Міністерство освіти і науки України
Кам’янець-Подільський національний університет імені Івана Огієнка
Фізико-математичний факультет
Кафедра комп’ютерних наук

Дипломна робота
магістра

з теми: **«РОЗРОБКА МЕТОДУ ЗАБЕЗПЕЧЕННЯ КОНФІДЕНЦІЙНОСТІ ІНФОРМАЦІЇ В АСУ КРИТИЧНОЇ ІНФРАСТРУКТУРИ»**

Виконав: студент групи KN1-M22
спеціальності 122 Комп’ютерні науки
Богуш Дмитро Васильович

Керівник:
Моцик Р. В., доцент кафедри
комп’ютерних наук

Рецензент:
Оптасюк С. В., кандидат фізико-математичних наук, доцент,
доцент кафедри фізики

Кам’янець-Подільський – 2023

ЗМІСТ

ВСТУП

У сучасному світі інтенсивний розвиток технологій супроводжується розширенням та ускладненням критичних інфраструктур, таких як електроенергетика, транспорт, водопостачання та інші, що стає об'єктом зростаючої уваги у галузі кібербезпеки. Автоматизовані системи управління виявляються серцевиною цих інфраструктур та дозволяють ефективно контролювати та управляти великими мережами.

Кіберзагрози та атаки на інформаційні системи критичних інфраструктур стають все більш складними та вдосконаленими, загрожуючи не лише ефективності роботи, але й безпеці та стійкості цих систем. Тому проблема забезпечення кібербезпеки автоматизованих систем управління критичною інфраструктурою стає вельми актуальною.

Актуальність. Актуальність захисту інформації в автоматизованих системах управління критичної інфраструктури визначається різноманітними факторами, які включають в себе технологічний прогрес, зростання кількості кіберзагроз та залежність від інформаційних технологій.

У сучасному світі, де системи управління виробництвом, енергетичні мережі, транспортні системи та інші аспекти критичної інфраструктури стають все більш автоматизованими та підключеними до мережі, захист інформації стає вирішальною складовою стійкості та безпеки цих систем.

Зростання кількості кіберзагроз, таких як атаки вірусами, фішинг, атаки на вторгнення та інші, викликає необхідність постійного удосконалення заходів кібербезпеки. Враховуючи потенційні наслідки кібератак на критичну інфраструктуру, включаючи можливість втрати контролю над системами та порушення нормального функціонування, захист інформації стає завданням вищого пріоритету.

Крім того, розвиток мережі Інтернет та збільшення кількості підключених пристроїв у критичних системах підвищує ризики з точки зору безпеки. Тому актуальність захисту інформації в АСУ критичної інфраструктури не тільки

залишається важливою, але й постійно зростає в умовах швидкого технологічного розвитку та зміни кіберзагроз..

Тема роботи. Розробка методу забезпечення конфіденційності інформації в АСУ критичної інфраструктури.

Мета роботи. Для досягнення успішного результату із виконання магістерської роботи потрібно:

- проаналізувати сучасні рішення щодо забезпечення конфіденційності АСУ КІ;

- вивчення вразливостей, пов’язаних з людським фактором та технічними аспектами;

- вдосконалення функціоналу систем управління для забезпечення ефективного використання ресурсів;

- розробка та впровадження заходів забезпечення конфіденційності інформації в АСУ.

Предметом дослідження захисту інформації є процес розробки інформаційної складової згідно сучасних тенденцій кібербезпеки.

Об’єктом дослідження методу є системи захисту, передачі та обробки інформації.

У процесі дослідження будуть розглянуті такі ключові аспекти, як розробка сучасних методів виявлення та ідентифікації атак, аналіз вразливостей систем, вдосконалення процесів реагування на інциденти та підвищення освідомленності персоналу.

Практичним значенням даної магістерської кваліфікаційної роботи є надійний захист інформації в автоматизованих системах управління критичної інфраструктури.

Задачами проєкту є аналіз тенденцій на ринку програмних застосунків, які забезпечують захист даних, їх збереження, передачу та обробку, необхідного програмного забезпечення, реалізація власного або покращеного уже існуючого методу для надійного захисту інформації.

Структура роботи. Магістерська робота складається із вступу, двох розділів (1 теоретичного і 1 практичного), висновків, 14 рисунків, списку використаних джерел, обсяг роботи становить 53 сторінок.

1. АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ І ІСНУЮЧИХ ПІДХОДІВ ЗАХИСТУ ІНФОРМАЦІЇ В АСУ

Автоматизована система управління (АСУ) є комплексом технічних та програмних рішень, спрямованим на автоматизацію та оптимізацію управлінських процесів в різних сферах. Ця система об’єднує апаратне забезпечення, яке здійснює вимірювання та збір даних, програмне забезпечення, яке обробляє та аналізує ці дані, а також механізми впливу на контрольовані об’єкти.

