синтаксичного аналізу (парсінгу) логів за допомогою Lostash. Він повністю відображає структуру конфігураційного файлу. Формат конфігураційного файлу Logstash'a простий і зрозумілий. Він складається з трьох частин:

```
input {
  …
}
filter {
  …
}
output {
  …
}
```

Вхідних, фільтруючих і вихідних блоків може бути будь-яка кількість. Все залежить від потреб і можливостей апартаного забезпечення. Порожні рядки і рядки, що починаються з # Logstash ігнорує. Так що коментування конфігураційних файлів не викликає жодних проблем.

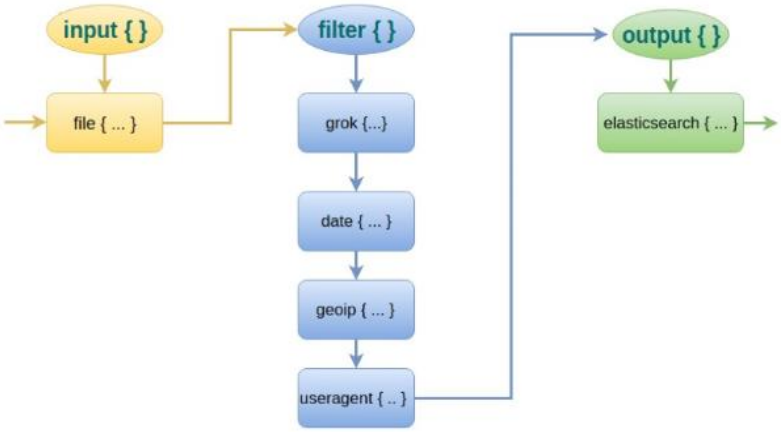



Рисунок 4.19 – Синтаксичний аналіз (парсінг) логів за допомогою Logstash

**Input**. Даний метод є вхідною точкою для логів. В ньому визначають по яким каналам логи будуть потропляти в Logstash.

**Filter**. В даному блоці налаштовуються основні маніпуляції з логами. Це може бути і розбивка по key = value, і видалення непотрібних параметрів, і заміна наявних значень, і використання geoip або DNS запитів для ip-адрес або назв хостів. Для маніпуляції з логами використовуються фільтри, які умовно залежно від особливостей події. Нижче наведено деякі додатки фільтрів:

aggregate - Агрегує інформацію про декілька подій, що починаються з одного завдання;

alter - Здійснює загальну зміну полів, які фільтр mutate не обробляє;

date - Розбирає дати з полів, які використовуються як часова мітка Logstash для події;

geoip - Додає географічну інформацію про IP-адресу;

grok - Розбирає неструктуровані дані подій на поля;

useragent - Розбирає рядки агента користувача на поля,

та багато інших.

**Output**. Назва цього блоку/методу говорить сама за себе - в ньому вказуються настройки для вихідних повідомлень. Аналогічно попереднім блокам, тут можна вказувати будь-яку кількість вихідних подблоков. Logstash пропонує декілька вихідних плагінів для зберігання відфільтрованих подій журналу в різних системах зберігання та пошуку.

Logstash може зберігати відфільтровані журнали у файлі, Elasticsearch Engine, stdout, AWS CloudWatch тощо. Мережеві протоколи, такі як TCP, UDP, Websocket, можуть також використовуватися в Logstash для передачі подій журналу у віддалені системи зберігання. У стеку ELK користувачі використовують двигун Elasticsearch для зберігання подій журналу.

На рисунку 4.20 показано приклад конфігурації Logstash (журнал доступу до веб серверу Apache).

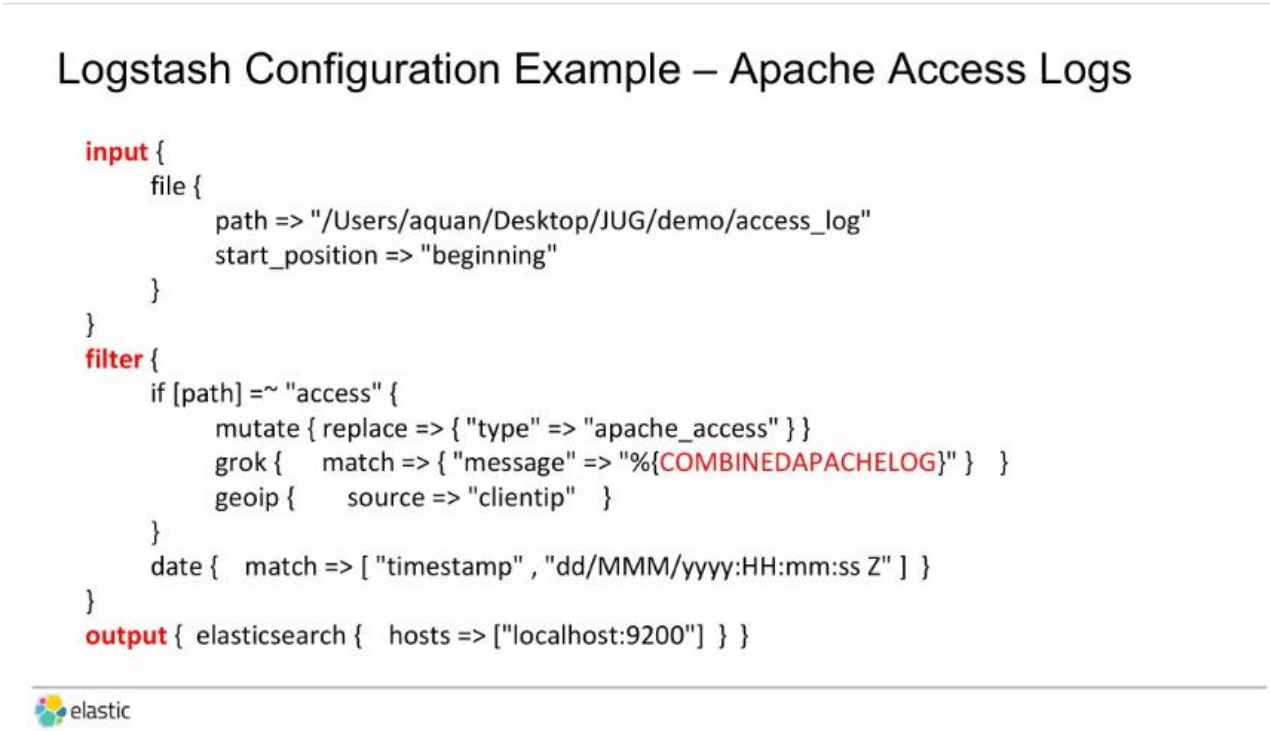



Рисунок 4.20 – Приклад конфігурації Logstash (журнал доступу до веб серверу Apache)

Також наведемо приклда конфігураційного файлу, який відповідає за прийом логів по протоколу syslog в Logstash та передачу даних з Logstash в ElasticSearch.

```
input {
  tcp {
    type => "syslog"
    port => 514
  }
}
input {
  udp {
    type => "syslog"
    port => 514
  }
}
output {
      elasticsearch {
```

```
        hosts => ["http://localhost:9200"]
        index => "logstash-example-%{+YYYY.MM.dd}" }
}
```

В наступному прикладі вказується порт, на якому будемо приймати підключення Filebeat.

```
$ sudo nano /etc/logstash/conf.d/01-input.conf
input {
input {
  beats {
    port => 5044
    ssl => true
    ssl_certificate => "/etc/ssl/logstash/logstash.crt"
    ssl_key => "/etc/ssl/logstash/logstash.key"
  }
}
```

**Elasticsearch** - це розподілена і масштабована пошукова система, що працює в режимі реального часу, яка забезпечує повнотекстовий і структурований пошук, а також аналітику. Вона зазвичай використовується для індексації і пошуку у великих обсягах даних логов, але також може використовуватися для пошуку різних типів документів.

Властивості:

- Розподілений, масштабований;
- Висока доступність;
- Багаторічна оренда;
- Підходить для розробників;
- Повнотекстовий пошук у режимі реального часу;
- Агрегації

Основні поняття, що використовуються Elasticsearch по суті відповідають поняттям в структурованих базах даних типу SQL, але різняться за назвою. Так можна провести наступне порівняння:

| **SQL** | **Elasticsearch** |
|---|---|
| База даних | Індекс |
| Таблиця | Тип |
| Рядок | Документ |

| Колонка | Поле |
|---|---|

У Elasticsearch дані зберігаються у вигляді документів JSON (Javascript Object Notation). Більшість сховищ даних NoSQL використовують JSON для зберігання своїх даних.

4.8 Система управління інформацією та подіями безпеки SIEM SPLUNK

Платформа безпеки Splunk відповідає критеріям для сучасних рішень SIEM. Вона також забезпечує аналітичні можливості та візуальне розуміння, які допомагають командам реагування прийняти більш швидке та розумне рішення в області безпеки.

Splunk пропонує локальні, хмарні та гібридні розгортання. Організації можуть вирішити свої основні випадки використання SIEM, користуючись Splunk Enterprise, який являється основною платформою, вона забезпечує збір, індексацію, пошук та можливість звітування.

Splunk Enterprise (далі Splunk) встановлює на хостовій системі процес splunkd. Splunkd – це розподілений сервер, який здійснює доступ, обробляє та індексує потокові дані. Він також опрацьовує пошукові запити. Splunkd опрацьовує та індексує дані, передаючи їх через серію конвеєрів, кожен з яких складається з серії процесорів. Конвеєри – це окремі потоки в процесі splunkd, кожен з яких налаштований на один фрагмент XML. Процесори – це окремі функції, які взаємодіють на потік даних через “|” (“трубу"). «Труби» можуть передавати друг другу дані через черги.

Основний каталог програмного забезпечення Splunk Enterprise $Splunk_HOME, який містить інформацію по конфігурації Splunk у відповідних директоріях і файлах, ієрархічно структурований. Ці файли ідентифікуються розширенням .conf та включають інформацію різних варіантів конфігурацій. Серед таких варіантів є:

- системні налаштування;
- інформацію про автентифікацію та авторизацію;
- карту індексів та їх налаштування;
- конфігурацію кластерів та серверу розгортання;
- об'єкти даних та збережені пошуки.

Більшість файлів конфігурації розміщується в директорії $Splunk_HOME/etc/system/default за замовчуванням.

Перелік основних конфігураційних файлів включає:

- alert_actions.conf (створення сповіщень);
- audit.conf (конфігурування аудиту та хешування подій);
- authentication.conf (вибір аутентифікації: стандартна від Splunk або LDAP);

- datamodels.conf (пари «атрибут»:«значення» для налаштування моделей даних);
- eventtypes.conf (визначення типів подій);
- indexes.conf (корегування конфігурації індексів);
- inputs.conf (визначення джерел надходження даних);
- limits.conf (встановлення обмежень на команди пошуку);
- outputs.conf (визначення параметрів надсилання даних);
- passwords.conf (відомості про обліковий запис додатку);
- server.conf (увімкнення SSL для Back-End Splunk (зв'язок між splunkd і Splunk Web), уточнення місця розміщення сертифікатів);
- sourcetypes.conf (правила для визначення типу даних, що надходять із різних джерел);
- web.conf (конфігурування Splunk Web, увімкнення HTTPS).

Структура файлів конфігурації визначається описом однієї або декількох станів.

Узагальнений вигляд файлу конфігурації:

[stanza1_header]
<attribute1> = <val1>
# коментар
<attribute2> = <val2>
...
[stanza2_header]
<attribute1> = <val1>
<attribute2> = <val2>
...

Станза – секція конфігураційного файлу. Опис станзи починається з текстового ряду, що містяться в дужках [] , і включає один або декілька параметрів конфігурації, визначених парою ключ / значення.

Архітектура Splunk зображена на рисунку 4.21.

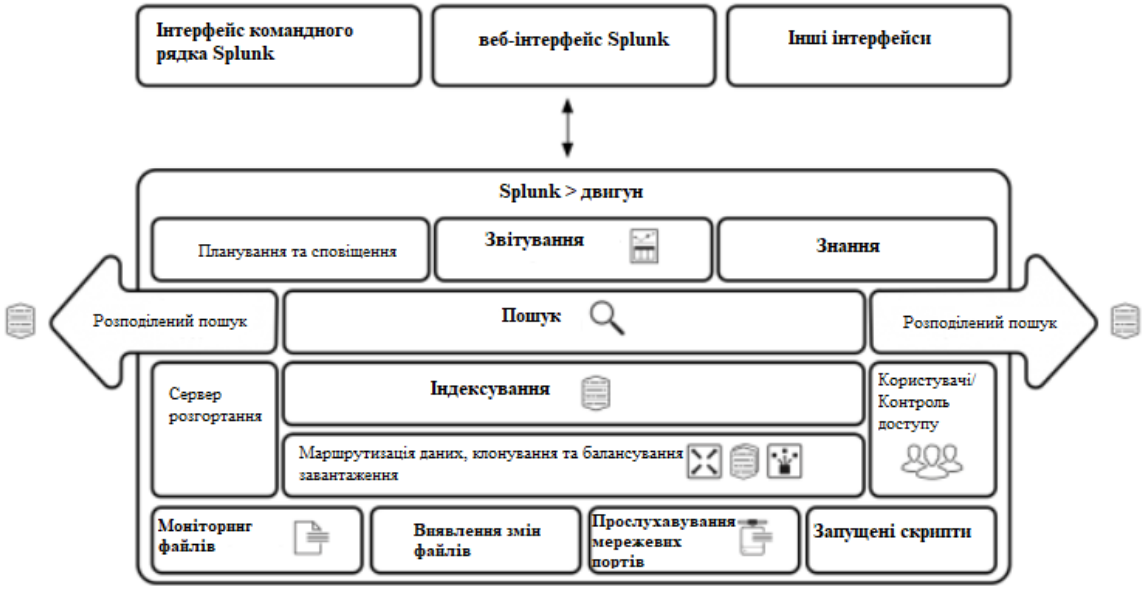



Рисунок 4.21 – Архітектура Splunk Enterprise

Процес splunkd запускає веб-сервер через порт 8089, з включеним по замовчуванню протоколом SSL/HTTPS. Що дозволяє віддалено керувати «двигуном» Splunk через веб-інтерфейс.

Двигун Splunk виконує наступні функції: індексування, пошук (з можливістю розподіленого пошуку), звітування, планування та оповіщення. Він забезпечує маршрутизацію даних їх клонування та балансування навантаження.

Дані в Splunk потрапляють через універсальний експедитор (рис. 4.22) (Splunk universal forwarder).

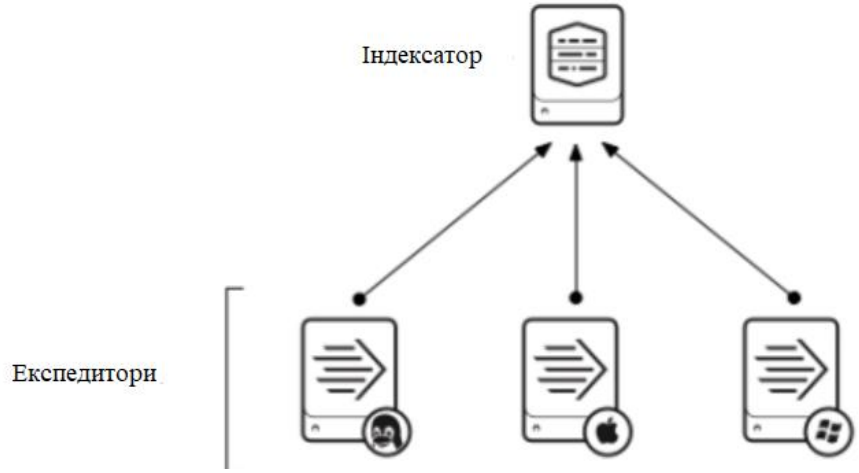



Рисунок 4.22 – Пересилання даних на сервер Splunk

Universal forwarder збирає дані з джерел даних або іншого експедитора та відправляє їх на сервер Splunk. Його головною перевагою є можливість встановлення на різних обчислювальних платформах. Він включає в себе тільки основні компоненти для пересилання даних на іншу платформу Splunk. Хоч в

нього і немає веб-інтерфейса, його можливо налаштовувати, керувати та масштабувати редагуючи конфігураційний файл.

Машинні дані проходять декілька етапів обробки для перетворення у події, що піддаються пошуку. Цей процес називається конвеєром даних (data pipeline) і включає чотирьохкрокову послідовність:

- прийом;
- аналіз;
- індексування;
- пошук.

На рис. 4.23 схематично відображено роботу конвеєру даних.

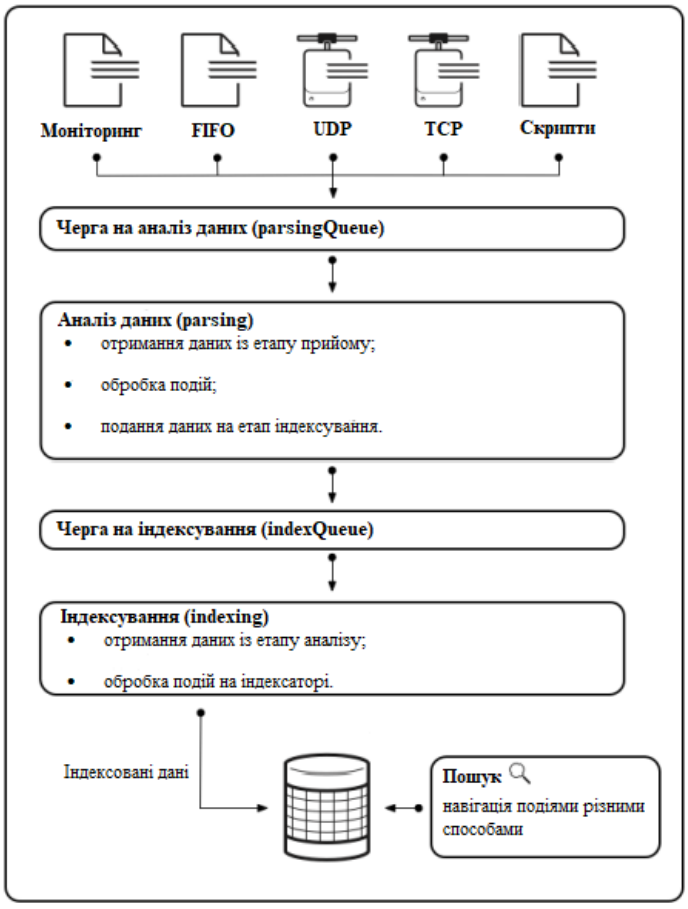



Рисунок 4.23 – Конвеєр даних Splunk Enterprise

Етап прийому включає:

- отримання машинних даних із джерела прийому;
- їх розбивка на 64К блоки;
- присвоєння кожному блоку метаданих;
- подання даних на етап аналізу.

Splunk Enterprise працює з великим розмаїттям даних, які надходять з журналів подій, веб-журналів, мережевого трафіку, системних метрик, архівних файлів тощо.

Чотири основних джерела прийому машинних даних:

- файли та каталоги;
- мережеві події (Splunk може індексувати дані з будь-якого мережевого порту та SNMP події);
- джерела Windows (журнал подій Windows, реєстр Windows, WMI, Active Directory);
- інші джерела (наприклад, черги FIFO та виконувані скрипти для отримання даних з налаштованих додатків та інших віддалених машин).

Прийом даних здійснюється на індексаторі або форвардері.

Клієнти які використовують Splunk Enterprise можуть будувати власні правила кореляції в реальному часі та інформаційні панелі. Для забезпечення цих можливостей була розроблена **Search Processing Language (SPL).**

SPL – охоплює всі команди пошуку, їх функції та аргументи. Команди пошуку повідомляють програмному забезпеченню Splunk, що робити з подією, яка була передана з індексаторів.

По мірі пошуку розпізнається більше інформації, яка може бути корисна в якості полів для пошуку. Тому можливо налаштувати програмне забезпечення Splunk на розпізнавання цих нових полів при індексації нових даних або створити нові поля які потрібно шукати. Цей збір даних допомагає створювати більш ефективні пошукові запити. В Splunk існують наступні типи пошуків:

- Пошук неопрацьованих подій – це пошук, який просто витягує події з індексатора або індексаторів та зазвичай використовується коли потрібно проаналізувати проблему. Прикладами таких пошуків можуть бути: перевірка коду помилок, кореляція подій, розслідування проблем безпеки та аналіз збоїв. Ці пошуки зазвичай не включають в себе команди пошуку, а результати представляють собою список неопрацьованих подій.
- Пошук з перетворенням – це пошук, який виконує будь-який тип статичного розрахунку по набору результатів. Це виконує пошук, коли спочатку виймаються події з індексатора, а потім передають цю подію в одну, або декілька команд пошуку. Ці пошуки завжди потребуватимуть полів та хоча б одну статичну команду. Прикладом таких пошуків можуть бути: отримання щоденної кількості подій помилок, підрахунок кількості входів визначеного користувача.
- Розріджений пошук – це пошук, який шукає одну подію, яка рідко зустрічається в великому наборі даних. Наприклад: пошук визначеної та унікальної IP-адреси або коду помилки.
- Щільний пошук – це пошук, який переглядає та сповіщає про велику кількість подій. Наприклад: підрахунок кількості помилок, які трапились, або пошук всіх подій, які трапились на визначеному хості.

Для ефективного використання SPL існує шість широких категорій для пошукових команд:

- Розподілена потокова передача;
- Централізована потокова передача;
- Перетворююча;
- Генеруючі;
- Оркестровки;
- Обробка набору даних;

Ця класифікація не являється взаємовиключною. Деякі команди вписуються тільки в одну категорію, а інші можуть належати декільком категоріям. Наприклад команда може бути потоковою, а також генеруючою.

Команда потокової передачі працює на кожну подію, оскільки вона повертається пошуком. По суті, одна подія і одна (або ні) подія (рис. 4.24).

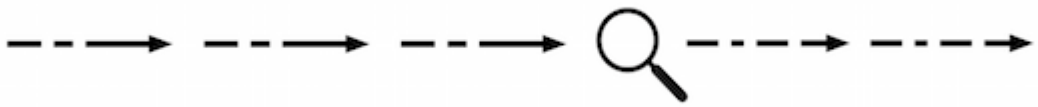


Рисунок 4.24 – Потокова передача

Команда без потоку вимагає подій від всіх індексаторів, перш ніж вона зможе працювати над усім набором подій (рис. 4.25). Багато перетворюючих команд, є не потоковими командами. Також є кілька команд, які не є перетворюючими, але є потоковими. Ці команди що не є перетворюючими, не транслюються, називаються командами на основі подій, що не є потоковими.

Рисунок 4.25 – Без-потокова передача

Команди, що не передаються в потоковому режимі, примушують весь набір подій виконуватись в пошуковій частині. Це вимагає руху великої кількості даних та втрати паралелізму.

В таблиці 4.1 описані розбіжності в обробці між деякими типами команд

Таблиця 4.1

Відмінності між типами команд

| | Розподілена потокова | Централізоване потокове | Обробка даних безпотокової передачі | Перетворююча |
|---|---|---|---|---|
| Може працювати на індексаторах | Так | Ні | Ні | Ні |
| Може вивести до кінцевого висновку | Так | Так | Ні | Ні |
| Вивід подій, якщо на вході подія | Так | Так | Так | Ні |

Коли команда виконується, вона виводить або подію, або результат в залежності від типу команди. Наприклад коли запускається команда сортування, ввід – подія, в вивід – подія в вказаному порядку сортування. Проте, перетворюючі команди не виводять подій. Перетворюючі команди виводять результати. Наприклад, команда stats виводить таблицю результатів розрахунку. Події, які використовуються для розрахунку цих результатів, більш не доступні. Після запуску команди перетворення, можливо запустити команду, яка очікує подію в якості вводу.

Команди обробки даних – це не потокові команди, яким потрібен весь набір даних, перш ніж команда можу бути запущена. Ці команди не являються перетворюючими, не розповсюджуваними, не потоковими і не команди оркестровки.

Потокова команда діє для кожної події, яка повертається пошуком. Для розподіленої потокової передачі порядок подій неважливий. Поширювана потокова команда – це команда яка може бути запущена на індексаторі, що покращить час обробки. Інші команди в пошуку визначають, чи виконується команда розподіленої потокової передачі на індексаторі:

- якщо на індексаторі можливо виконати всі команди перед командою поширюваної потокової передачі, то розповсюджувана потокова команда виконується на індексаторі;
- якщо будь-яка команда перед командою поширюваної потокової передачі повинна бути запущена в пошуковій частині, решта команд в пошуку повинні бути запущені в пошуковій частині. Коли обробка пошуку переходить до

пошукової частини, вона не може повернутись до індексатора.

Розподілені потокові команди можуть застосовуватись до підмножин індексованих даних паралельно.

Для централізованих потокових команд порядок подій не має значення. Централізована потокова команда виконує перетворення до кожної події, поверненій пошуком. Але на відмінну від розподілених потокових команд, централізована потокова команда працює тільки на пошуковій частині.

Команда перетворення впорядковує результати пошуку в таблицю даних. Ці команди “перетворюють” вказану комірку значення для кожної події в числові значення, які програмне забезпечення Splunk може використовувати для статистичних цілей. Перетворюючі команди не потокові.

Команди генерації витягує інформацію з індексаторів, без будь-яких перетворень. Більшість таких команд призначені для створення звітів. В залежності від типу команди результати повертаються в вигляді списку або таблиці.

Команда  оркестровки – це команда, керуюча деякими аспектами обробки пошуку. Це не напряму впливає на кінцевий набір результатів пошуку.

Наприклад можливо виконати команду оркестровки, щоб включити або виключити пошукову оптимізацію, яка допомагає швидше завершити загальний пошук.

Існує декілька команд, яким необхідний весь набір даних для запуску команд. Ці команди називаються – команди обробки набору даних. Ці команди не перетворюються, не розповсюджуються, не потокові та не оркестровки. Прикладами таких команд: сортування, події в деякому режимі кластера.

Всі вище перелічені команди можливо також розбити на функціональні категорії (рис. 4.26)


Рисунок 4.26 – Категорії команд SPL

Пошук складається з серії команд, розділених символом “|”. Перший запис, розділений пробілами після кожного символу “труби” керує виконуваною командою. Окрема частина тексту для кожної команди обробляється в спосіб, специфічний для даної команди.

Зображення проіндексованих даних та процес виконання команди пошуку показане на рисунку 4.27. (Для зразку розглядається наступний пошук sourcetype=syslog ERROR | top user | fields - percent)

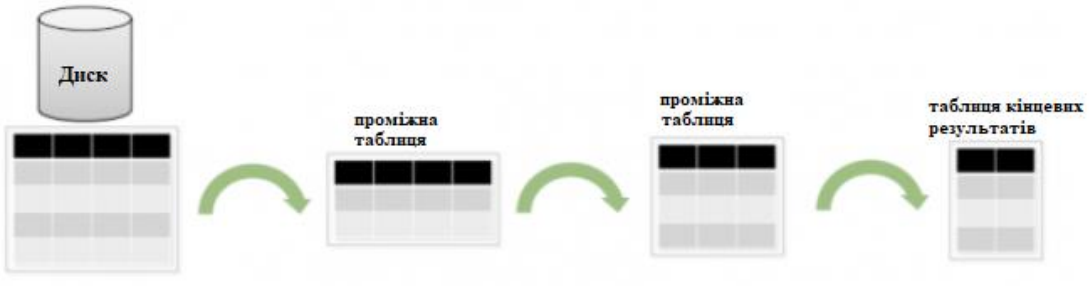



Рисунок 4.27 – Процес виконання команди пошуку

Диск - всі проіндексовані дані, представляють собою таблицю визначеного розміру с колонками, які представляють поля та рядки, які представляють події. Перша проміжна таблиця результатів показує менше рядків – представляє підмножину подій витягнуту з індексатора, відповідного пошукового терміну “sourcetype=syslog ERROR”. Другі проміжні результати в таблиці містять меншу

кількість колонок, які представляють собою результати команди top «top user», яка сумує події в список десяти кращих користувачів та відображає користувача, кількість та відсоток. Потім «поля – відсотки» видаляють колонки, які показують відсотки, так щоб залишитись з меншою таблицею кінцевих результатів.

Стандартно запит SPL (рис. 4.28) можна розділити на декілька етапів: фільтрація та вибір потрібних даних (search and filter), потім створення нових полів на основі вже існуючих (munge), агрегація даних та обчислення статистик (report), та перейменування полів та сортування (cleanup).

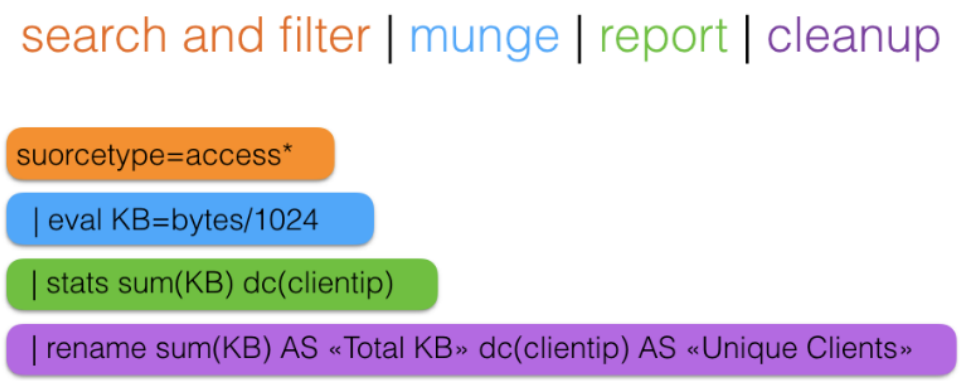



Рисунок 4.28 – SPL запит

Строки пошуку можуть бути можуть бути довгими та складним для читання. Панель пошуку містить функції, які допомагають користувачам читати, аналізувати або інтерпретувати синтаксис SPL. Функція підсвічування синтаксису відображає частини SPL в різних кольорових гамах.

В доповнення до кольорових тем є можливість використовувати автоматичне формування номерів рядків, щоб полегшити читання результатів пошуку.

На рис. 4.29 відображено категоризацію найчастіше вживаних SPL команд.

| Категорія | Опис | Команди |
|---|---|---|
| **Sorting** | Упорядкування подій | sort |
| **Filtering** | Фільтрування результатів для отримання меншої кількості подій | search<br>where<br>dedup<br>head<br>tail |
| **Grouping** | Групування подій за ознаками | transaction |
| **Reporting** | Створення звітів за результатами пошуку | top/rare<br>stats<br>chart<br>timechart |
| **Modifying Adding fields** | Виділення окремих полів, зміна або додавання нових полів для збагачення результатів пошуку | fields<br>replace<br>eval<br>rex<br>lookup |

Рисунок 4.29 – Категоризація SPL команд

За допомогою SPL адміністратори безпеки можуть створювати відповідні пошуки, які можливо відобразити в вигляді інформаційних панелей та звітів, що значно полегшує оперативному виявленню та реагуванню на кіберінциденти.

# 5. ТЕХНОЛОГІЯ ВІРТУАЛЬНИХ ПРИВАТНИХ МЕРЕЖ

## 5.1 Основи криптографії, інфраструктури відкритих ключів та криптопротоколів

Одними із сервісів віртуальних приватних мереж, які будуть розглядатися в цьому розділі, є забезпечення конфіденційності інформації й процеса автентифікації. Ці сервіси забезпечується за допомогою використання криптографічних алгоритмів й криптопротоколів.

Криптографічний алгоритм, або шифр - це математична формула, що описує процеси шифрування і розшифрування. Щоб зашифрувати відкритий текст, криптоалгоритм працює в сполученні з ключем - словом, числом або фразою.

**Криптографічних алгоритмів** існує безліч. В загальному вони поділяються на безключові, одноключові й двоключові алгоритми.

Криптографія з ключем. Алгоритм впливу на передані дані відомий усім стороннім особам, але він залежить від деякого параметра - "ключа", яким володіють лише відправник і одержувач.

- Симетричні криптоалгоритми. Для зашифровки і розшифровки повідомлення використовується один і той же блок інформації (ключ).
- Асиметричні криптоалгоритми. Алгоритм такий, що для зашифровки повідомлення використовується один ( «відкритий») ключ, відомий усім бажаючим, а для розшифровки - інший ( «закритий»), який існує тільки в одержувача.

В залежності від кількості ключів, які застосовуються у конкретному алгоритмі:

- Безключові КА – не використовують в обчисленнях ніяких ключів;
- Одноключові КА – працюють з одним додатковим ключовим параметром (якимсь таємним ключем);
- Двухключові КА – на різних стадіях роботи в них застосовуються два ключових параметри: секретний та відкритий ключі.

В задежності від характеру впливів, що виробляються над даними, алгоритми підрозділяються на:

- Перестановочні - Блоки інформації (байти, біти, більші одиниці) не змінюються самі по собі, але змінюється їх порядок проходження, що робить інформацію недоступною сторонньому спостерігачеві.
- Підстановочні - Самі блоки інформації змінюються за законами криптоалгоритму. Переважна більшість сучасних алгоритмів належить цій групі.

Залежно від розміру блоку інформації криптоалгоритми поділяються на:

- Потокові шифри - Одиницею кодування є один біт. Результат кодування не залежить від минулого раніше вхідного потоку. Схема застосовується в системах передачі потоків інформації, тобто в тих випадках, коли передача інформації починається і закінчується в довільні моменти часу і

може випадково перериватися. Найбільш поширеними представниками потокових шифрів являються скремблери.

- Блочні шифри - Одиницею кодування є блок з декількох байтів (в даний час 4-32). Результат кодування залежить від усіх вихідних байтів цього блоку. Схема застосовується при пакетній передачі інформації та кодування файлів.

Хешування - (англ. Hashing) - перетворення вхідного масиву даних довільної довжини в вихідний бітовий рядок фіксованої довжини. Такі перетворення також називаються хеш-функціями або функціями згортки, а їх результати називають хешем, хеш-кодом або дайджестом повідомлення (англ. message digest). Хешування застосовується для порівняння даних: якщо у двох масивів хеш-коди різні, масиви гарантовано розрізняються; якщо однакові - масиви, швидше за все, однакові. У загальному випадку однозначної відповідності між вихідними даними і хеш-кодом немає в силу того, що кількість значень хеш-функцій менше, ніж варіантів вхідного масиву; існує безліч масивів, що дають однакові хеш-коди - так звані колізії. Імовірність виникнення колізій відіграє важливу роль в оцінці якості хеш-функцій. Існує безліч алгоритмів хешування з різними характеристиками (розрядність, обчислювальна складність, крипостійкість і т. п.). Вибір тієї чи іншої хеш-функції визначається специфікою розв'язуваної задачі. Найпростішими прикладами хеш-функцій можуть служити контрольна сума або CRC.

Електроний підпис - електронні дані, які додаються підписувачем до інших електронних даних або логічно з ними пов’язуються і використовуються ним як підпис. Одним із видів електронного підпису є Електроонний цифровиий пі́дпис (ЕЦП) (англ. digital signature). Він отримується за результатом криптографічного перетворення набору електронних даних, який додається до цього набору або логічно з ним поєднується і дає змогу підтвердити його цілісність та ідентифікувати підписувача. Електронний цифровий підпис накладається за допомогою особистого ключа та перевіряється за допомогою відкритого ключа.


Рисунок 5.1 – Класифікація криптографічних алгоритмів

**Управління ключами.**

Крім вибору підходящої для конкретної ІС криптографічного системи, важлива проблема - управління ключами. Як би не була складна і надійна сама криптосистема, вона заснована на використанні ключів. Якщо для забезпечення конфіденційного обміну інформацією між двома користувачами процес обміну ключами тривіальний, то в ІС, де кількість користувачів становить десятки і сотні управління ключами - серйозна проблема. Під ключовою інформацією розуміється сукупність всіх діючих в ІС ключів. Якщо не забезпечено досить надійне управління ключовою інформацією, то заволодівши нею, зловмисник отримує необмежений доступ до всієї інформації. Управління ключами - інформаційний процес, що включає в себе три елементи:

- генерацію ключів;
- накопичення ключів;
- розподіл ключів.

**Генерація ключів**. На самому початку розмови про криптографічні методи було сказано, що не варто використовувати невипадкові ключі з метою легкості їх запам'ятовування. У серйозних ІС використовуються спеціальні апаратні і програмні методи генерації випадкових ключів. Як правило використовують датчики ПВЧ (псевдовипадкових чисел). Однак ступінь випадковості їх генерації повинен бути достатньо високим. Ідеальним генераторами є пристрої на основі «натуральних» випадкових процесів. Наприклад, випадковим математичним об'єктом є десяткові знаки ірраціональних чисел, які обчислюються за допомогою стандартних математичних методів.

**Накопичення ключів**. Під накопиченням ключів розуміється організація їх зберігання, обліку та видалення. Оскільки ключ є найпривабливішим для зловмисника об'єктом, який відкриває йому шлях до конфіденційної інформації, то питанням накопичення ключів слід приділяти особливу увагу. Секретні ключі ніколи не повинні записуватися в явному вигляді на носії, який може бути зчитаний або скопійований. У досить складній ІС один користувач може працювати з великим об'ємом ключової інформації, і іноді навіть виникає необхідність організації міні-баз даних по ключовій інформації. Такі бази даних відповідають за прийняття, зберігання, облік і видалення використовуваних ключів. Отже, кожна інформація про використовувані ключі повинна зберігатися в зашифрованому вигляді. Ключі, що зашифровують ключову інформацію називаються майстер-ключами. Бажано, щоб майстер-ключі кожен користувач знав напам'ять, і не зберігав їх взагалі на будь-яких матеріальних носіях. Дуже важливою умовою безпеки інформації є періодичне оновлення ключової інформації в ІС. При цьому перепризначатися повинні як звичайні ключі, так і майстер-ключі. В особливо відповідальних ІС оновлення ключової інформації бажано робити щодня. Питання оновлення ключової інформації пов'язаний і з третім елементом управління ключами - розподілом ключів.

**Розподіл ключів**. Розподіл ключів - найвідповідальніший процес в управлінні ключами. До нього пред'являються дві вимоги:

- Оперативність і точність розподілу;
- Скритність ключів, що розподіляються.

Останнім часом помітний зсув в бік використання криптосистем з відкритим ключем, в яких проблема розподілу ключів відпадає. Проте розподіл ключової інформації в ІС вимагає нових ефективних рішень. Розподіл ключів між користувачами реалізуються двома різними підходами:

1. Шляхом створення одного чи кількох центрів розподілу ключів. Недолік такого підходу полягає в тому, що в центрі розподілу відомо, кому і які ключі призначені і це дозволяє читати всі повідомлення, що циркулюють в ІС. Можливі зловживання істотно впливають на захист.

2. Прямий обмін ключами між користувачами інформаційної системи. У цьому випадку проблема полягає в тому, щоб надійно засвідчити справжність суб'єктів. Для обміну ключами можна використовувати криптосистеми з відкритим ключем, використовуючи той же алгоритм RSA.

Як узагальнення сказаного про розподіл ключів слід сказати наступне. Завдання управління ключами зводиться до пошуку такого протоколу розподілу ключів, який забезпечував би:

- можливість відмови від центру розподілу ключів;
- взаємне підтвердження справжності учасників сеансу;
- підтвердження достовірності сеансу механізмом запиту-відповіді, використання для цього програмних або апаратних засобів;
- використання при обміні ключами мінімального числа повідомлень.

Питання розподілу ключів й електроного підпису вирішується шляхом побудови **Інфраструктури Відкритих Ключів (Public Key Infrastruture).**

Інфраструктура відкритих ключів (англ. Public key infrastructure, PKI) — інтегрований комплекс методів та засобів (набір служб), призначених забезпечити впровадження та експлуатацію криптографічних систем із відкритими ключами. Інфраструктура відкритих ключів складається з програмного забезпечення, кртиптографічних технологій та служб, які дозволяють підприємствам та організаціям захищати канали зв'язку в комп'ютерних мережах. Вона поєднує електронні цифрові сертифікати, асиметричні алгоритми шифрування, та центри сертифікації у єдину мережеву архітектуру.

Інфраструктура відкритих ключів побудована на криптосистемах з відкритим ключем. Технологія PKI полягає у використанні двох математично пов'язаних цифрових ключів, що мають такі властивості:

- один ключ може бути використаний для шифрування повідомлення, що може бути розшифровано тільки за допомогою іншого ключа;
- навіть якщо відомий один ключ, за допомогою обчислень практично неможливо визначити інший. Один із ключів відкритий для всіх, а інший має

приватний характер і зберігається в захищеному місці. Ці ключі можуть бути використані для автентифікації чи шифрування цифрового підпису електронних даних.

PKI служить не лише для створення цифрових сертифікатів, але і для зберігання великої кількості сертифікатів і ключів, забезпечення резервування і відновлення ключів, взаємної сертифікації, ведення списків анульованих сертифікатів і автоматичного відновлення ключів та сертифікатів після закінчення терміну їхньої дії.

**Основними складовими інфраструктури відкритих ключів** є центри сертифікації, центри реєстрації (засвідчення), репозиторії та архіви сертифікатів. Основними користувачами інфраструктури є: держателі (англ. Certificate Holders) та користувачі сертифікатів (англ. Certificate User) – також відомі як «сторона, що довіряє» (англ. Relying party).