Основна ідея АСУ полягає в тому, щоб забезпечити автоматизоване управління та контроль за процесами, що відбуваються в системі. Система отримує дані від сенсорів і вимірювальних приладів, обробляє ці дані за допомогою контролерів і, в результаті, генерує команди для виконавчих механізмів. Цей замкнений цикл дозволяє системі динамічно реагувати на зміни в навколишньому середовищі або в самому об’єкті управління.

Використовуючи АСУ, організації можуть досягти покращення в ефективності, якості та безпеці своєї діяльності.

Ключові складові АСУ включають сенсори та вимірювальні пристрої для збору даних, контролери для обробки цих даних та визначення оптимальних стратегій управління, а також виконавчі механізми для впливу на систему відповідно до згенерованих команд.

АСУ визначають сучасний підхід до управління технічними та виробничими процесами, привносячи ефективність та автоматизацію в різні сфери людської діяльності.

АСУ критичної інфраструктури (АСУ КІ) - це система автоматизованого управління, яка використовується для контролю та управління критичною інфраструктурою, яка охоплює об’єкти та системи, які мають важливе значення для безпеки, економіки та суспільного функціонування.

Це може включати в себе такі сектори, як енергетика, транспорт, водопостачання та водовідведення, телекомунікації, фінансова система, охорона

здоров’я, харчова промисловість тощо. АСУ КІ використовується для забезпечення ефективного та безпечного управління цими системами.

Оскільки критична інфраструктура може бути об’єктом потенційних кібератак, безпека та надійність системи АСУ КІ є надзвичайно важливою. Вона повинна мати захист від кіберзагроз, включаючи віруси, хакерські атаки та інші форми кіберзлочинності. Також, АСУ КІ може включати системи моніторингу та реагування на екстрені ситуації для негайного реагування на можливі загрози та відновлення нормального функціонування.

Предметна область АСУ критичної інфраструктури (АСУ КІ) охоплює широкий спектр секторів та об’єктів, які є важливими для нормального функціонування суспільства, економіки та національної безпеки. Серед них одні з ключових секторів та об’єктів, які входять у предметну область:

- Енергетика. Включає електростанції, підстанції, електромережі та інші об’єкти, що забезпечують електропостачання.

- Транспортні системи. Автомобільний, залізничний, авіаційний, водний та інші види транспорту.

- Телекомунікаційні системи. Забезпечують зв’язок та обмін інформацією між різними об’єктами та системами.

- Водопостачання та водовідведення. Об’єкти, пов’язані із забезпеченням водою та водовідведенням.

- Фінансові системи. Банківські установи та інші фінансові об’єкти, які грають важливу роль у функціонуванні економіки.

- Охорона здоров’я. Лікарні, медичні центри, аптечні мережі та інші об’єкти здоров’я.

- Харчова промисловість. Об’єкти, пов’язані із виробництвом, переробкою та постачанням харчових продуктів.

- Хімічна та нафтова промисловість. Об’єкти, що виробляють хімічні речовини та нафтопродукти.

- Галузі національної безпеки. Включають військові об’єкти, об’єкти забезпечення безпеки та об’єкти, які забезпечують національну оборону.

Предметна область АСУ КІ може включати і інші галузі в залежності від конкретної країни та її потреб. Оскільки ці об’єкти є критичними для суспільства, безпека їхньої експлуатації та функціонування стає пріоритетом.

АСУ критичної інфраструктури піддаються різноманітним загрозам, які можуть бути кібернетичного, фізичного чи соціального характеру:

- Кібератаки: віруси, черв’яки та троянські програми - зловмисний код, який може внедрюватися у системи і руйнувати їхню роботу. Деніал-сервіс атаки (DDoS) - спроби перевантажити мережу або систему, забороняючи легітимним користувачам отримувати доступ.

- Фізичні загрози: фізичні атаки на обладнання та інфраструктуру, пожежі, повені та інші стихійні лиха, природні катастрофи, які можуть призвести до пошкодження обладнання та перерв у роботі систем.

- Соціальні інженерні атаки такі як фішинг та обман користувачів, шахрайські методи для отримання конфіденційної інформації шляхом маніпулювання користувачами, інсайдерські загрози - зловмисники, які мають доступ до системи через внутрішніх працівників.

- Вразливості в програмному забезпеченні, недоліки безпеки в програмах, які можуть бути використані для несанкціонованого доступу, недостатні заходи захисту, їх відсутність або недостатність, таких як слабкі паролі, недостатні обмеження доступу.