**Центр сертифікації ключів** (англ. Certificate Authority, CA) – юридична особа незалежно від форми власності або фізична особа, яка є суб'єктом підприємницької діяльності, що надає послуги щодо сертифікації відкритих ключів. Центр сертифікації складається з апаратного, програмного забезпечення та обслуги. Центри сертифікації мають два важливих атрибути: назву та відритий ключ. Центри сертифікації виконують чотири основні завдання:

- видача сертифікатів (тобто, центр створює та підписує електронний цифровий сертифікат);
- оброблення статусу сертифікатів та підтримання списків анульованих сертифікатів (англ. CRL);
- поширення поточного списку дійсних та анульованих сертифікатів, аби користувачі мали можливість перевірити стан сертифікату;
- підтримання архівних даних про сертифікати та їхній стан.

Центри сертифікації інколи можуть делегувати відповідальність за деякі з цих задач до інших складових інфраструктури. Центр сертифікації ключів разом із об'єктами, яким видано сертифікати цього центру утворюють домен (ЦСК-домен).

**Центр реєстрації** (англ. Registration Authority, RA) – об'єкт, що відповідає за ідентифікацію та аутентифікацію суб'єктів сертифікатів, але не підписує і не випускає сертифікати (тобто, RA делегуються деякі задачі від імені центру сертифікації).

**Архів** – це база даних, яка зберігає та захищає інформацію для розв'язання різних суперечок, які можуть виникнути у майбутньому. Основним завданням архіву є безпечне зберігання інформації, необхідної для встановлення вірності «старих» електронних підписів.

Центр сертифікації формує **цифрові сертифікати ключа**. Цифровий сертифікат електронного ключа зазвичай містить відкритий ключ, основні дані (реквізити) підписувача –власника особистого ключа, термін дії сертифікату, найменування та реквізити центру сертифікації ключів та електронний цифровий

підпис самого центру сертифікації. Крім того, сертифікат може містити додаткову інформацію про центр сертифікації, власника сертифіката, або інформацію про передбачені способи використання відкритого ключа. Підписувачі звертаються до центрів сертифікації аби отримати свій сертифіакт електронного ключа, аби в подальшому користуватись електронним ключем для підтвердження своєї особи та мати можливість завіряти документи електронним цифровми підписом.

При організації взаємодії абонентів по VPN з'являється поняття протоколу, а точніше комунікаційного та криптографічного протоколу.

**Поняття протоколу.** *Протокол* (protocol) — опис розподіленого алгоритму, в процесі виконання якого два (чи більше) учасники послідовно виконують певні дії і обмінюються повідомленнями. Послідовність кроків протоколу групується в цикли (раунди). В якості учасників (інакше – суб'єктів, сторін) протоколу можуть виступати не тільки користувачі чи абоненти, а й процеси, які виконують яку-небудь функціональну роль, наприклад клієнтські і серверні додатки. Припускається, що всі учасники виконують в ньому якусь активну роль, а пасивні спостерігачі не являються учасниками протоколу.

*Комунікаційний протокол* встановлює послідовність дій учасників при передачі інформації чи інформаційному обміні. Звичайний комунікаційний протокол забезпечує встановлення з'єднання/сеансу, вибір маршруту, виявлення спотворень, відновлення інформації, що передається і т. п.

Безпека протоколу виражається в забезпеченні гарантій виконання таких властивостей, які характеризують безпеку як доступність, конфіденційність, цілісність та ін. Для забезпечення функцій безпеки, які відповідають цим властивостям, в протоколах застосовують спеціальні конструкції. Протокол, який забезпечує підтримку хоча б одної із функцій безпеки, називають *захищеним*, або, точніше, *протоколом забезпечення безпеки* (security protocol).

*Криптографічний протокол* (cryptographic protocol) — протокол, призначений для виконання функцій криптографічної системи; в процесі його виконання учасники використовують криптографічні алгоритми. В даному випадку під *криптографічною системою* розуміють забезпечення безпеки інформації криптографічними методами. На сьогодні, основними функціями криптографічної системи є забезпечення конфіденційності, цілісності, автентифікації, неможливості відмови та стеження. В якості підсистем вона може включати системи шифрування, системи ідентифікації, системи імітозахисту, системи цифрового підпису та деякі інші, а також ключову систему, яка забезпечує роботу решти систем.

Криптографічні протоколи можна класифікувати різними способами, наприклад:

*за числом учасників*

- двосторонній;
- трьохсторонній;

• багатосторонній;

*за числом повідомлень, що передаються:*

• інтерактивний (присутній взаємний обмін повідомленнями);

• не інтерактивний (лише однократна передача); не інтерактивні протоколи часто називають *схемами.*

Найбільш змістовним є підхід, в основі класифікації якого лежить функціональне (цільове) призначення протоколу. Якщо припустити, що протокол виконує одну функцію, то отримуємо наступні типи протоколів:

• протокол забезпечення цілісності повідомлень:

з автентифікацію джерела;

без автентифікації джерела;

• протокол (схема) цифрового підпису:

протокол індивідуального/групового цифрового підпису;

з відновленням/без відновлення повідомлення;

протокол цифрового підпису всліпу;

протокол конфіденційного цифрового підпису;

протокол цифрового підпису з доведенням підробки;

• протокол ідентифікації (автентифікації учасників):

односторонньої автентифікації;

двосторонньої (взаємної) автентифікації;

• конфіденційна передача:

звичайний обмін повідомленнями

широкомовна/циркулярна передача;

чесний обмін секретами;

протокол прив’язки до біту (рядка);

• протокол розподілу ключів:

протокол (схема) попереднього розподілу ключів;

протокол передачі ключа (обміну ключами);

протокол спільної виробки ключа (відкритого розподілу ключів);

протокол парний/груповий;

протокол (схема) розділення секрету;

протокол (розподілу ключів) телеконференції.

*Протокол автентифікації повідомлень* (message authentication protocol) — криптографічний протокол, призначений для забезпечення цілісності повідомлень, які передаються від одного учасника до іншого; під цілісністю розуміється гарантована отримувачу можливість впевнитися, що повідомлення надійшло від заявленого відправника в неспотвореному вигляді.

*Протокол (або схема) ідентифікації* (identification protocol) — протокол автентифікації сторін, які взаємодіють між собою, але не довіряють одна одній.

*Схема цифрового підпису* в найпростішому випадку складається із двох алгоритмів: формування та перевірки підпису. Якщо потрібно приховати зміст документу від учасника, який підписує його (підпис всліпу) або якщо без участі

представника, який підписує, процедура перевірки неможлива (конфіденційний цифровий підпис), вимагаються спеціальні протоколи формування та перевірки підпису.

*Задача розподілу ключів* (key distribution protocol), необхідних для функціонування криптографічної системи, – одна з центральних в криптографії, оскільки в більшості випадків стійкість криптографічної системи базується на недоступності ключів. Тому *протоколи розподілу ключів* **акумулюють** більшість властивостей, які пред'являються до криптографічних протоколів. Вони повинні забезпечувати не лише конфіденційність, а також і автентифікацію всіх аспектів взаємодії: ключа, джерела, відправника та отримувача, часу сеансу, щоб гарантувати, що жоден інший учасник, крім попередньо визначеного другого учасника (і, можливо, інших довірених учасників), не зміг отримати доступ до жодного з секретних ключів.

*Протокол обміну секретами* (secret exchange protocol) — криптографічний протокол з двома учасниками, в якому обмін секретами організований таким чином, щоб у випадку його припинення (за будь-якої причини) знання учасників про секрети одне одного були приблизно однаковими.

*Протокол прив'язки до біту* (bit commitment protocol) — криптографічний протокол з двома учасниками (відправником та отримувачем), через який відправник передає біт інформації (бітове зобов'язання) з дотриманням двох умов: 1) після передачі біта отримувачу (так званого етапу прив'язки) відправник не зможе змінити його значення; 2) отримувач не зможе самостійно визначити значення біта і дізнається про нього лише після виконання відправником так званого етапу розкриття.

*Протокол підкидання (по телефону) монети* (coin flipping (by telephone) protocol) — криптографічний протокол, який дозволяє двом учасникам, які не довіряють один одному, згенерувати спільний випадковий біт. Головна властивість таких протоколів полягає в тому, що якщо хоча б один із учасників чесний, то згенерований біт буде випадковим незалежно від дій іншого учасника. Існують узагальнення на випадок кінцевих бітових строк та довільного числа учасників.

Цікавим видом протоколів розподілу ключів є так звані *групові протоколи* (group-oriented protocol), в яких ключі розподіляються між групами учасників. Якщо всі групи, які мають на це право, формують однакові ключі, то такий протокол називають *протоколом розподілу секрету* (secret sharing protocol). Якщо ж в різних груп повинні бути різні ключі, то це — протокол телеконференції.

В *протоколах групового підпису* (group signature protocol) припускається одночасна участь попередньо визначеної групи учасників, причому у випадку відсутності хоча б одного учасника групи формування підпису неможливе.

Наведену вище класифікацію можна уточнити, якщо при цьому окремо розглядати примітивні та прикладні криптографічні протоколи.

*Примітивний криптографічний протокол* (primitive cryptographic protocol) — це криптографічний протокол без самостійного прикладного значення, який використовується як базовий компонент при побудові прикладних криптографічних протоколів. Як правило, він вирішує якусь абстрактну задачу. Приклади — протокол обміну секретами, протокол прив'язки до біту, протокол підкидання монети (по телефону).

*Прикладний криптографічний протокол* (application cryptographic protocol) призначений для розв'язку практичних задач забезпечення функцій — сервісів безпеки за допомогою криптографічних систем. Варто зазначити, що прикладні протоколи, як правило, забезпечують не одну, а відразу декілька функцій безпеки. Більше того, такі протоколи, як IPsec, насправді, є великими сімействами різних протоколів, які включають багато різних варіантів для різних ситуацій та умов застосування.

Класифікацію криптографічних протоколів можна проводити за іншими ознаками:

- за типом криптографічних систем, що використовуються:

протокол на основі симетричних криптографічних систем;

протокол на основі асиметричних криптографічних систем;

змішаний протокол;

- за способом функціонування:

інтерактивний/неінтерактивний протокол;

одно-, дво-, трьох-, четирьохкроковий протокол;

протокол з арбітром (протокол з посередником);

двосторонній/з довіреною третьою стороною (з центром довіри) протокол;

працюючий в реальному часі (on-line)/у відкладеному режимі (off-line) і т. п.

*Протокол з арбітром або протокол з посередником* (arbitrated protocol) — криптографічний протокол, в якому для розв'язання суперечок між учасниками вимагається арбітраж.

Якщо обмін повідомленнями здійснюється за участю спеціально виділеного учасника, який володіє довірою інших сторін, то говорять *про протоколи з довіреною третьою стороною або про протоколи з центром довіри.* Якщо учасник повинен звертатися до центру постійно в кожному сеансі протоколу, то говорять про роботу у відкладеному режимі (off-line). Інколи виділяють ще режим in-line, коли весь обмін повідомленнями проходить через центр довіри.

Особливий клас протоколів складають *протоколи доведення:*

- інтерактивне/не інтерактивне доведення;
- протокол доведення знання (якого-небудь факту);
- протокол доведення вміння розв'язувати яку-небудь задачу.

*Доведення інтерактивне* (interactive proof) здійснюється шляхом виконання протоколу двома учасниками — тим, хто доводить, та тим, хто

перевіряє; в процесі роботи учасники обмінюються повідомленнями, які, зазвичай, залежать від випадкових чисел. Ці числа можуть утримуватися в таємниці. Мета того, хто доводить — запевнити перевіряючого в істинності деякого твердження. Перевіряючий або приймає, або заперечує доказ. На відміну від звичайного математичного поняття «доведення» в даному випадку доведення несе не абсолютний, а ймовірнісний характер і характеризується двома ймовірностями. Якщо твердження, що доводиться, – вірне, то імовірність вірності доведення повинна наближатися до одиниці при збільшенні числа повторів протоколу. Якщо це твердження хибне, то при збільшенні числа повторів протоколу ймовірність правильності доведення повинна наближатися до нуля.

*Доведення знання* (proof of knowledge) — доведення інтерактивне, при якому той, хто доводить, переконує перевіряючого в тому, що він володіє секретною інформацією, не розкриваючи її. До категорії доведення знання відносяться протоколи ідентифікації. Центральною властивістю таких протоколів є *розголошення нульове* (zero-knowledge property) ¯ властивість протоколу доведення знання, яка забезпечує таке його виконання, при якому ніяку інформацію про твердження, що доводиться, крім факту його істинності, не можливо отримати шляхом обману перевіряючим із переданих повідомлень за час, який поліноміально залежить від сумарної довжини цих повідомлень.

## 5.2 Класифікація та основи побудови віртуальних приватних мереж

Об’єднання локальних мереж і окремих комп’ютерів через відкрите зовнішнє середовище передавання інформації в єдину віртуальну мережу, яка забезпечує захист інформації, що в ній циркулює, називається **захищеною (або приватною) віртуальною мережею** (англ. – Virtual Private Network, VPN)

Термін «віртуальна» означає, що така мережа формується як деяка підмножина реальної мережі, з каналами зв’язку, що моделюються реальними каналами

Особливою ознакою віртуальної приватної (захищеної) мережі є її відокремлення від реальної мережі, яке повинне бути достатньо надійним для гарантування конфіденційності та цілісності інформації, що в ній передається, а також для забезпечення автентифікації сторін і унеможливлення відмовлення від авторства (англ. – Non-Repudiation)

Незважаючи на те, що комунікації здійснюються по мережах з меншим або невідомим рівнем довіри (наприклад, по публічним мережам), **рівень довіри** до побудованої логічної мережі не залежить від рівня довіри до базових мереж завдяки використанню засобів криптографії (шифрування, аутентифікації, інфраструктури відкритих ключів, засобів для захисту від повторів і змін, що передаються по логічній мережі, повідомлень).

**Сервіси VPN:**

- забезпечення конфіденційності;

- забезпечення цілісності;
- автентифікація та запобігання відмовленню від авторства.

**Забезпечення конфіденційності.**

Найпростішим і найпоширенішим способом забезпечення конфіденційності інформації є її шифрування, або криптографічне закриття. Незважаючи на те, що самі алгоритми шифрування дуже складні, їх реалізація великих проблем не викликає. Доволі значну проблему становить керування ключами, особливо в разі значного збільшення кількості користувачів. В реалізації VPN керування ключами є одною з головних проблем, що потребує надійного і ефективного рішення

Шифрування має неминучий побічний ефект – деяку втрату продуктивності. Апаратно реалізоване шифрування звільняє пристрої захисту від додаткового навантаження, пов'язаного з виконанням алгоритмів шифрування, і забезпечує кодування трафіка без втрати швидкості обміну. У разі здійснення атаки на засоби захисту VPN існує загроза підміни програмних компонентів, в тому числі саме тих, що забезпечують шифрування. Загроза несанкціонованого впливу на апаратні засоби є значно менш ймовірною. Для апаратної реалізації шифрування застосовуються спеціалізовані інтегральні схеми прикладної орієнтації (англ. – Application-Specific Integrated Circuit, ASIC).

**Забезпечення цілісності.**

Цілісність контролюється використанням математичних алгоритмів хешування. Важливо підкреслити, що криптографічні механізми не забезпечують захист цілісності, а лише дозволяють впевнитись, що цілісність не була порушена, або, навпаки, виявити порушення

Алгоритми хешування також потребують значних ресурсів процесора. Це дає підстави реалізовувати виконання цих алгоритмів в апаратних засобах з використанням інтегральних схем прикладної орієнтації

**Автентифікація та запобігання відмовленню від авторства**.

Запобігання відмовленню від авторства (англ. – Non-Repudiation) – це додаткова функція, що реалізується на базі автентифікації. У захищеному спілкуванні часто виникають випадки, коли крім підтвердження того, що абонент є саме тим, за кого він себе намагається видати, важливо отримати незаперечні докази того, що повідомлення одержано від конкретного користувача. Також буває необхідним доказове підтвердження того, що певний користувач дійсно одержав деяке повідомлення. Ці функції захисту у ряді випадків повинні бути невід'ємною складовою реалізації VPN

**Способи утворення захищених віртуальних каналів.**

Будь-який з двох вузлів віртуальної мережі, між якими формується захищений тунель, може належати кінцевій чи проміжній точці потоку повідомлень, який захищають. Відповідно можливі різні способи утворення захищеного віртуального каналу.

- кінцеві точки тунелю співпадають з кінцевими точками потоку повідомлень;
- кінцевою точкою захищеного тунелю обирають брандмауер або граничний маршрутизатор локальної мережі, захищений тунель утворюється лише у публічній мережі;
- в якості кінцевих точок захищеного тунелю виступають засоби, що встановлені не на комп'ютерах користувачів, а на площах провайдерів Інтернет.

**Класифікувати VPN** рішення можна за кількома основними параметрами:

1) За типом використовуваного середовища:

- захищені VPN мережі. Найбільш поширений варіант приватних мереж. З його допомогою можливо створити надійну і захищену підмережу на основі ненадійної мережі, як правило, Інтернету. Приклади захищених VPN: IPSec , OpenVPN і PPTP;
- довірчі VPN мережі. Використовуються у випадках, коли середовище, через яке передається інформація можна вважати надійним і необхідно вирішити лише завдання створення віртуальної підмережі в рамках більшої мережі. Питання забезпечення безпеки стають неактуальними. Прикладами подібних VPN вирішенні є: MPLS і L2TP. Ці протоколи перекладають завдання забезпечення безпеки на інші, наприклад L2TP, як правило, використовується в парі з IPSec.

2) За способом реалізації:

- VPN мережі у вигляді спеціального програмно–апаратного забезпечення. Реалізація VPN мережі здійснюється за допомогою спеціального комплексу програмно–апаратних засобів. Така реалізація забезпечує високу продуктивність і, як правило, високу ступінь захищеності;
- VPN мережі у вигляді програмного рішення. Використовують персональний комп'ютер зі спеціальним програмним забезпеченням, що забезпечує функціональність VPN;
- VPN мережі з інтегрованим рішенням. Функціональність VPN забезпечує комплекс, вирішальний також завдання фільтрації мережевого трафіку, мережевого екрану і забезпечення якості обслуговування.

3) За призначенням:

- Intranet VPN. Використовують для об'єднання в єдину захищену мережу декількох розподілених приватних мереж, які обмінюються даними за відкритими каналах зв'язку. Інтранет технології використовуються для організації захищеного з'єднання між підрозділами одного або різних організайії, об'єднаних корпоративними мережами зв'язку;
- Remote Access VPN. Використовують для створення захищеного каналу між сегментом корпоративної мережі і одиночним користувачем, який, працюючи вдома, підключається до корпоративних ресурсів з домашнього комп'ютера або,

перебуваючи у відрядженні, підключається до корпоративних ресурсів за допомогою ноутбука;

- Extranet VPN. Використовують для мереж, до яких підключаються «зовнішні» користувачі (клієнти). Рівень довіри до них набагато нижче, ніж до співробітників, тому потрібне забезпечення спеціальних «рубежів» захисту, запобігають або обмежують доступ останніх до особливо цінною, конфіденційної інформації.

Схему класифікації VPN наведено на рисунку 5.2.

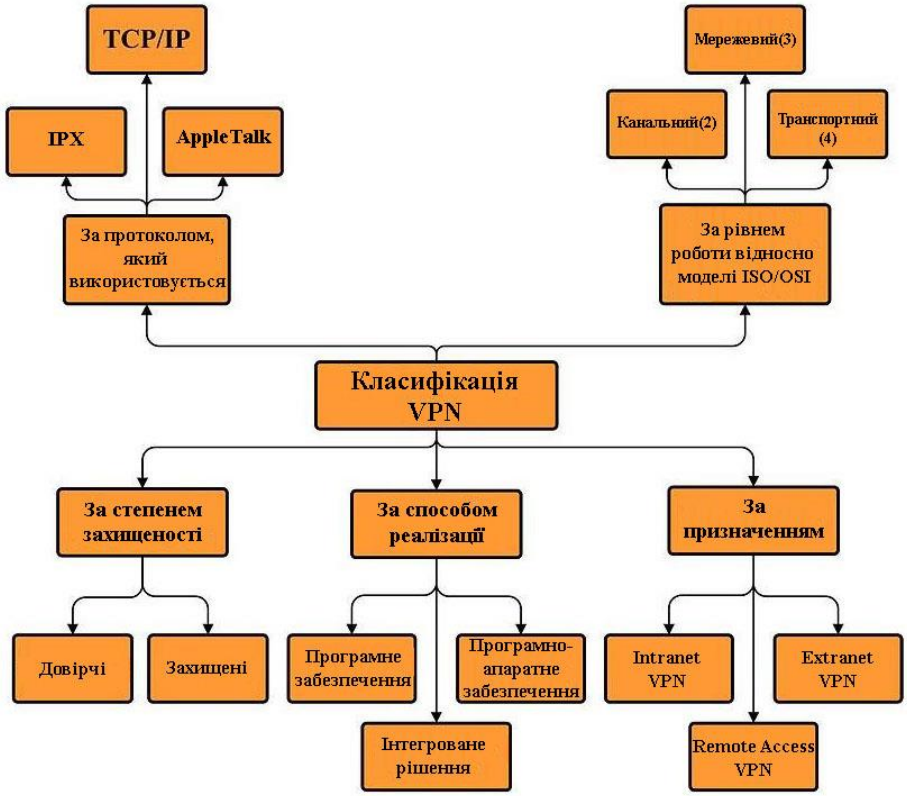



Рисунок 5.2 – Класифікація VPN

**Основні компоненти VPN:**

- VPN-шлюз - мережеве пристрій, підключений до декількох мереж, виконує функції шифрування, ідентифікації, аутентифікації, авторизації та тунелювання. Може бути реалізований як програмно, так і апаратно.
- VPN-клієнт (хост) реалізується програмно. Виконує функції шифрування і аутентифікації. Мережа може бути побудована без використання VPN-клієнтів.
- Тунель - логічний зв'язок між клієнтом і сервером. В процесі реалізації тунелю використовуються методи захисту інформації.
- Граничний сервер - це сервер, який є зовнішнім для корпоративної

мережі. В якості такого сервера може виступати, наприклад, брендмауери або система NAT.

- Забезпечення безпеки інформації VPN - ряд заходів щодо захисту трафіку корпоративної мережі при проходженні по тунелю від зовнішніх і внутрішніх загроз.

**Схеми взаємодії провайдера й клієнта:**

- Користувацька схема - обладнання розміщується на території клієнта, методи захисту інформації та забезпечення QoS організовуються самостійно.
- Провайдерська схема - засоби VPN розміщуються в мережі провайдера, методи захисту інформації та забезпечення QoS організовуються провайдером.
- Змішана схема - використовується при взаємодії клієнта з декількома провайдерами.

**Підтримка VPN на різних рівнях моделі OSI:**

- Канальний рівень: L2TP, PPTP і ін. (Авторизація і аутентифікація), Технологія MPLS (встановлення тунелю).
- Мережевий рівень: IPSec (архітектура «хост-шлюз» і «шлюз-шлюз», підтримка шифрування, авторизації та аутентифікації, проблеми з реалізацією NAT).
- Транспортний рівень: SSL / TLS (архітектура «хост-хост» з'єднання з кінця в кінець, підтримка шифрування і аутентифікації, реалізований тільки для підтримки TCP-трафіку).

5.3 Протокол реалізації VPN PPTP

Протокол **PPTP** (Point-to-Point Tunneling Protocol), призначений для створення захищених віртуальних каналів при доступі віддалених користувачів до локальних мереж через Інтернет. Він передбачає створення криптозахищеного тунелю на канальному рівні моделі OSI як для випадку прямого з'єднання віддаленого комп'ютера з відкритою мережею, так і для випадку приєднання його до відкритої мережі по телефонній лінії через провайдера. При встановленні з'єднання використовується застарілий протокол двухточкового зв'язку Poinl–to–Point Protocol (далі – PPP) і є його розширенням.

Основний опис протоколу поданий в RFC2637 відкритим міжнародним співтовариством проектувальників, учених, мережних операторів і провайдерів (Internet Engineering Task Force, далі – IETF). За замовчуванням протокол використовує передачу даних за допомогою TCP пакетів і резервує 1723 порт.

**Протокол PPP** (Point-to-Point Protocol - протокол двухточечного зв'язку) є стандартним протоколом Інтернету. Протокол PPP являє собою ціле сімейство протоколів, в яке, зокрема, входять:

- протокол управління лінією зв'язку (Link Control Protocol, LCP) дозволяє встановлювати і тестувати канали зв'язку, домовлятися про параметри їх використання, відключати канали зв'язку, коли вони не потрібні. Цей протокол

підтримує синхронні і асинхронні лінії, біт і байт-орієнтоване кодування;

- протокол управління мережею (Network Control Protocol, NCP);
- багатоканальний протокол PPP (Multi Link PPP, MLPPP);
- протокол аутентифікації за паролем (Password Authentication Protocol, PAP);
- протокол аутентифікації по квитуванню виклику (Challenge Handshake Authentication Protocol, CHAP).

**Інкапсуляція кадрів PPP в IP.**

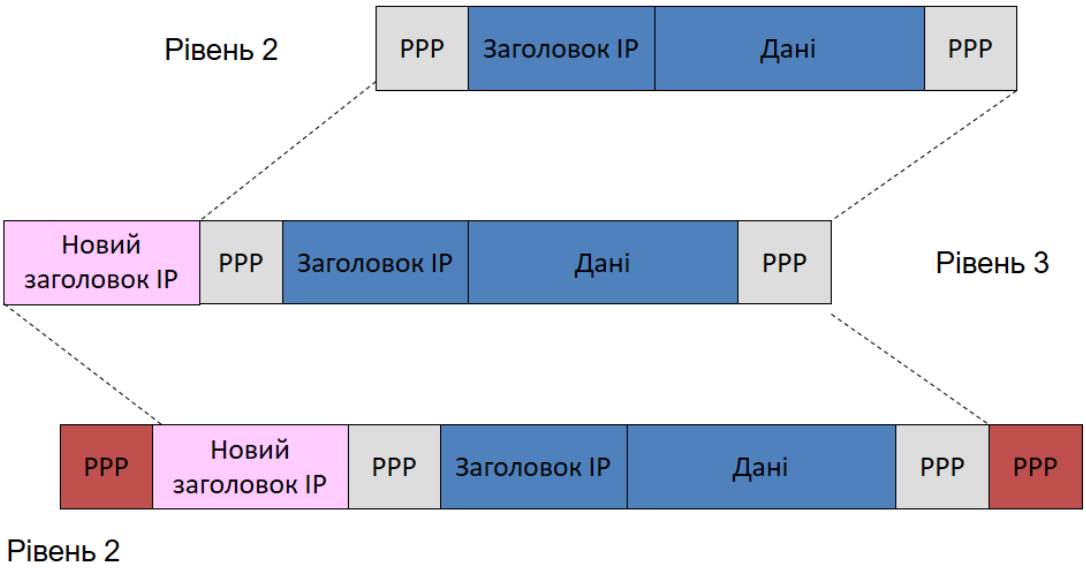



Рисунок 5.3 – Інкапсуляція кадрів PPP в IP

Принцип роботи PPP.

Для того, щоб організувати зв'язок через канал з безпосереднім з'єднанням, PPP на початку відправляє пакети LCP для завдання конфігурації з'єднання, а також перевірки каналу передачі даних. Після того, як канал встановлений і пакетом LCP виконано необхідне узгодження факультативних засобів, PPP відправляє пакети NCP, щоб вибрати і визначити конфігурацію одного або більше протоколів мережного рівня. Як тільки конфігурація кожного вибраного протоколу визначена, дейтаграми з кожного протоколу мережного рівня можуть бути відправлені через цей канал. Канал зберігає свою конфігурацію до тих пір, поки пакети LCP або NCP явно не закриють його або доки не станеться яка–небудь зовнішня подія (наприклад, збіжить термін бездіяльності таймера або втрутиться який–небудь користувач).

Протокол контролю каналу LCP

Протокол PPP для достатньої універсальності і застосовності до широкої різноманітності систем включає протокол контролю каналу LCP. LCP використовується, щоб автоматично погоджувати опції формату інкапсуляції, змінювати межі розмірів пакетів, виявляти зациклення ланки і інші помилкові ситуації, пов'язані з відмінностями конфігурацій, і розривати зв'язок.

Існує три класи пакетів LCP:

- пакети для організації каналу зв'язку. Використовуються для організації і вибору конфігурації каналу;
- пакети для завершення дії каналу. Використовуються для завершення дії каналу зв'язку;
- пакети для підтримки працездатності каналу. Використовуються для підтримки і відладки каналу.

Протоколи контролю мережі NCP

Канали PPP мають багато проблем з використовуваним сімейством мережевих протоколів. Наприклад, призначення і управління адрес IP, які є проблемою навіть в локально обчислювальних системах, є особливо важкими для комутованих каналів точка–точка. Ці проблеми вирішуються використанням стека протоколів контролю мережі NCP, кожен з яких відповідає за певні функції, потрібні відповідними протоколами мережного рівня.

Протокол автентифікації

Для деяких каналів може знадобитися, щоб одноранговий об'єкт підтвердив свою достовірність перед дозволом обмінюватися пакетами протоколу мережного рівня. Ця опція конфігурації забезпечує спосіб узгодження типу протоколу встановлення достовірності. За замовчуванням, встановлення достовірності не потрібно. Користувач може включити декілька опцій конфігурації протоколу автентифікації в пакетах «Запит конфігурації».

У процесі конфігурування, підтримання та завершення з'єднання «точка-точка», ланка передачі даних PPP проходить кілька різних стадій, які визначені в наступній спрощеної діаграмі станів:

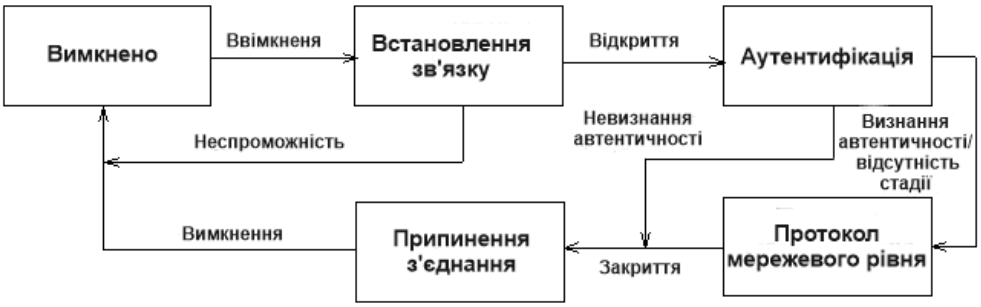



Рисунок 5.4 – Діаграма станів протоколу PPP

**Протокол PPTP**.

- Протокол PPTP (англ. – Point-to-Point Tunneling Protocol) був розроблений компанією Microsoft за підтримки компаній Ascend Communications, 3Com/Primary Access, ECI-Telematics та US Robotics.
- Фактично, цей протокол є розширенням протоколу PPP (англ. – Point-to-Point Protocol), яке дозволяє створювати криптозахищені тунелі на канальному рівні моделі OSI.

- Протокол PPTP отримав статус проекту стандарту Internet, однак в якості стандарту так і не був затверджений. .
- Головне призначення протоколу – організація доступу віддалених користувачів до локальних мереж як у випадку прямого з'єднання віддаленого комп'ютера з публічною мережею, так і у випадку підключення до публічної мережі по телефонній лінії через провайдера. При цьому для віддаленого користувача, який підключений до локальної мережі через сервер віддаленого доступу (RAS), імітується знаходження його комп'ютера безпосередньо в локальній мережі. Це досягається завдяки тунелюванню пакетів повідомлень.

PPTP тунель складається з каналу управління і з каналу передачі даних. Спочатку встановлюється канал управління, а потім канал даних. Організація каналу даних здійснюється за допомогою протоколу інкапсулювання Generic Routing Encapsulation (далі – GRE). GRE інкапсулює мережевий рівень, наприклад IPX, AppleTalk, щоб забезпечити можливість їх передачі в IP–мережах. Однак GRE не має можливості встановлювати сесії і забезпечувати захист даних від зловмисників. Для цього використовується здатність PPTP створювати з'єднання для управління тунелем. Застосування GRE в якості методу інкапсуляції обмежує поле дій PPTP тільки мережами IP. На рисунку 5.5 зображено архітектуру пртоколу PPTP.


Рисунок 5.5 – Архітектура протоколу PPTP

Шифрування за допомогою PPTP гарантує, що ніхто не зможе отримати доступ до даних при пересиланні через Інтернет. В даний час підтримуються два методи шифрування:

- Протокол шифрування Microsoft Point–to–Point Encryption (далі – MPPE) сумісний тільки з Microsoft Challenge Handshake Authentication Protocol (далі – MS–CHAP);
- EAP–TLS вміє автоматично вибирати довжину ключа шифрування при узгодженні параметрів між клієнтом і сервером.

MPPE підтримує роботу з ключами довжиною 40, 56 або 128 біт. PPTP змінює значення ключа шифрування після кожного прийнятого пакета. Протокол

MMPE розроблявся для каналів зв'язку точка–точка, в яких пакети передаються послідовно, і втрата даних дуже мала. У цій ситуації значення ключа для чергового пакета залежить від результатів дешифрування попереднього пакету. При побудові віртуальних мереж через мережі загального доступу ці умови дотримуватися неможливо, так як пакети даних часто приходять до одержувача не в тій послідовності, в якій були відправлені. Тому PPTP використовує для зміни ключа шифрування порядкові номери пакетів. Це дозволяє виконувати дешифрацию незалежно від попередніх прийнятих пакетів.

**Автентифікація в PPTP**

Для автентифікації в PPTP можуть застосовуватись різні протоколи. В реалізації PPTP від Microsoft підтримуються протоколи:

- PAP (англ. – Password Authentication Protocol – протокол автентифікації за паролем). Передбачає передачу ідентифікаторів і паролів у відкритому вигляді
- CHAP (англ. – Challenge-Handshaking Authentication Protocol – протокол автентифікації за процедурою рукостискання). Передбачає одержання від сервера випадкового числа і шифрування на ньому пароля. Таким чином, не лише пароль не передається по мережі у відкритому вигляді, але й зашифровані образи пароля кожного разу різні

**Етапи встановлення з'єднання за технологією PPTP**

*Першим етапом* є ініціація з'єднання, що в свою чергу ділиться на три підетапи:

1. Обмін пакетами TCP, які містять в собі контрольні біти (SYN, ACK) між сервером та клієнтом.

2. Встановлення з'єднання. Сервер та клієнт обмінюються між собою повідомленням управління (Start Control Connection Request) для встановлення з'єднання управління між концентратором доступу (PPTP Access Concentrator, далі – PAC) і мережевим сервером (PPTP Network Server, далі – PNS) та контрольним повідомленням (Start Control Connection Reply), яке надсилається у відповідь на отримане повідомлення запиту і містить код результату, що вказує на результат спроби встановлення з'єднання управління.

3. Інціація виклику. Сервер надсилає клієнту запит вихідного виклику (Outgoing–Call–Request Message). Це повідомлення про керування PPTP, яке передається від PNS, щоб вказати, що повинен бути встановлений вихідний виклик від PAC. Клієнт надсилає відповідь на вихідний виклик (Outgoing–Call–Reply Message). У відповіді вказується результат спроби вихідного дзвінка та інформація про конкретні параметри, що використовуються для виклику. Він надає інформацію, яка дозволяє PNS регулювати передачу даних PAC для цієї сесії.

*Другий етап* – встановлення з'язку. Між PNS та PAC ініціюється PPP сесія, а першим кроком до її встановлення є налаштування і тестування каналу передачі даних за допомогою протоколу керування з'язком (далі – LCP).

Встановлення LCP можна розділити на чотири фази:

– організація каналу і узгодження його конфігурації. Ця фаза завершується після того, як буде відправлений і прийнятий пакет підтвердження конфігурації.

– визначення якості каналу зв'язку. У цій фазі перевіряється, чи є якість каналу достатньою для виклику протоколів мережного рівня.

– узгодження конфігурації протоколів мережевого рівня. Після того, як LCP завершить фазу визначення якості каналу зв'язку, відповідними NCP може бути вибрана конфігурація мережних протоколів, і вони можуть бути у будь–який момент викликані і звільнені для подальшого використання. Якщо LCP закриває цей канал, він інформує про це протоколи мережного рівня, щоб вони могли вжити відповідні заходи.

– припинення дії каналу. Відбувається за запитом користувача та у разі фізичної події (втрата носія або закінчення періоду бездіяльності таймера).

*Третій етап* встановлення з'єднання – автентифікація**.** Після встановлення PPP–сесії передбачено додаткову фазу автентифікації. Протокол перевірки автентичності (Challenge–Handshake) використовується для періодичної перевірки ідентичності вузла за допомогою тристороннього рукостискання. Це робиться після встановлення початкового посилання, і може бути повторене в будь–який час після встановлення посилання.

Після завершення етапу встановлення посилання, аутентифікатор відправляє повідомлення "виклик" до вузла. Вузол відповідає значенням, розрахованим за допомогою односторонньої хеш–функції. Аутентифікатор перевіряє відповідь проти власного розрахунку очікуваного значення хешу. Якщо значення збігаються, автентифікація підтверджується; інакше з'єднання повинно бути припинено. У випадкових інтервалах, аутентифікатор відправляє новий виклик рівному і повторює ці кроки.

*Четвертий етап* – узгодження стиснення даних. Після встановлення зв'язку, за бажанням можна обговорити додаткові об'єкти. Одним з таких об'єктів є компресія даних, яка відповідає за налаштування, увімкнення та вимикання алгоритмів стиснення даних на обох кінцях посилання "точка–точка та сигналізації про несправність механізму стиснення/декомпресії.

*П'ятий етап* – фаза протоколу мережевого рівня. На цьому рівні відбувається обмін протоколом керування IP (IPCP), який відповідає за налаштування, увімкнення та вимикання модулів протоколу IP на обох кінцях зв'язку точка–точка.

*Шостий етап* – інкапсуляція. PPTP використовує вдосконалений механізм GRE, щоб забезпечити керований потоком і керований перевантаженням інкапсульований дейтаграмний сервіс для перенесення пакетів PPP. Ці удосконалення дозволяють забезпечити низький рівень перевантаження та управління потоками на тунелях, які використовуються для передачі даних

користувача між PAC і PNS. PPTP не диктує конкретні алгоритми, які будуть використовуватися для цього контролю низького рівня, але він визначає параметри, які повинні бути передані для того, щоб дозволити таким алгоритмам працювати. Приклад інкапсуляції пакетів за допомогою GRE зображено на рисунку 5.6. Де Eth – оригінальні дані канального рівня, IP – оригінальні дані мережевого рівня, Payload – дані користувача (корисне навантаження), Outer Eth – сформований фрейм для інкапсуляції в тунель, Outer IP – сформований пакет для ідентифікації на мережевому рівні. Оригінальні дані приховані за сформованими за допомогою GRE з метою правильної адресації у разі декапсуляції.

PPTP вимагає створення тунелю для кожної сполученої пари PNS–PAC. Цей тунель використовується для перенесення всіх пакетів PPP для сеансів користувача для сеансів, що включають задану пару PNS–PAC. Ключ, який присутній у заголовку GRE, вказує, до якого сеансу належить певний пакет PPP.

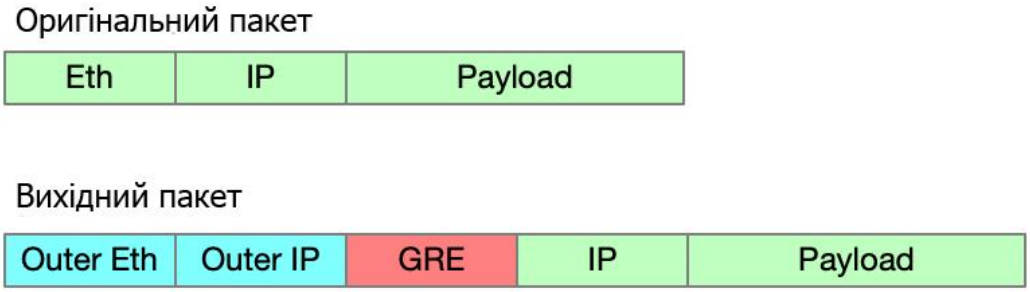



Рисунок 5.6 – Інкапсуляція пакетів з використанням GRE

5.4 Протокол реалізації VPN L2TP

Layer 2 Tunneling Protocol (L2TP) – це протокол тунелювання канального рівня [6]. Він є розширенням протоколу тунелювання PPTP, який використовується постачальниками інтернет–послуг для забезпечення роботи віртуальної приватної мережі VPN через Інтернет.

Протокол L2TP розроблений організацією IETF. Протокол розроблявся як протокол захищеного тунелювання PPP–трафіку через мережі загального призначення з довільною середовищем. Робота над цим протоколом велася на основі протоколів PPTP і L2F.

Згідно специфікації протоколу L2TP роль серверу віддаленого доступу провайдера повинен виконувати концентратор L2TP Access Concentrator (далі – LAC), який забезпечує віддаленому користувачеві мережевий доступ до його локальної мережі через Інтернет. Як сервер віддаленого доступу локальної мережі повинен виступати мережевий сервер L2TP Network Serve (далі – LNS), що функціонує на сумісних з протоколом PPP платформах.