- Втручання в сигнали та комунікації, перехоплення та зміна даних в мережі, спроби перехопити або змінити передані дані, шпигунство та прослуховування, незаконне збирання інформації.

- Недостатність резервного копіювання та відновлення, втрата даних, неможливість відновлення даних у випадку пошкодження.

Для боротьби з цими загрозами, системи АСУ КІ повинні використовувати комплексний підхід до кібербезпеки.

Захист автоматизованих систем управління критичної інфраструктури є критично важливим завданням в умовах зростаючих кіберзагроз. Ефективність заходів захисту визначається комплексним підходом, що охоплює різноманітні

аспекти безпеки. Це включає ідентифікацію та аутентифікацію, контроль доступу, шифрування даних, системи моніторингу та виявлення вторгнень, фізичну безпеку, оновлення та патчі, захист від внутрішнього загрози, аварійне відновлення, відповідність стандартам та законодавству, а також постійне вдосконалення заходів безпеки.

Цей комплексний підхід спрямований на забезпечення конфіденційності, цілісності та доступності інформації, а також запобігання порушенням, втратам та вириванню систем з ладу. Кожен з цих елементів має свою важливість у створенні надійної системи захисту, спроможної відповісти на сучасні кіберзагрози та зберігати стабільність та безпеку критичних об’єктів інфраструктури.

1.1 Методи забезпечення захисту інформації в АСУ

На сьогодні склалося два підходи до проблеми забезпечення безпеки АСУ, назвемо їх умовно фрагментарний та комплексний.

Фрагментарний підхід орієнтується на протидію точно визначеним загрозам за певних умов.

Прикладами реалізації такого підходу є, наприклад, спеціалізовані антивірусні засоби, окремі засоби реєстрації і керування, автономні засоби шифрування і т.д. Головна відмінність фрагментарного підходу - відсутність єдиного захищеного середовища обробки інформації.

Головною перевагою фрагментарного підходу є його висока вибірковість щодо конкретної загрози, обумовлююча і його основний недолік локальність дії.

Фрагментарні заходи захисту забезпечують ефективний захист конкретних об’єктів АСУ від конкретної загрози, але не більше. Невелика видозміна загрози веде до втрати ефективності захисту; розпитування гранити дію таких заходів на всю АСУ практично неможливо. Особливістю комплексного підходу є створення захищеного середовища обробки інформації в АСУ, що поєднує різноманітні заходи протидії загрозам (правові, організаційні, програмно-технічні). Захищене

середовище обробки інформації будується на основі розроблених для конкретної АСУ правил обробки критичної інформації.

Організація захищеного середовища обробки інформації дозволяє гарантувати (у рамках розробленої політики безпеки) рівень безпеки АСУ. Недоліками підходу є висока чутливість до помилок встановлення та налаштування засобів захисту, складність управління, обмеження на свободу дій користувачів АСУ.

Комплексний підхід застосовують для захисту великих АСУ або невеликих АСУ, які обробляють дорогу інформацію або виконують відповідальні завдання. При цьому спосіб реалізації комплексного захисту визначається специфікою АСУ, іншими об'єктивними та суб'єктивними факторами.

Для всіх великих організацій характерно те, що порушення безпеки інформації в їхній АСУ може завдати величезних матеріальних збитків як самим організаціям, так і їхнім клієнтам. Тому ці організації змушені приділяти особливу увагу гарантіям безпеки, що веде до необхідності реалізації комплексного захисту.

Комплексного підходу дотримуються більшість Державних та великих комерційних підприємств та установ, він знайшов своє відображення у стандартах і цілеспрямовано проводиться в життя, наприклад, Міністерством оборони США в особі Національного Центру Комп'ютерної Безпеки (NCSC).

Під системою захисту АСУ будемо надалі розуміти єдину сукупність правових та морально-етичних норм, організаційних (адміністративних) заходів та програмно-технічних засобів, спрямованих на протидію загрозам АСУ з метою зведення до мінімуму можливої шкоди користувачам та власникам системи.

Етап аналізу можливих загроз АСУ необхідний для фіксування на певний момент часу стану АСУ (конфігурації апаратних та програмних засобів, технології обробки інформації) та визначення можливих впливів на кожен компонент системи.

Забезпечити захист АСУ від всіляких впливів на неї неможливо хоча б тому, що неможливо повністю встановити перелік загроз та способів їх реалізації.

Тому треба вибрати з безлічі можливих впливів лише ті, які можуть реально відбутися і завдати серйозної шкоди власникам і користувачам системи.

На етапі планування формується система захисту як єдина сукупність заходів протидії різної природи.