На відміну від PPTP, протокол L2TP не прив'язаний до протоколу IP, тому він може бути використаний в мережах з комутацією пакетів, наприклад в

мережах ATM (Asynchronous Transfer Mode) або в мережах з ретрансляцією кадрів. Крім того, в протокол L2TP додана важлива функція управління потоками даних, а також ряд відсутніх у специфікації протоколу PPTP функцій захисту. Зокрема, включена можливість роботи з протоколами Authentication Header (далі – AH) і Encapsulating Security Payload (далі – ESP) стека протоколів IPSec.

L2TP використовує два види пакетів: керуючі та інформаційні повідомлення. Керуючі повідомлення використовуються при встановленні, підтримці та анулювання тунелів і викликів. Інформаційні повідомлення використовуються для інкапсуляції PPP–кадрів пересилаються по тунелю. Керуючі повідомлення використовують надійний контрольний канал в межах L2TP, щоб гарантувати доставку. Інформаційні повідомлення якщо відбувається їх втрата, не відсилаються повторно. Архітектуру протоколу представлено на рисунку 5.7, де зображено взаємозв'язок кадрів PPP і керуючих повідомлень над керуванням L2TP і каналами даних. PPP–кадри передаються через ненадійний канал даних, інкапсульовані спочатку в L2TP, а потім в транспортні пакети, такі як UDP, Frame Relay, ATM і т.д. Керуючі повідомлення надсилаються через надійний керуючий канал L2TP, який передає пакети в межах того ж пакетного транспорту.

Необхідна процедура встановлення PPP–сесії тунелювання L2TP включає в себе два етапи:

– встановлення керуючого каналу для тунелю;
– формування сесії відповідно до запиту вхідного або вихідного дзвінка.

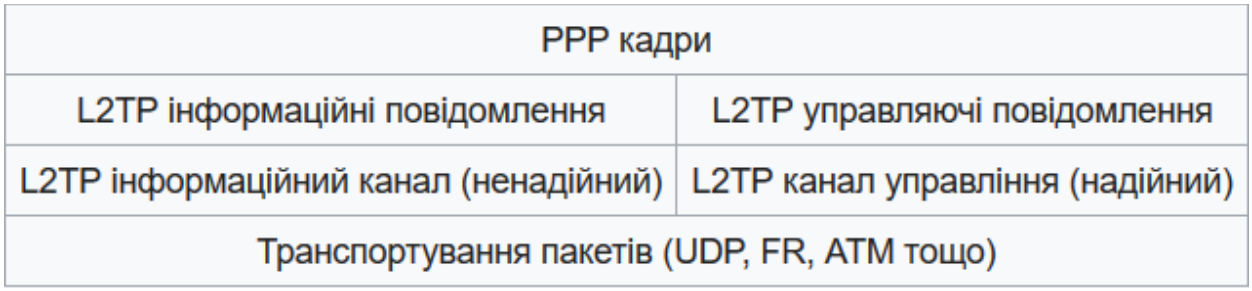



Рисунок 5.7 – Архітектура протоколу L2TP

Тунель і відповідний керуючий канал повинні бути сформовані до ініціалізації вхідного або вихідного дзвінка. Сесія повинна бути реалізована до того, як L2TP зможе передавати PPP–кадри через тунель. В одному тунелі можуть існувати кілька сесій між одними і тими ж LAC і LNS. На відміну від своїх попередників, протоколів PPTP і L2F, протокол L2TP надає можливість відкривати між кінцевими абонентами відразу декілька тунелів, кожен з яких може бути виділений для окремого додатка. Ці особливості забезпечують гнучкість і безпеку тунелювання.

Реалізація L2TP не використовує шифрування MPPE для PPP–датаграм. Служби шифрування протоколу L2TP працюють на основі IPSec в

транспортному режимі. Комбінація протоколів L2TP і IPSec називається L2TP/IPSec. Протокол L2TP/IPSec забезпечує більш високий ступінь захисту даних, ніж PPTP, оскільки використовує алгоритм шифрування 3DES або AES. Якщо такий високий рівень захисту не потрібен, можна використовувати алгоритм DES з одним 56–розрядним ключем. Шифрування даних DES/3DES використовує ключі шифрування, отримані в процесі узгодження за протоколом Internet Key Exchange (далі –IKE). Крім того, за допомогою алгоритму HMAC (Hash Message Authentication Code) забезпечується аутентифікація даних, для чого цей алгоритм створює хеш довжиною 128 розрядів.

Представлена на рисунку 5.8 інкапсуляція пакетів L2TP/IPSec виконується на двох рівнях. Перший рівень – інкапсуляція L2TP. До PPP кадру (IP датаграми) додаються заголовки L2TP і UDP. Другий рівень – інкапсуляція IPSec. До даного повідомленя L2TP додаються заголовок і замикач протоколу ESP IPSec. В кінці все це інкапсулюється в IP–пакет для відправлення або декапсуляції у зворотному випадку.

Замикач перевірки достовірності IPSec забезпечує цілісність і перевірку достовірності повідомлення. У IP заголовку міститься початковий і кінцевий IP адреси, відповідні VPN клієнтові і VPN серверу.

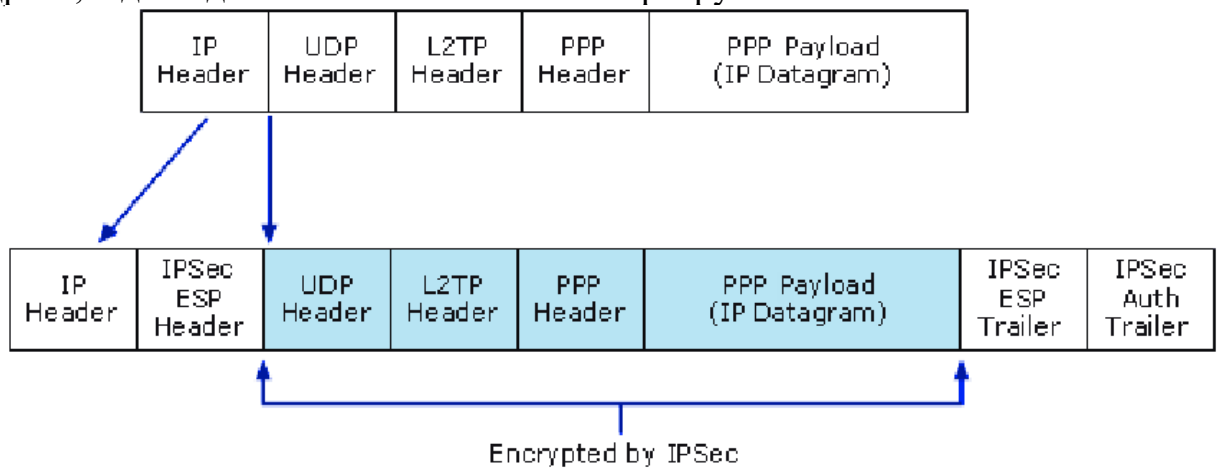



Рисунок 5.8 – Інкапсуляція пакетів L2TP/IPSec

Шифрування L2TP відбувається з використанням протоколу ESP IPSec.

5.5 Розширення безпеки мережевого рівня – протокол IPSEC

**IPSec** (безпека протоколу Інтернету) – це стек протоколів, які забезпечують безпеку комунікацій Інтернету на рівні IP. Найбільш поширеним поточним використанням IPSec є забезпечення VPN, він також може забезпечувати безпеку від хоста до хоста.

Стек протоколів IPSec використовується для автентифікації учасників обміну, тунелювання трафіку і шифрування IP– пакетів. Основне призначення протоколу IPSec – забезпечення безпечної передачі даних в мережах IP. Оскільки архітектура IPSec забезпечує сумісність з протоколом IPv4, то його підтримку

досить реалізувати на обох кінцях з'єднання й проміжні мережеві вузли можуть не мати інформації про IPSec. Протокол IPSec може захищати трафік як поточної версії протоколу IPv4, так і версії IPv6, яка поступово впроваджується в Інтернет.

Для того, щоб забезпечити автентифікацію, конфіденційність і цілісність даних, що передаються, стек протоколів IPSec побудований на базі стандартизованих криптографічних технології:

− обміни ключами здійснюються згідно з алгоритмом Діфі–Хеллмана для розподілу секретних ключів між користувачами у відкритій мережі;

− використовуються асиметричні криптографічні алгоритми для підпису обміну ключами за схемою Діфі–Хеллмана, щоб гарантувати достовірність двох сторін і зіпобігти атакам типу «людина–по–середині»;

− використовуються цифрові сертифікати для підтвердження достовірності відкритих ключів;

− використовуються блокові симетричні алгоритми шифрування даних;

− використовуються алгоритми автентифікації повідомлень на базі функції хешування.

Архітектуру стека протоколів IPSec представлено на рисунку 5.9. В неї входять три протоколи: протокол автентифікаційного заголовка AH, протокол інкапсулюючого захисту ESP і протокол узгодження параметрів віртуального каналу і управління ключами IKE.

Протокол узгодження параметрів віртуального каналу і управління ключами IKE визначає спосіб ініціалізації захищеного каналу, включаючи узгодження алгоритмів крипто захисту, що використовуються, а також процедури обміну і управління секретними ключами у рамках захищеного з'єднання.

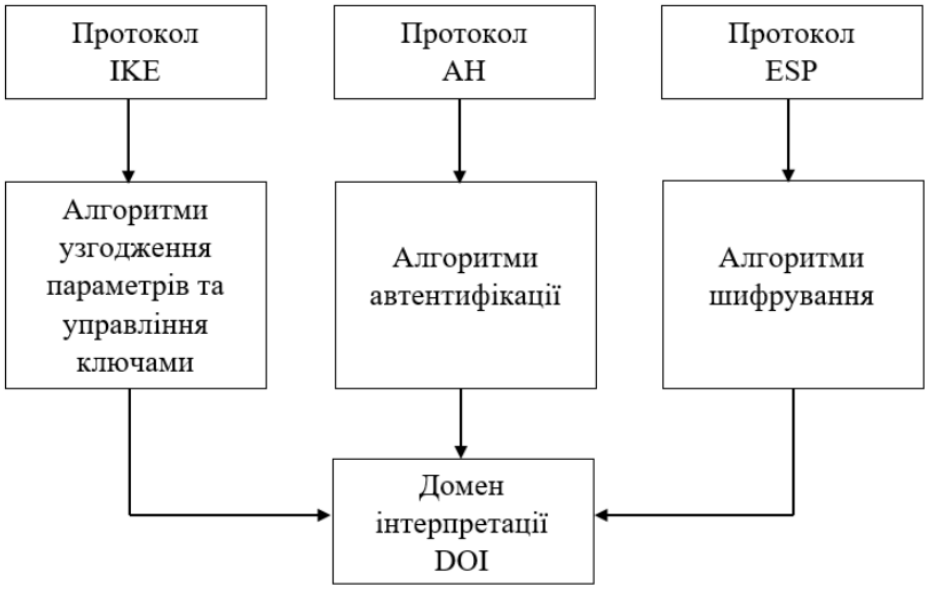



Рисунок 5.9 – Архітектура стека протоколів IPSec

**Рівні архітектури IPSec.**

- **Верхній рівень** – протоколи захисту віртуального каналу і узгодження параметрів захисту. Протоколи AH та ESP не залежать від конкретних алгоритмів шифрування й автентифікації. Можуть застосовуватись різні методи автентифікації, типи ключів, алгоритми шифрування та розподілу ключів. Протоколи AH та ESP зареєстровані організацією IANA (Internet Address Naming Authority) під номерами 51 та 50, відповідно.

**Протокол автентифікаційного заголовку (Authentication Header, AH).**

Описаний в RFC-4302, IP Authentication Header / S. Kent. – December 2005. Протокол AH передбачає: автентифікацію джерела даних, перевірку їхньої цілісності і справжності після одержання, захист від нав'язування повторних повідомлень.

**Протокол інкапсулюючого захисту вмісту (Encapsulating Security Payload, ESP).** Описаний в RFC-4303, IP Encapsulating Security Payload (ESP) / S. Kent. – December 2005. Протокол ESP крім усіх функцій протоколу AH забезпечує ще й криптографічне закриття пакетів повідомлень.

**Протокол узгодження параметрів віртуального каналу й керування ключами (англ. – Internet Security Association Key Management Protocol, ISAKMP).** Описаний в RFC-4306, Internet Key Exchange (IKEv2) Protocol / C. Kaufman, Ed. – December 2005. Призначений для попереднього узгодження алгоритмів та їхніх параметрів сторонами, що взаємодіють за протоколами AH та ESP. Забезпечує створення сторонами, що взаємодіють, спільного контексту, елементи якого в подальшому вони можуть вільно використовувати.

- **Середній рівень** – криптографічні алгоритми, що використовуються в протоколах AH та ESP, а також певні алгоритми узгодження і керування ключами, які використовує протокол ISAKMP
- **Нижній рівень** – так званий "домен інтерпретації" (Domain of Interpretation, DOI). Це, фактично, база даних, яка містить інформацію про усі протоколи і алгоритми, що застосовуються в IPSec, а також про їхні параметри, ідентифікатори тощо. Наявність такої бази пояснюється тим, що відкрита архітектура IPSec припускає застосування протоколів і алгоритмів, які не розроблялись для неї чи з урахуванням її вимог. Необхідною умовою застосування сторонніх алгоритмів автентифікації або шифрування (наприклад, тих, що відповідають національним стандартам) є реєстрація їх у домені інтерпретації

**Асоціації захисту (SA)** це контекст, у якому взаємодіють сторони, що використовують технологію IPSec, визначають терміном "асоціація захисту" (Security Association, SA). Асоціація захисту функціонує на основі угоди, що заключається сторонами. Елементами асоціації захисту є:

- учасники зв'язку: IP-адреси відправника й одержувача;
- криптографічний алгоритм;

- порядок обміну ключами;
- розміри ключів;
- термін дії ключів;
- алгоритм автентифікації.

Асоціації захисту утворюються відповідно до протоколу ISAKMP

**Автентифікаційний заголовок (AH)**

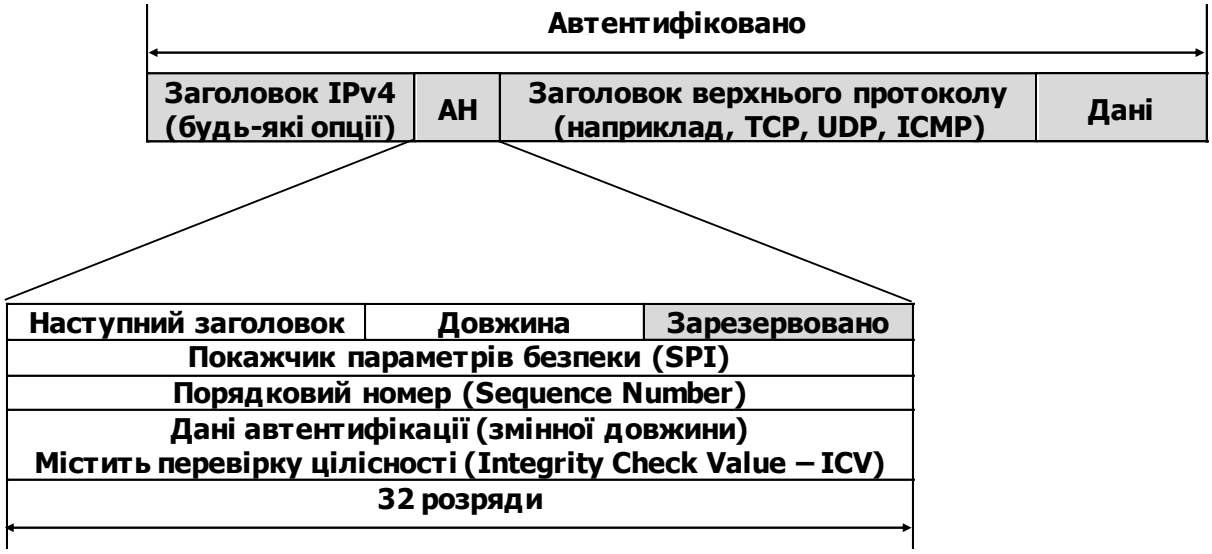



Рисунок 5.10 – Структура Заголовку автентифікації (AH)

- Поле SPI (Security Parameters Index) – це “покажчик параметрів безпеки”. Це 32-розрядне число, що вказує на протоколи захисту, що використовуються. В ньому включені індекси алгоритмів і типи ключів. Фактично, воно визначає асоціацію захисту.
- Порядковий номер (Sequence Number) визначає кількість пакетів, що відправлені, і забезпечує захист від хибного повторення даних

**Протокол інкапсулюючого захисту (ESP).**

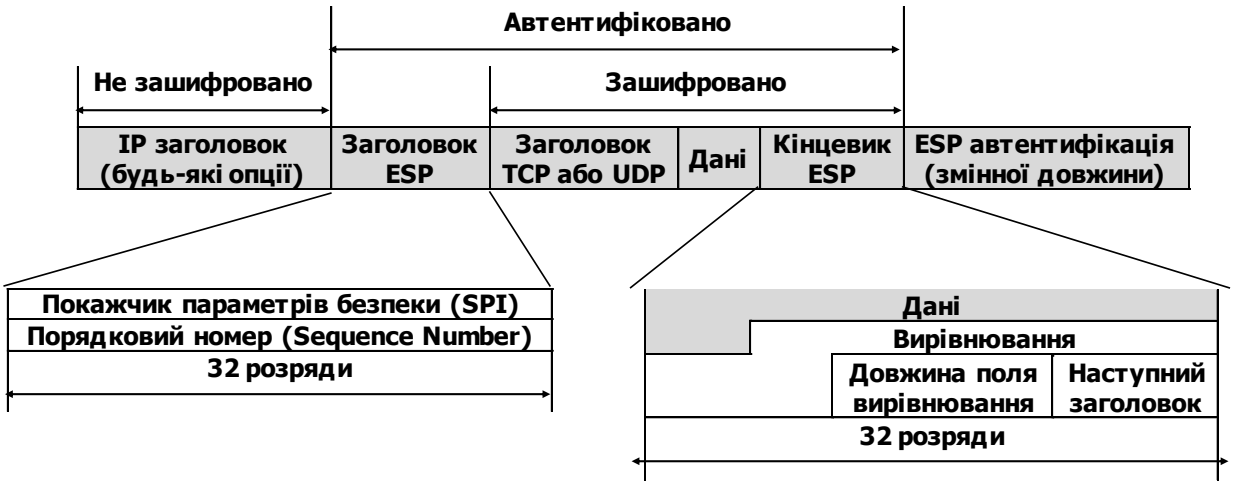



Рисунок 5.11 – Структура протоколу інкапсулюючого захисту (ESP)

Протокол ESP забезпечує шифрування IP-інформації на рівні пакетів. Передбачено використання різних алгоритмів шифрування. Протокол ESP забезпечує автентифікацію даних із застосуванням різних алгоритмів автентифікації

Слід звернути увагу на таке:

- заголовок ESP розташований між заголовком IP та рештою вмісту пакета;
- поля покажчика SPI та порядкового номера виконують ту ж функцію, що й у заголовку AH;
- поле заголовку TCP (або UDP, або іншого протоколу), дані та кінцевик (трейлер) ESP зашифровані;
- поле вирівнювання має змінну довжину в діапазоні 0-255 біт, і забезпечує, по-перше, що поле “Наступний заголовок” закінчується на межі 32-розрядного слова, а по-друге, що розмір зашифрованої частини кратний розміру блоку застосованого алгоритму шифрування;
- ESP забезпечує автентифікацію даних у тому ж порядку, що й AH.

IPSec може функціонувати у двох режимах: *транспортний* і *тунельний*. У транспортному режимі шифруються тільки дані IP–пакету, а початковий заголовок зберігається. Транспортний режим, як правило, використовується для встановлення з’єднання між кінцевими пристроями. Він може також використовуватися між шлюзами для захисту тунелів, організованих іншим способом. У тунельному режимі шифрується увесь початковий IP– пакет: дані, заголовок, маршрутна інформація, а потім він вставляється в поле даних нового пакету, тобто відбувається інкапсуляція. Тунельний режим може використовуватися для підключення видалених комп'ютерів до віртуальної приватної мережі або для організації безпечної передачі даних через відкриті

канали зв’язку між шлюзами для об’єднання різних частин віртуальної приватної мережі.

Інкапсуляція IPSec для тунельного режиму.

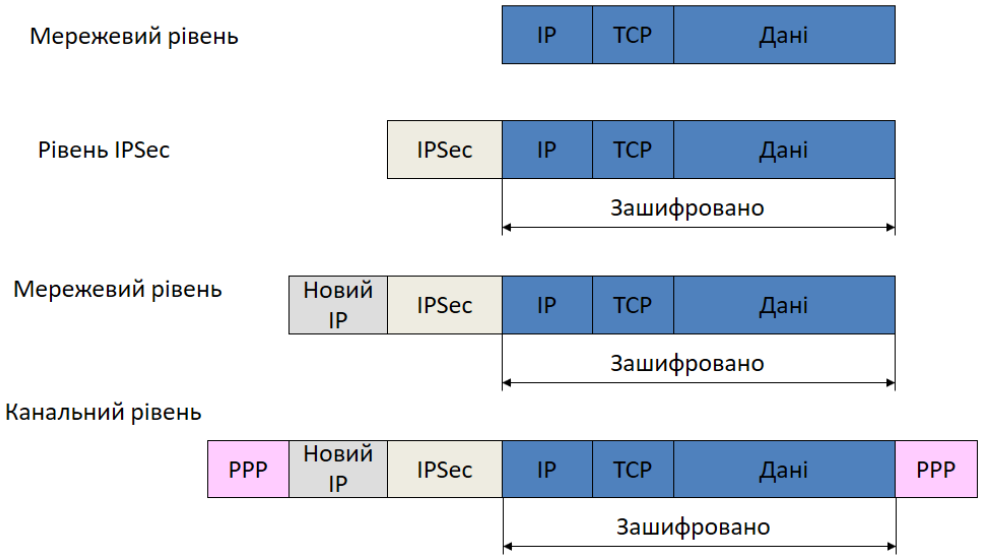



Рисунок 5.12 – Інкапсуляція IPSec для тунельного режиму

Інкапсуляція IPSec для транспортного режиму

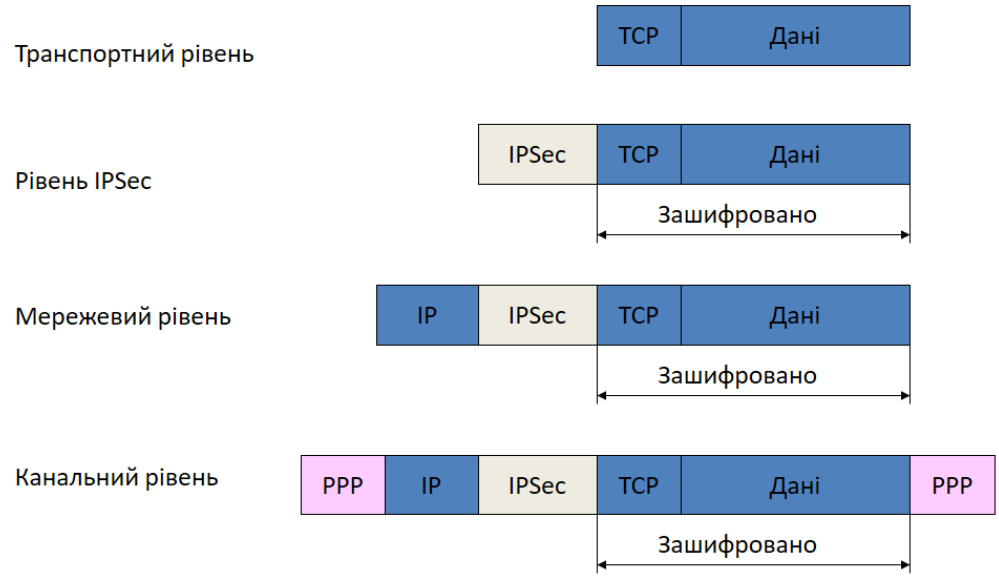



Рисунок 5.13 – Інкапсуляція IPSec для транспортного режиму

Інкапсуляція з аутентифікацією (ESP)

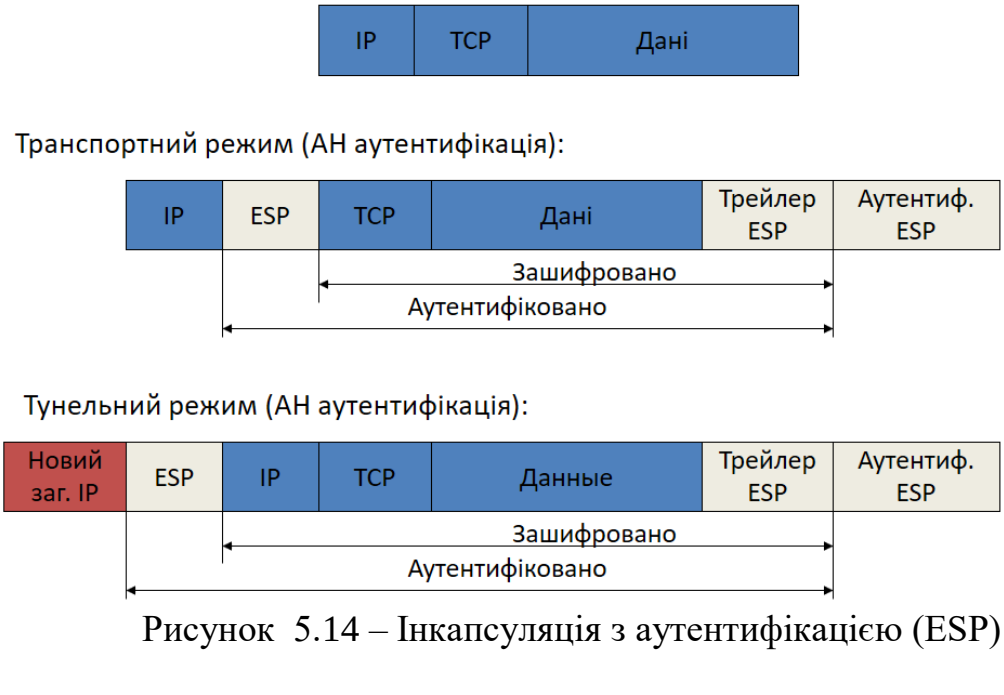



Рисунок 5.14 – Інкапсуляція з аутентифікацією (ESP)

**Обмін ключами**

В IPSec застосовуються два способи передачі ключів:

- Вручну.

Ключі вручну завантажуються у відповідні пристрої IPsec безпосередньо на об'єктах. Шифруванню ці ключі не піддаються, вони або передаються системному адміністратору особисто, або надсилаються поштою. Введення ключів вручну виправдано лише у невеликій мережі

- Шляхом обміну через IP-мережу (Internet Key Exchange, IKE).

Коли масштаби мережі зростають, виникає потреба в механізмі створення асоціацій захисту за вимогою (SA on Demand). За створення асоціацій захисту відповідає протокол ISAKMP, який описує базові технології, але не специфікує конкретні алгоритми. Для обміну ключами можуть застосовуватись окремі протоколи. Зокрема протокол Oakley, що використовує алгоритм Діффі-Хелмана

Поєднання протоколів ISAKMP та Oakley було відомо як специфікації ISAKMP/Oakley, тепер воно отримало назву протоколу IKE

**Протокол IKE.**

Призначений для узгодження параметрів асоціацій захисту, що створюються, і для автентифікованого обміну ключами, якими будуть користуватись учасники цих асоціацій. Дозволяє утворити між двома учасниками обміну (IKE SA) автентифікований захищений тунель, за яким будуть узгоджуватись параметри асоціації захисту, що створюється для IPSec. Протокол діє на базі UDP, передбачає використання порту 500.

Може функціонувати у трьох режимах:

- Основний режим (Main Mode) застосовується, коли дві сторони вперше встановили зв’язок, щоби узгодити параметри асоціації захисту, яка забезпечить конфіденційність їх подальшого обміну
- Активний режим (Aggressive Mode) є скороченою версією основного режиму, має те ж призначення, що й основний режим, і може використовуватись замість нього
- Швидкісний режим (Quick Mode) застосовується, коли асоціація захисту вже створена в результаті використання основного або активного режиму, але існує необхідність в узгодженні функцій захисту або обміну новими ключами. Оскільки захищений канал був утворений ще до застосування швидкісного режиму, останній забезпечує надійний захист без додаткових витрат, які притаманні основному або активному режиму

**Автентифікація у протоколі IKE.**

Протокол IKE передбачає кілька способів автентифікації:

- Коли спільно використовуються одні й ті ж ключі. Всі хост-системи (або шлюзи VPN) володіють одними й тими ж таємними ключами. IKE автентифікує різних учасників обміну по хешу ключа.
- При використанні криптографії з відкритим ключем. Кожна сторона генерує випадкове число і шифрує його відкритим ключем іншої сторони. Автентифікація відбувається, коли інша сторона може розрахувати хеш-функцію цього випадкового числа і надіслати результат першій стороні.
- Технології цифрового підпису. Кожний пристрій «підписує» набори даних, що відсилає іншій стороні. Цей метод подібний до шифрування відкритим ключем, але додатково забезпечує захист від відмовлення від авторства.

При використанні асиметричної криптографії (цифровий підпис, шифрування відкритим ключем), необхідно використання цифрових сертифікатів, що підтверджують взаємну відповідність і справжність відкритих та секретних ключів. Протокол IKE дозволяє отримати доступ до сертифікату в односторонньому порядку або у формі обміну при виконанні сторонами процедури IKE

**Етапи встановлення з’єднання за технологією L2TP/IPSec**

*Перший етап* – встановлення з’язку. Між LNS та LAC ініціюється PPP сесія:

- встановлення контрольного з'єднання за допомогою LCP;
- автентифікація тунелю;
- встановлення сесії;
- встановлення вхідного дзвінка;
- встановлення вихідного дзвінка.

*Другий етап* – інкапсуляція L2TP**.** Кадр PPP вкладається в оболонку з заголовком L2TP і заголовком UDP.

*Третій етап* – встановлення ISAKMP–тунелю. Клієнт та сервер автентифікують один одного і домовляються про параметри установки спеціального з'єднання, призначеного тільки для обміну інформацією про бажаних алгоритмах шифрування і інші деталі майбутнього IPSec –тунелю. Насамперед узгоджуються хеши і алгоритми шифрування, далі йде обмін ключами Діффі–Хеллмана (DH), потім відбувається процес автентифікації.

*Четвертий етап* – встановлення політики безпеки. Клієнт і сервер погоджують загальну політику IPSec SA, отримують загальні секретні ключі для алгоритмів протоколів IPSec (AH або ESP), встановлюють IPSec –тунель.

*П'ятий етап* – інкапсуляція IPSec. L2TP–повідомлення вкладається в оболонку з заголовком і трейлером IPSec ESP, трейлером перевірки автентичності IPSec , що забезпечує цілісність повідомлення та перевірку справжності, і заголовком IP. Відбувається шифрування даних.

## 5.6 Протоколу SSL/TLS та його реалізація OpenVPN

Transport Layer Security (далі – TLS) – криптографічний протокол, що забезпечує захищену передачу даних між вузлами в мережі Інтернет. Заснований на SSL–протоколі версії 3.0. TLS використовує асимметричное шифрування для автентифікації, симетричне шифрування для конфіденційності та коди автентичності повідомлень для збереження цілісності повідомлення. Ядром протоколу є технологія комплексного використання асиметричних і симетричних криптосистем.

В основу протоколу, представленої на рисунку 5.15 ввійшли 2 протоколи: протокол запису та протокол діалогу, який в свою чергу, складається з трьох підпротоколів (протокол зміни специфікації шифру, протокол сповіщення та протокол рукопотискання).

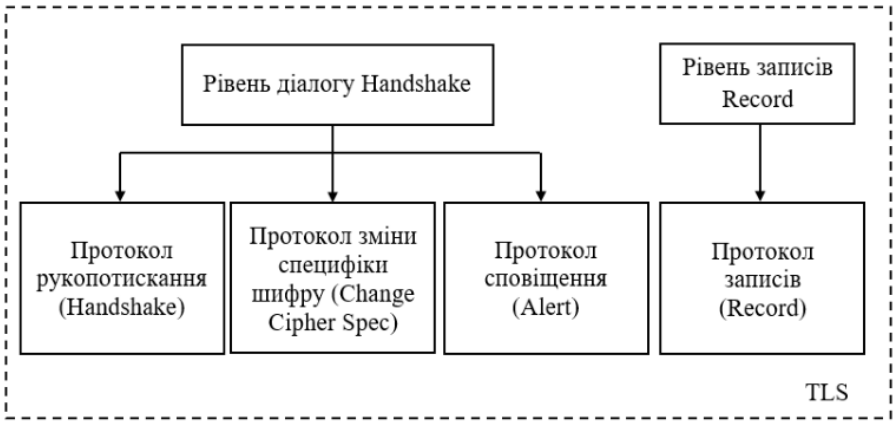



Рисунок 5.15 – Архітектура протоколу TLS

**Структура протоколу Record SSL/TLS**

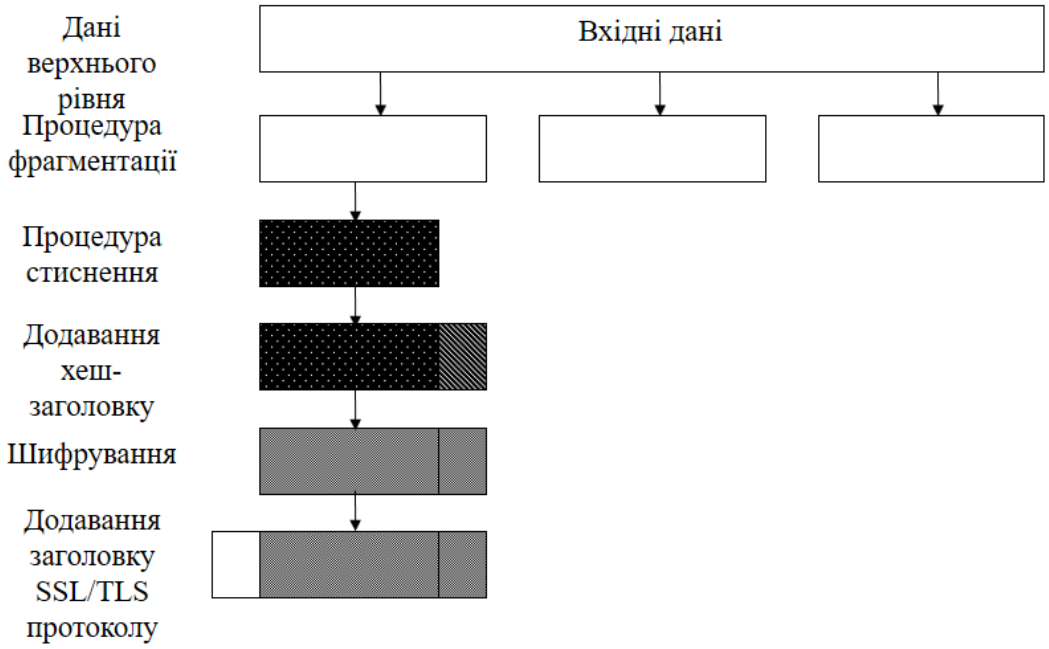



Рисунок 5.16 – Структура протоколу Record SSL/TLS

Технологія TLS VPN базується на **OpenVPN**. Створений в 2002 році, OpenVPN – це інструмент з відкритим вихідним кодом, який використовується для побудови site–to–site VPN мереж з використанням TLS протоколу. Він виконує роль безпечного тунелю для передачі даних через виділений 1194 порт з використанням протоколів транспортного рівня TCP та UDP. UDP є більш ефекитвним з тієї причини, що через тунель проходить трафік мережевого рівня і вище, якщо використовується TUN–з'єднання, або трафік канального рівня і вище, якщо використовується TAP–з'єднання. Це означає, що OpenVPN для клієнта виступає протоколом канального або навіть фізичного рівня, а значить, надійність передачі даних може забезпечуватися вищестоящими по OSI рівнями, якщо це необхідно, що робить протокол UDP за своєю концепцією найбільш близьким до OpenVPN. Якщо ж налаштувати тунель на роботу з TCP, сервер в типовому випадку буде отримувати TCP–сегменти OpenVPN, які містять інші TCP–сегменти від клієнта. В результаті в ланцюзі виходить подвійна перевірка на цілісність інформації, що абсолютно не має сенсу, тому що надійність не підвищується, а швидкості з'єднання знижуються.

**Компоненти мережі OpenVPN:**

- засвідчувальний центр CA. Видає сертифікати за запитом вузлів мережі VPN, підписані сертифікатом засвідчувального центру. Надає вузлам мережі VPN свій власний сертифікат для перевірки засвідчувальної сторони. Керує списком відгуків сертифікатів CRL.
- сервер OpenVPN. ПЗ сервера OpenVPN створює тунель всередині

незахищеної мережі, наприклад, Інтернету. Цей тунель забезпечує безпечний зашифрований трафік між вузлами – учасниками обміну даними в мережі OpenVPN.

- клієнт OpenVPN. ПЗ клієнта OpenVPN встановлюється на всі вузли, яким необхідний захищений канал передачі даних з сервером OpenVPN. За відповідного налаштування сервера OpenVPN можлива захищена передача даних між клієнтами OpenVPN, а не тільки між клієнтами і сервером OpenVPN.
- сертифікати (публічні ключі) X.509. Сертифікати X.509 являють собою публічні ключі, завірені засвідчуваним центром CA. Вони використовуються для зашифровування даних. Факт запевнення сертифіката засвідчуваним центром CA дозволяє ідентифікувати сторону, передаючу зашифровані дані.
- приватні ключі. Приватні ключі секретні. Вони повинні створюватися і зберігатися на кожному вузлі мережі OpenVPN, призначені для розшифрування даних і ніколи не повинні передаватися по мережі.
- список відкликаних сертифікатів CRL. Містить список сертифікатів, які втратили довіру. Він створюється і редагується на вузлі засвідчувального центру CA. Щоб відключити вузол від мережі, досить занести його сертифікат в список CRL.

Для забезпечення безпеки керуючого каналу і потоку даних протокол TLS використовує бібліотеку OpenSSL. Це дозволяє задіяти весь набір алгоритмів шифрування, доступних в даній бібліотеці. Також може використовуватися пакетна автентифікація HMAC, для забезпечення більшої безпеки, і апаратне прискорення для поліпшення продуктивності шифрування. Інкапсуляцію TLS пакетів з використанням OpenVPN зображено на рисунку 5.17, де зображено утворений OpenVPN тунель, що інкапсулюється в UDP–сегмент, як певне корисне навантаження.

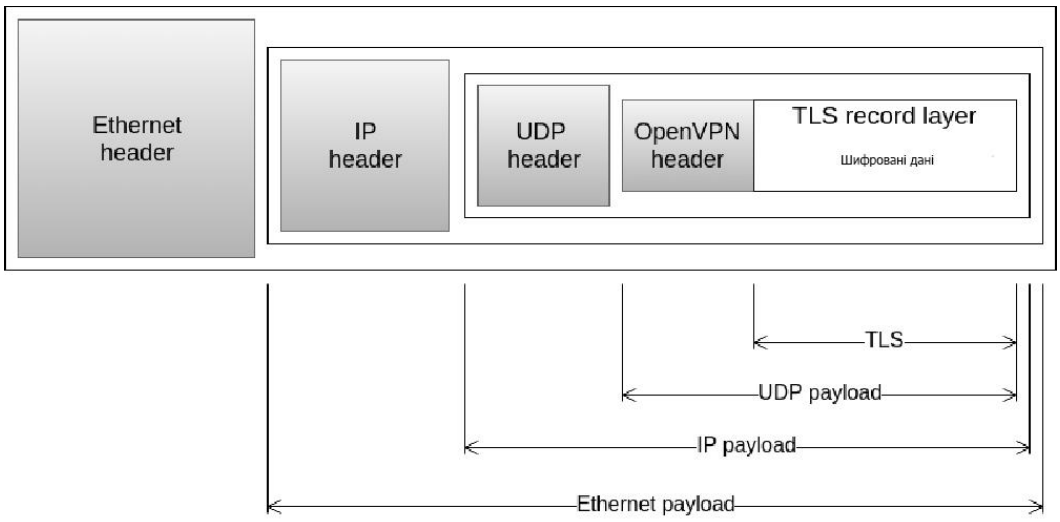



Рисунок 5.17 – Інкапсуляція TLS пакетів з використанням OpenVPN

**Етапи встановлення з'єднання технології TLS на базі OpenVPN**

*Перший етап* – ініціація з'єднання. Клієнт підключається до сервера через тунель OpenVPN. Це супроводжується трьома повідомленнями, останнє з яких вказує на встановлення OpenVPN тунелю.

*Другий етап* – узгодження параметрів з'єднання. Перед тим, як почати обмін даними через TLS, клієнт і сервер повинні узгодити параметри з'єднання, а саме: версія використовуваного протоколу, спосіб шифрування даних, а також перевірити сертифікати, якщо це необхідно. Схема початку з'єднання називається TLS Handshake, яка включає вирішення наступних завдань:

- аутентифікація сторін;
- узгодження криптографічних алгоритмів і алгоритмів стиснення, які будуть використовуватися при захищеному інформаційному обміні;
- формування загального секретного майстер–ключа;
- генерація на основі сформованого майстер–ключа загальних секретних сеансових ключів для криптозахисту інформаційного обміну.

*Третій етап* – обмін ключами. Найчастіше в TLS використовується обмін ключами по алгоритму RSA. Клієнт генерує симетричний ключ, підписує його за допомогою відкритого ключа сервера і відправляє його на сервер. У свою чергу, на сервері ключ клієнта розшифровується за допомогою закритого ключа, після чого обмін ключами оголошується завершеним.

Слід відзначити, що шифрування з відкритим ключем використовується тільки в процедурі TLS Handshake під час початкового налаштування з'єднання. Після настройки тунелю в справу вступає симетрична криптографія, і спілкування в межах поточної сесії зашифровано саме встановленими симетричними ключами. Це необхідно для збільшення швидкодії, так як криптографія з відкритим ключем вимагає значно більше обчислювальної потужності.

*Четвертий етап* – автентифікація**.** Автентифікація є невід'ємною частиною кожного TLS з'єднання. Розглянемо найпростіший процес автентифікації між клієнтом і сервером:

1. Сервер і клієнт генерують власні відкриті і закриті ключі.
2. Обмін відкритими ключами.
3. Клієнт генерує повідомлення, шифрує його своїм закритим ключем і відправляє на сервер.
4. Сервер використовує отриманий від клієгта ключ, щоб розшифрувати повідомлення і таким чином перевіряє справжність отриманого повідомлення.

У протоколі TLS дані ланцюга довіри засновані на сертифікатах автентичності, що надаються спеціальними органами, званими центрами сертифікації – certificate authorities (далі – CA). Центри сертифікації проводять

перевірки і, якщо виданий сертифікат скомпрометований, то даний сертифікат відгукується.

З виданих сертифікатів складається ланцюжок довіри. Коренем його є так званий "Root CA certificate" – сертифікат, підписаний великим центром, довіра до якого незаперечна.

5.7 Захищений протокол віддаленого доступу SSH.

**SSH** - мережевий протокол прикладного рівня, що дозволяє проводити віддалене управління операційною системою і туннелювання TCP-з'єднань. Шифрує весь трафік, включаючи і передані паролі. SSH допускає вибір різних алгоритмів шифрування. SSH-клієнти та SSH-сервери доступні для більшості мережевих операційних систем.

SSH підтримує можливість аутентифікації за RSA-ключом, що забезпечує максимальний рівень безпеки для каналу передачі даних, а також двофакторну аутентифікацію віддалених користувачів.SSH дозволяє безпечно передавати в незахищеною середовищі практично будь-який інший мережевий протокол. Таким чином, можна не тільки віддалено працювати на комп'ютері через командну оболонку, але і передавати по шифрованому каналу звуковий потік або відео (наприклад, з веб-камери) . Також SSH може використовувати стиснення переданих даних для подальшого їх шифрування, що зручно, наприклад, для віддаленого запуску клієнтів X Window System.

Перша версія протоколу, SSH-1, була розроблена в 1995 році дослідником Тату Улененом з Технологічного університету Хельсінкі, Фінляндія. SSH-1 був написаний для забезпечення конфіденційності, ніж протоколи rlogin, telnet і rsh. У 1996 році була розроблена безпечніша версія протоколу, SSH-2, несумісна з SSH-1. Протокол набув ще більшої популярності, і до 2000 року у нього було близько двох мільйонів користувачів. В даний час під терміном «SSH» зазвичай мається на увазі саме SSH-2, тому що перша версія протоколу з огляду істотних недоліків зараз практично не застосовується. У 2006 році протокол був затверджений робочою групою IETF як Інтернет-стандарт.

Поширені дві реалізації SSH: пропріетарна (комерційна) та безкоштовна. Безкоштовна реалізація називається OpenSSH. До 2006 року 80% комп'ютерів мережі Інтернет використовувало саме OpenSSH. Пропріетарна реалізація розробляється організацією SSH Inc., Вона безкоштовна для некомерційного використання. Ці реалізації містять практично однаковий набір команд.

Протокол SSH-2, на відміну від протоколу telnet, стійкий до атак прослуховування трафіку («сніфінг»), але нестійкий до атак «людина посередині». Протокол SSH-2 також стійкий до атак шляхом приєднання посередині (англ. session hijacking) — неможливо включитися у вже встановлену сесію або перехопити її. Для запобігання атак «людина посередині» при підключенні до хосту, ключ якого ще не відомий клієнту, клієнтське ПО показує користувачеві «зліпок ключа» (key fingerprint). Рекомендується ретельно

перевіряти показуваний клієнтським ПО «зліпок ключа» (key fingerprint) зі зліпком ключа сервера, бажано отриманим по надійним каналах зв'язку або особисто.

Підтримка SSH реалізована у всіх UNIX-подібних системах, і на більшості з них в числі стандартних утиліт присутні клієнт і сервер ssh. Існує безліч реалізацій SSH-клієнтів і для не-UNIX ОС. Велику популярність протокол отримав після широкого розвитку аналізаторів трафіку і способів порушення роботи локальних мереж, як альтернативне небезпечному протоколу Telnet рішення для управління важливими вузлами.

Для роботи по SSH потрібен SSH-сервер і SSH-клієнт. Сервер прослуховує з'єднання від клієнтських машин і при встановленні зв'язку виробляє аутентифікацію, після чого починає обслуговування клієнта. Клієнт використовується для входу на віддалену машину і виконання команд. Для з'єднання сервер і клієнт повинні створити пари ключів — відкритих і закритих — і обмінятися відкритими ключами. Зазвичай використовується також і пароль.

**Основні функції SSH:**

- Безпечні команди доступу до хосту. SSH дає можливість виконувати безпечні команди доступу до хосту, такі як SSH (віддалена оболонка), slogin (віддалений вхід в систему), scp (віддалене копіювання);
- X11 Forwarding. SSH надає вбудований механізм для виконання віддалених клієнтів X Window.
- Перенаправлення портів. SSH може переадресовувати портів, передаючи трафік з одного порту однієї машини на інший порт іншої машини. При цьому переданий трафік шифрується.

**Забезпечення безпеки SSH.**

Безпека протоколу досягається використанням декількох рішень, які зводять до мінімуму ризик використання з'єднання:

- Шифрування з'єднання, яке може виконуватися одним з методів, обраних в процесі переговорів. Шифрування з'єднання не дозволяє просто перехопити і використовувати трафік. Вибір алгоритму шифрування робить систему більш гнучкою, дозволяючи не використовувати алгоритми, в яких виявлені слабкі місця або які не може підтримувати одна зі сторін;
- Аутентифікація сервера виконується при будь-якому з'єднанні. Це не дозволяє виконати заміну сервера або підміну трафіку;
- Аутентифікація клієнта може виконуватися одним з декількох доступних способів. Це з одного боку може підвищити надійність аутентифікації, з іншого - робить систему більш гнучкою і спрощує її використання;
- Перевірка цілісності пакетів дозволяє відстежити будь-які незаконні зміни в трафіку з'єднання. При виявленні таких змін, з'єднання негайно розривається;

- Тимчасові параметри аутентифікації не дозволяють скористатися даними з'єднання в тому, випадку, якщо через деякий час після перехоплення воно все-таки було розшифровано. Старіння зазвичай відбувається через годину;

SSH захищає від:

- Підміни IP-адрес (IP-спуфинг), коли віддалений хост посилає пакети від імені іншого хоста;
- Підміни DNS-записів (DNS-спуфинг), коли змінюється запис на сервері DNS і в результаті з'єднання встановлюється з бажаним хостом, а з тим, на який вказує новий запис;
- Перехоплення відкритих паролів і інших даних, які передаються у відкритому вигляді і будь-хто, хто має фізичний доступ до каналу, може їх впізнати.

**Архітектура протоколу SSH.**

SSH складається з трьох основних елементів:

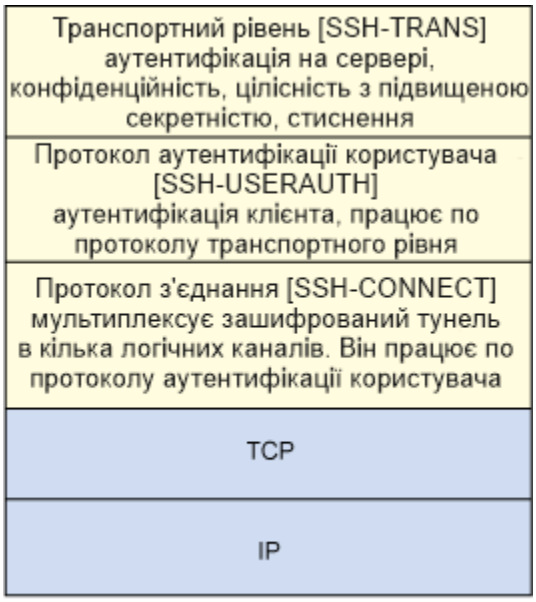



Рисунок. 5.18 - Архітектура протоколу SSH

**Архітектура протоколу SSH.**

**SSH протокол транспортного рівня** (Transport Layer Protocol). Він надає послуги серверної аутентифікації, конфіденційності та цілесності. Цей протокол також може забезпечувати стиснення й працює поверх будь-якого надійного протоколу транспортного рівня (наприклад, TCP).

**SSH протокол аутентифікації користувачів** (User Authentication Protocol) забезпечує аутентифікацію користувача на стороні клієнта й працює поверх SSH Transport Layer Protocol

**SSH протокол з'єдання** (Connection Protocol) відповідає за мультеплексування захищеного тунелю SSH Transport Layer й User Authentication

Protocols в кілька логічних каналів. Ці логічні канали можуть бути використані для широкого спектру цілей: захист інтерактивних сесій оболонки, перенаправлення TCP портів, проведення з'єднань X11.

**Архітектура системи SSH**

SSH має близько десятка різних взаємодіючих компонентів. Під «компонентом» не обов'язково мають на увазі «програму». SSH також має ключі, сеанси та інші складові. Рисунок 5.19 ілюструє основні компоненти та їх зв’язки один з одним.

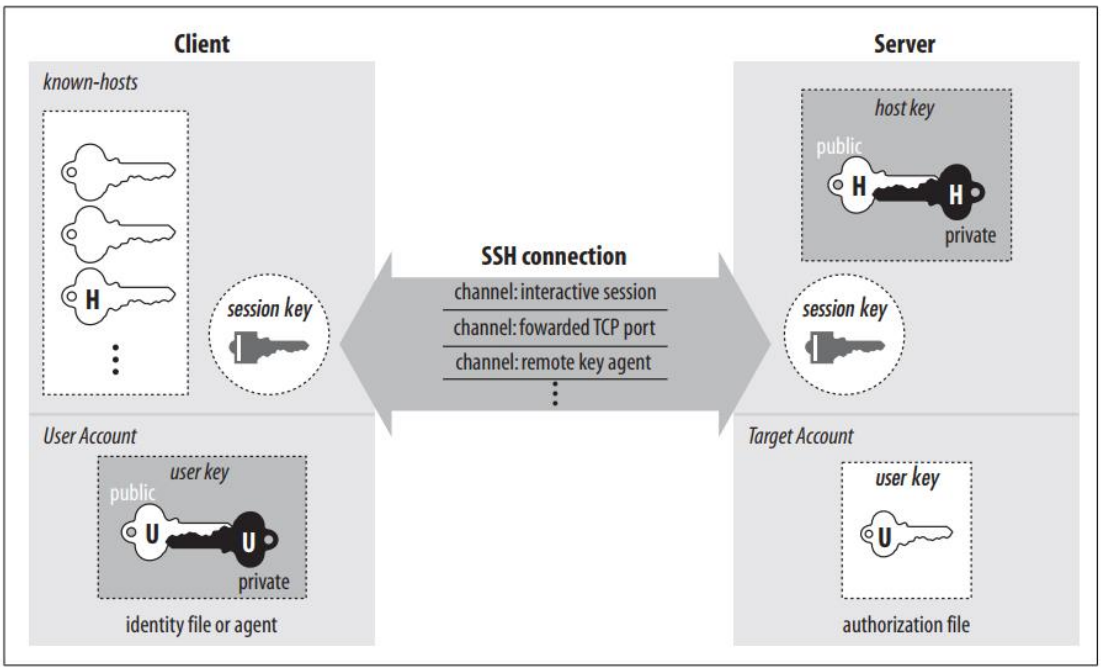



Рисунок 5.20 – Архітектура системи SSH

**Сервер.** Програма, яка дозволяє вхідні SSH-з'єднання з машиною, обробляючи автентифікацію, авторизацію тощо. У більшості реалізацій Unix SSH сервер є sshd.

**Клієнт**. Програма, яка підключається до серверів SSH і робить запити, такі як «увійти в систему» або «скопіювати цей файл». У OpenSSH і Tectia основними клієнтами є ssh, scp і sftp.

**Сесія**. Постійний зв'язок між клієнтом та сервером. Він починається після того, як клієнт успішно автентифікується на сервері, і закінчується, коли з'єднання припиняється. Сесії можуть бути інтерактивними або пакетними.

**Ключ.** Відносно невеликий обсяг даних, як правило, від десятків до 1000 або 2000 біт, використовуваних як параметр для криптографічних алгоритмів, таких як шифрування або автентифікація повідомлень. Ключ певним чином прив'язує роботу алгоритму до власника ключа: при шифруванні він гарантує, що лише той хто володіє цим ключем (або пов'язаний з ним), може розшифрувати повідомлення; при автентифікації він дозволяє згодом переконатися, що власник

ключа фактично підписав повідомлення. Існує два види ключів: симетричний або секретний ключ, а також асиметричний або відкритий ключ. Асиметричний ключ складається з двох частин: публічного та приватного компонентів. SSH має кілька типів ключів, які узагальнено в Таблиці 5.1:

Таблиця 5.1

Ключі SSH

| Ім’я | Час життя | Ким створено | Тип | Призначення |
|---|---|---|---|---|
| Ключ користувача | Стійкий persistent | користувач | публічний | ідентифікація користувача до сервера |
| Ключ хоста | стійкий | адміністратор | публічний | ідентифікація сервера/машини |
| Сесійний ключ | 1 сесія | клієнт (та сервер) | закритий | захист з’єднання |
| Ключ сервера | 1 година | сервер | публічний | захист сесійного ключа (лише в SSH1) |

**Ключ користувача.** Стійкий асиметричний ключ, який клієнти використовують як доказ ідентичності користувача. (У одного користувача може бути багато ключів)

**Ключ хоста**. Стійкий асиметричний ключ, який використовується сервером як підтвердження своєї ідентичності, а також клієнтом при доведенні ідентичності свого хоста як частини автентифікації на базі хоста. Якщо машина працює на одному сервері SSH, хост-ключ також однозначно ідентифікує апарат. (Якщо на машині працює декілька серверів SSH, у кожного може бути інший хост-ключ або вони можуть бути спільними.)

**Ключ сесії**. Симетричний ключ для шифрування зв'язку між клієнтом SSH та сервером. Дві сторони поділяються безпечним способом під час налаштування з'єднання SSH, щоб підслуховувач не міг його виявити. Обидві сторони мають ключ сеансу, який вони використовують для шифрування своїх комунікацій. Коли сеанс SSH закінчується, ключ знищується. З'єднання SSH має кілька ключів сеансу: у кожному напрямку (сервер до клієнта та клієнт - сервер) є ключі для шифрування та інші для перевірки цілісності.

**Ключ сервера**. Тимчасовий асиметричний ключ, що використовується в протоколі SSH-1. Він регенерується сервером через регулярні проміжки часу (за замовчуванням щогодини) та захищає сеансовий ключ. Цей ключ ніколи не зберігається на диску, а його приватний компонент ніколи не передається через з'єднання в будь-якій формі.

**Генератор ключів**. Програма, яка створює стійкі ключі (ключі користувачів та хост-ключі) для SSH.

**База даних** відомих хостів.

**Колекція ключів хоста**. Клієнти та сервери посилаються на цю базу даних для автентифікації один одного.

**Агент**. Програма, яка зберігає в пам'яті ключі користувачів, тому користувачі не повинні повторно вводити свої парольні фрази. Агент відповідає на запити щодо операцій, пов’язаних із ключами, таких як підписання автентифікатора, але він не розголошує самі ключі.

**Підписувач**. Програма, яка підписує пакети автентифікації на основі хостів.

**Файл конфігурації**. Набір налаштувань для адаптації поведінки клієнта або сервера SSH.

Не всі ці компоненти потрібні для впровадження SSH. Звичайно сервери, клієнти та ключі є обов'язковими, але у багатьох реалізаціях немає агента, а деякі навіть не включають генератор ключів.

Для роботи по SSH потрібен SSH-сервер і SSH-клієнт. Сервер прослуховує з'єднання від клієнтських машин і при встановленні зв'язку виробляє аутентифікацію, після чого починає обслуговування клієнта. Клієнт використовується для входу на віддалену машину і виконання команд.

**Аутентифікація сервера** відбувається за допомогою інфраструктури відкритих ключів. Клієнт, який хоче встановити з'єднання з сервером шифрує дані відомим йому відкритим ключем сервера і відправляє їх сервера. Сервер повинен розшифрувати їх за допомогою відомого тільки йому секретного ключа, і відправити їх назад. Так клієнт може бути впевнений в тому, чи є хост тим, за кого себе видає.

Аутентифікація сервера по протоколу SSH виконується за допомогою інфраструктури відкритих ключів. Відкритий ключ сервера клієнт отримує при першому з'єднанні з ним.

Аутентифікація сервера дає можливість не покладатися на службу імен та маршрутизацію пакетів. У тому випадку, якщо порушника вдалося підмінити запис в DNS або перенаправити IP-пакети на свій хост, аутентифікація не пройде, оскільки хост не володіє необхідними секретними ключами.

**Аутентифікація клієнта SSH.**

Методи аутентифікації клієнтів, які використовує SSH:

- Хост-аутентифікація
- Аутентифікація за допомогою відкритих ключів
- Kerberos-аутентифікація
- Парольная аутентифікація.

*За хостом.* Метод аналогічний використовуваному в r-командах. У тому випадку, якщо з'єднання встановлюється з привілейованого порту, і файл .rhosts дозволяє вхід в систему, він дозволяється. Цей метод є потенційно небезпечним, рекомендується не використовувати його. Для підвищення рівня своєї безпеки метод може бути доповнений RSA-аутентифікацией клієнтського хоста.

*Відкритий ключ.* Клієнт відправляє серверу відкритий ключ. Якщо сервер знає його, він просить клієнта довести, що той знає і секретний ключ теж. Якщо клієнт може це довести, значить аутентифікація вважається успішною.

*Керберос*. Аутентифікація проводиться за схемою v5 Kerberos.

*Пароль.* У самому крайньому випадку, якщо не вдалося провести аутентифікацію не одним з перерахованих способів, використовується традиційна аутентифікація за допомогою пароля. Принцип аутентифікації аналогічний тому, який, наприклад, використовується в Telnet з тією різницею, що пароль передається по зашифрованому каналу.

**Алгоритм встановлення з'єдення по протоколу SSH.**

Алгоритм встановлення з'єдення по протоколу SSH можна розділити на три рівні, кожен з яких розташовується над попереднім: транспорт (відкриття захищеного каналу), аутентифікація, підключення.

Алгоритм встановлення зв’язку по протоколу SSH:

а) Установка TCP-з’єднання.

На цьому етапі відбувається мережеве підключення клієнта до сервера на TCP-порт, зазначений в опції Port (за замовчуванням: 22) у файлі конфігурації сервера /etc/ssh/sshd_config.

б) Відкриття захищеного каналу

1) Обмін ідентифікаційними даними.

Після встановлення TCP-з’єднання, клієнт і сервер (далі по тексту – сторони) обмінюються версіями SSH-протоколу та іншими допоміжними даними, необхідними для з’ясування сумісності протоколів і для вибору алгоритмів роботи.

2) Вибір алгоритмів обміну ключами шифрування, стиснення і т. п.

При роботі SSH використовується досить багато алгоритмів, одні з них використовуються для шифрування, другі для обміну ключами, треті для стиснення передаваних даних і т. п. На цьому кроці сторони надсилають одна одній списки підтримуваних алгоритмів, найбільший пріоритет мають алгоритми на початку кожного списку. Потім порівнюють алгоритми отриманих списках з алгоритмами, наявними в системі, і вибирають перший збігся в кожному списку.

3) Отримання сесійного ключа шифрування

Процес отримання сесійного ключа може відрізнятися в залежності від версії алгоритму, але в загальних рисах зводиться до наступного:

Сервер відсилає клієнтові свій ключ (DSA, RSA або т. п. за домовленістю між сторонами, виробленими в попередніх пунктах).

Якщо клієнт здійснює з'єднання з даним сервером вперше (про що говорить відсутність запису у файлі /home/username/.ssh/known_hosts у клієнта), то користувачеві буде поставлено питання про довіру ключа сервера. Якщо ж з'єднання з даним сервером вже встановлювалося раніше, то клієнт порівнює присланий ключ з ключем, записаним в /home/username/.ssh/known_hosts. Якщо ключі не співпадають, то користувач отримає попередження про можливу спробу злому.

Як тільки клієнт визначився з довірою до ключа сервера, за допомогою однієї з реалізацій алгоритму Діффі-Хеллмана клієнт і сервер генерує сеансовий ключ, який буде використовуватися для симетричного шифрування каналу.

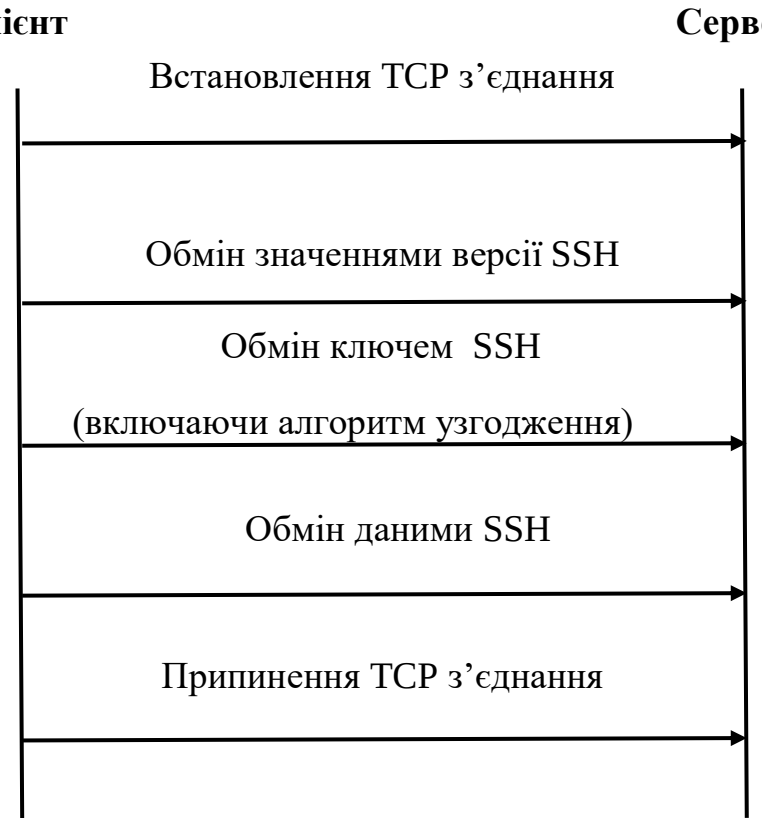
в) Автентифікація клієнта.

Коли клієнт і сервер встановили канал для зашифрованою передачі даних, вони можуть провести автентифікацію по паролю або ключів.

г) Рівень підключення.

Після проведення всіх перерахованих вище процедур, користувач отримує можливість передавати команди сервера або копіювати файли. На цьому рівні забезпечується: мультиплексування каналів (можливість роботи безлічі каналів до одного сервера за рахунок об'єднання їх в один канал), тунелювання, тощо.

На рисунку 5.21 показаний алгоритм встановлення з’єднання між




клієнтом та сервером по протоколу SSH.

Рисунок 5.21 – Алгоритм встановлення з’єднання по протоколу SSH

## 5.8 Pretty Good Privacy - PGP

**PGP** (англ. Pretty Good Privacy) — комп'ютерна програма, також бібліотека функцій, що дозволяє виконувати операції шифрування і цифрового підпису повідомлень, файлів та іншої інформації, поданої в електронному вигляді, у тому числі прозоре шифрування даних на запам'ятовуючих пристроях, наприклад, на твердому диску. Програма була написана Пилипом Цимерманом в 1991 році.

PGP має безліч реалізацій, сумісних між собою і з багатьма іншими програмами (GnuPG, FileCrypt та інші) завдяки стандарту OpenPGP (RFC 4880), але вони мають різний набір функціональних можливостей. Існують реалізації PGP для всіх найбільш поширених операційних систем. Крім вільно розповсюджуваних реалізацій є ще й комерційні.

У 1996 році криптограф Брюс Шнайер охарактеризував ранню версію PGP як «найближчу до криптосистем військового рівня». На даний момент не відомо жодного способу злому даних, зашифрованих PGP, за допомогою повного

перебору або уразливості криптоалгоритма. Ранні версії PGP були теоретично уразливими, тому рекомендується користуватися сучасними версіями.

Криптографічна стійкість PGP заснована на припущенні, що використовувані алгоритми стійкі до криптоаналізу на сучасному обладнанні. Наприклад, у PGP перших версій для шифрування ключів сесії використовувався алгоритм RSA, заснований на односторонній функції факторизації. У PGP версії 2 додатково можна використовувати алгоритм IDEA. У подальшому були додані інші алгоритми шифрування. У жодного використовуваного алгоритму немає відомих вразливостей.

**Принцип роботи PGP**

PGP поєднує в собі найкращі сторони симетричної криптографії і криптографії з відкритим ключем. PGP — це гібридна криптосистема.

Коли користувач зашифровує дані за допомогою PGP, програма для початку їх стискає. Стиск скорочує час модемної передачі і заощаджує дисковий простір, а також, що важливіше, підвищує криптографічну стійкість. Більшість криптоаналітичних техніків засновано на статистичному аналізі шифротексту в пошуках ознак відкритого тексту. Стиск зменшує число таких ознак, що істотно підсилює стійкість до криптоаналізу.

Потім, PGP створює сеансовий ключ, тобто одноразовий симетричний ключ, застосовуваний тільки для однієї операції. Цей сеансовий ключ являє собою псевдовипадкове число, згенероване від випадкових рухів мишки і натискання клавіш. Сеансовий ключ працює на основі дуже надійного, швидкого симетричного алгоритму, яким PGP зашифровує стиснуте повідомлення; у результаті виходить шифротекст. Як тільки дані зашифровані, сеансовий ключ також шифрується, але уже відкритим ключем одержувача. Цей зашифрований відкритим ключем сеансовий ключ прикріплюється до шифротексту і передається разом з ним одержувачеві.

Розшифрування відбувається в зворотному порядку. PGP одержувача використовує його закритий ключ для витягу сеансового ключа з повідомлення, яким шифротекст вихідного послання відновлюється у відкритий текст.(як показано на рисунку 5.22).

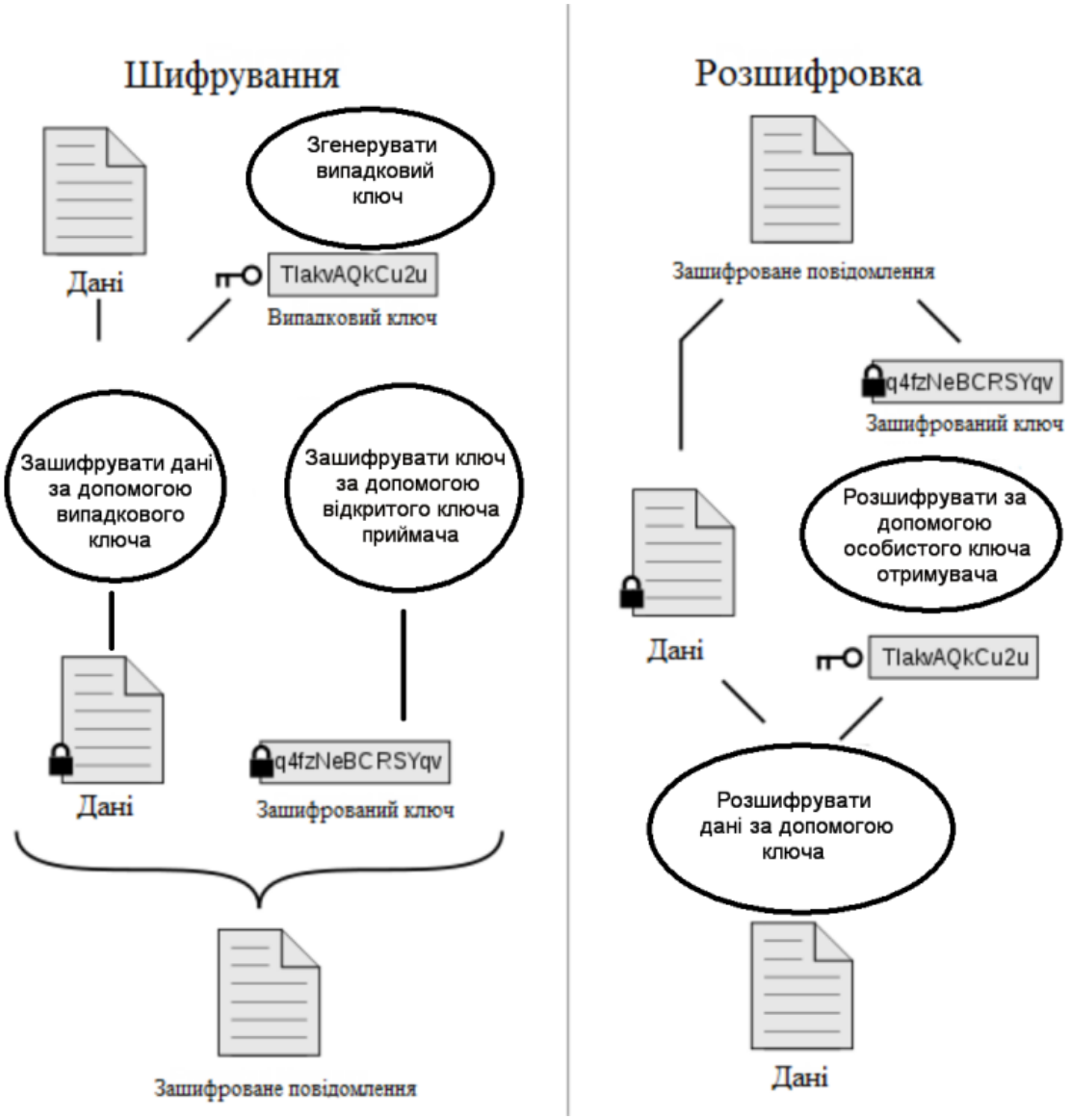



Рисунок 5.22 – Шифрування PGP

Таким чином, комбінація цих двох криптографічних методів поєднує зручність шифрування відкритим ключем зі швидкістю роботи симетричного алгоритму. Симетричне шифрування в тисячі разів швидше асиметричного. Шифрування відкритим ключем, у свою чергу, надає просте рішення проблеми керування ключами і передачі даних. При використовувані їх спільно, швидкість виконання і керування ключами взаємно доповнюються і поліпшуються без якого-небудь збитку безпеки.

**Ступені довіри в PGP**

Найвищий рівень довіри — безумовна довіра (Implicit Trust) — це довіра вашій власній ключовій парі. PGP думає, що якщо ви володієте закритим ключем, то повинні довіряти і діям відповідного відкритого. Існує три ступені довіри, що ви можете привласнити чужому відкритому ключеві:

- повна довіра;

- часткова довіра;
- немає довіри.

Існує також **три рівні дійсності**:

- справжній;
- можливо справжній;
- невизначений.

Щоб дати іншому ключеві повноваження поручительства, ви:

1. Берете справжній ключ, який або підписаний вами, або іншим довіреним поручителем.
2. Встановлюєте рівень довіри, який, на вашу думку, заслуговує власник.

Для прикладу представимо, що на вашому зв'язуванні є ключ Олесі. Ви визначили дійсність її ключа і, підписуючи його, указуєте на це. Вам відомо, що Олеся — активний прихильник ретельної перевірки чужих ключів. Тому ви наділяєте її Повною довірою, що, фактично, перетворює її в Центр сертифікації: якщо Олеся підпише чужий ключ, він буде вірним на вашому зв'язуванні апріорі.

PGP вимагає одну довіру Цілком або дві довірі частково, щоб встановити ключ як справжній. Метод PGP прирівнювання двох Часткових до одній Повного аналогічний тому, як іноді від вас вимагають два види документів, що засвідчують особистість. Ви можете порахувати Олесю частково надійної, також порахувати Олександра, який частково заслуговує довіри. Є ризик, що кожний з них окремо може випадково підписати липовий ключ, так що ви, імовірно, не станете надавати Повної довіри жодному. Однак, імовірність того, що обоє вони підпишуть той самий липовий ключ, досить мала.

**Ключі та зв'язки ключів**

У PGP використовується 4 типи ключів:

- одноразові сеансові ключі;
- відкриті ключі;
- закриті ключі;
- парольні ключі схеми симетричного шифрування.

Вимоги до ключів

- Наявність засобів генерації непередбачуваних сеансових ключів.
- Бажано наявність у користувачів декількох пар відкритих / закритих ключів. Це використовується для того, щоб при взаємодії з різними користувачами використовувати різні ключі. З'являється необхідність використання засобів, що дозволяють ідентифікувати конкретні ключі.
- Об'єкти PGP повинні підтримувати файли власних пар ключів і відкритих ключів кореспондентів

Ідентифікатори ключів

При використанні користувачем однієї пари відкритий/закритий ключ, одержувач відразу б знав яким ключем розшифровувати отриманий сеансовий ключ. Однак кожен користувач має невідоме кількість ключів. Для визначення

користувачем відкритого ключа, яким був зашифрований сеансовий ключ, в PGP кожному відкритому ключу привласнюється такий ідентифікатор, який виявляється з великою ймовірністю унікальним для даного користувача. Ідентифікатор, що пов'язується з відкритим ключем, розміщується в молодших 64 розрядах ключа. Отже, ідентифікатор KUa = KUa mod 264

Ідентифікатори використовуються і в ЦП. Для ідентифікації того, який відкритий ключ необхідно використовувати одержувачу, в розділ ЦП включається 64-бітовий ідентифікатор. При отриманні повідомлення, одержувач перевіряє відповідність ідентифікатора відкритого ключа та перевіряє підпис.

Зв'язки ключів

Ідентифікатори ключів в PGP дуже важливі. Вони включаються в будь-яке повідомлення, де потрібно забезпечити конфіденційність і аутентифікацію. Ключі необхідно зберігати деяким стандартизованим чином. Структура, яка використовується в PGP, передбачає створення в кожному вузлі пари структури даних: одну для зберігання пар відкритий / секретний ключ, іншу для зберігання відкритих ключів інших користувачів. Відповідно називаються вони зв'язкою особистих ключів і зв'язкою відкритих ключів.

Управління відкритими ключами

Для взаємодії користувача А з іншими користувачами, йому потрібно мати відкриті ключі цих користувачів. Для мінімізації ризику того, що зв'язка відкритих ключів містить помилкові ключі, запропоновані наступні варіанти дій:

- Отримання ключа від В фізично.
- Перевірка ключа по телефону.
- Отримання ключа від довіреної посередника D. Для цього D створює підписаний сертифікат, який містить відкритий ключ В, час створення ключа і термін дії. D генерує профіль SHA-1 цього сертифіката, шифрує за допомогою закритого ключа і приєднує до сертифікату. Отриманий сертифікат може бути доставлений А.
- Отримання відкритого ключа У від надійного засвідчує центру. Сертифікат створюється і підписується уповноваженим вузлом. А може отримати до нього доступ, використовуючи логін і пароль.

**СПИСОК ДЖЕРЕЛ**


1. Бирюков А.А. информационная безопасность: защита и нападение. – 2-е изд., перераб. и доп. – М.: ДМК Пресс, 2017, - 434 с.
2. Шаньгин В.Ф. Защита информации в клмпьютерных системах и сетях. – М.: ДМК Пресс, 2012, - 592 с.
3. Vacca, John R. Network and system security. – Syngress, - 432 p.
4. Шелухин О.И., Сакалема Д.Ж., Филинова А.С. Обнаружение вторжений в компьютерные сети (сетевые аномалии). – М.: Горячая линия—Телеком, 2013. — 220 с.
5. Безопасность современных информационных технологий : монография / Е. В. Стельмашонок [и др.]; под общ. ред. Е. В. Стельмашонок. – СПб. : СПбГИЭУ, 2012. – 408 с.
6. Остапов С. Е. Технології захисту інформації : навчальний посібник / С. Е. Остапов, С. П. Євсеєв, О. Г. Король. – Х. : Вид. ХНЕУ, 2013. – 476 с.
7. Гончарова Л.Л., Возненко А.Д., Стасюк О.І., Коваль Ю.О. Основи захисту інформації в телекомунікаційних та комп'ютерних мережах. – К., 2013. – 435 с.
8. Оглтри Т. Firewalls. Практическое применение межсетевых экранов: Пер. с англ. - М.: ДМК Пресс, 2001, - 400 с.
9. Грайворонський М.В., Новіков О.М. Безпека інформаційно-комунікаційних систем. – К.: Видавнича група BHV, 2009. – 608с.
10. Мишин, Д.В. Анализ защищенности распределенных информационных систем. Идентификация ресурсов корпоративной сети передачи данных : Практикум / Д.В. Мишин, Ю.М. Монахов ; Владим. Гос. Ун-т. – Владимир : Изд-во Владим. Гос. Ун-та, 2012. – 97с.
11. Макаренко С. И. Информационная безопасность: учебное пособие для студентов вузов. – Ставрополь: СФ МГГУ им. М. А. Шолохова, 2009. – 372 с.
12. Лапонина О.Р. Основы сетевой безопасности. Часть 1. Межсетевые экраны: Учебное пособие / О.Р. Лапонина — М.: Национальный Открытый Университет «ИНТУИТ» , 2014. — 378 с.
13. Запечников С.В., Милославская Н.Г., Толстой А.И. Основы построения виртуальных частных сетей: Учеб. пособие для вузов. М.: Горячая линия – Телеком, 2003. – 249 с.
14. Браун Стивен. Виртуальные частные сети. – Лори, 2001. – 502 с.
15. Jesse Russell. Deep packet inspection. – Книга по Требованию, 2013 – 136 с.
16. Duane Wessels. Squid: The Definitive Guide. – O'Reilly Media, 2004 – 472 с.
17. Гуз А.Р. Справочное руководство Nmap. – SPecialiST RePack, 2017 – 47 с.
18. Далле Вакке А. Zabbix. Практическое руководство / пер. с англ. А. Н. Киселева. – М.: ДМК Пресс, 2017. – 356 с.
19. Крис Сандерс. Анализ пакетов. Практическое руководство по использованию Wireshark и tcpdump для решения реальных проблем в локальных сетях. – Диалектика, 2017 – 448 с.

20. Кейт Е. Страссберг, Ричард Г. Гондек, Гари Ролли. Полный справочник по брандмауэрам. – Вильямс, 2004 – 848 с.

21. Котенко И.В., Саенко И.Б., Полубелова О.В., Чечулин А.А. Технологии управления информацией и событиями безопасности для защиты компьютерных сетей / Проблемы информационной безопасности. Компьютерные системы, 2012. №2. С. 57–68.

22. Graham Bartlett, Amjad Inamdar. IKEv2 IPsec Virtual Private Networks: Understanding and Deploying IKEv2, IPsec VPNs, and FlexVPN in Cisco IOS. – Cisco Press, 2016 – 608 с.

Навчальне видання

**ЖИЛІН** Артем Вікторович
**ШАПОВАЛ** Олександр Миколайович
**УСПЕНСЬКИЙ** Олександр Анатолійович

# ТЕХНОЛОГІЇ ЗАХИСТУ ІНФОРМАЦІЇ В ІНФОРМАЦІЙНО-ТЕЛЕКОМУНІКАЦІЙНИХ СИСТЕМАХ

**Навчальний посібник**

*Інститут спеціального зв'язку та захисту інформації*
*Національного технічного університету України*
*«Київський політехнічний інститут імені Ігоря Сікорського»*
*вул. Верхньоключова, 4, м. Київ, Україна*
*Тел. 454-91-51*

*В авторській редакції*
*Надруковано з оригінал-макета замовника*

Надруковано

Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського»
Свідоцтво про державну реєстрацію: серія ДК № 5354 від 25.05.2017 р.
просп. Перемоги, 37,
м. Київ, 03056

Підп. до друку 27.05.2021. Формат 60×84 1/16. Папір офс. Гарнітура Times.
Спосіб друку – електрографічний. Ум. друк. арк. 12,56. Обл.-вид. арк. 19,33.
Наклад 50 пр. Поз. 21-1-2-003. Зам. № 21-065.

Видавництво «Політехніка» КПІ ім. Ігоря Сікорського
вул. Політехнічна, 14, корп. 15
03056, м. Київ
тел. (044) 204-81-78