За способами здійснення всі заходи забезпечення безпеки комп'ютерних систем поділяються на правові (законодавчі), морально-етичні, адміністративні, фізичні та технічні (апаратні та програмні). До правових заходів захисту відносяться чинні країни закони, укази та інші нормативні акти, що регламентують правила поводження з інформацією обмеженого використання та відповідальність за їх порушення. Цим вони перешкоджають несанкціонованому використанню інформації та є стримуючим фактором для потенційних порушників.

До морально-етичних заходів протидії відносяться всілякі норми поведінки, які традиційно склалися або складаються в міру поширення ЕОМ у країні чи суспільстві. Ці норми здебільшого є обов'язковими, як законодавчо затверджені, проте їх недотримання веде зазвичай до падіння авторитету, престижу людини, групи осіб чи організації. Морально-етичні норми бувають як неписані (наприклад, загальновизнані норми чесності, патріотизму тощо), і оформлені у певний звід (статут) правил чи розпоряджень. Найбільш характерним прикладом останніх є Кодекс професійної поведінки членів Асоціації користувачів ЕОМ США. Зокрема, вважаються неетичними навмисні чи ненавмисні дії, які:

- порушують нормальну роботу комп'ютерних систем;

- Викликають додаткові невиправдані витрати ресурсів (машинного часу, пам'яті, каналів зв'язку тощо);

- порушують цілісність збереженої та оброблюваної інформації;

- порушують інтереси інших законних користувачів і т.д.

Адміністративні заходи захисту - це заходи організаційного характеру, які регламентують процеси функціонування системи обробки інформації, використання її ресурсів, діяльність персоналу, і навіть порядок взаємодії

користувачів із системою в такий спосіб, щоб максимально утруднити чи виключити можливість реалізації загроз безпеці. Вони включають:

- розробку правил обробки інформації в АСУ;

- заходи, що здійснюються при проектуванні, будівництві та обладнанні обчислювальних центрів та інших об'єктів АСУ (облік впливу стихії, пожеж, охорона приміщень, організація захисту від встановлення апаратури, що прослуховує тощо);

- заходи, що здійснюються при доборі та підготовці персоналу (перевірка нових співробітників, ознайомлення їх з порядком роботи з конфіденційною інформацією, з заходами відповідальності за порушення правил се обробки; створення умов, за яких персоналу було б невигідно допускати зловживання тощо);

- організацію надійного пропускного режиму;

- організацію обліку, зберігання, використання та знищення документів та носіїв з конфіденційною інформацією;

- розподіл реквізитів розмежування доступу (паролів, профілів повноважень тощо);

- організацію підготовки та прихованого контролю за роботою користувачів та персоналу АСУ;

- заходи, що здійснюються при проектуванні, розробці, ремонті та модифікаціях обладнання та програмного забезпечення (сертифікація використовуваних технічних та програмних засобів, суворе санкціонування, розгляд та затвердження всіх змін, перевірка їх на задоволення вимог захисту, документальне відображення змін тощо).

Фізичні заходи захисту - це різного роду механічні, електро- або електронно-механічні пристрої та споруди, спеціально призначені для створення фізичних перешкод на можливих шляхах проникнення та доступу потенційних порушників до компонентів системи та інформації, що захищається.

Технічними (апаратно-програмними) засобами захисту називаються різні електронні пристрої та спеціальні програми, які виконують (самостійно або в

комплексі з іншими засобами) функції захисту (ідентифікацію та автентифікацію користувачів, розмежування доступу до ресурсів, реєстрацію подій, криптографічний захист інформації тощо).

Найкращі результати досягаються при системному підході до питань забезпечення безпеки АСУ та комплексному використанні різних заходів захисту на всіх етапах життєвого циклу системи, починаючи з ранніх стадій її проектування.

Очевидно, що у структурах із низьким рівнем правопорядку, дисципліни та етики ставити питання про захист інформації просто безглуздо. Насамперед треба вирішити правові та організаційні питання.

Адміністративні заходи відіграють значну роль у забезпеченні безпеки АСУ. Ці заходи необхідно використовувати тоді, коли інші методи та засоби захисту просто недоступні (відсутні або занадто дорогі). Однак це зовсім не означає, що систему захисту необхідно будувати виключно на основі адміністративних методів, як це часто намагаються зробити чиновники, далекі від технічного прогресу. Цим заходам притаманні серйозні недоліки, такі як:

- низька їх надійність без відповідної підтримки з боку фізичних, технічних та програмних засобів (люди схильні до порушення будь-яких встановлених правил, якщо їх можна порушити);

- застосування для захисту лише адміністративних заходів зазвичай призводить до паралічу діяльності АСУ та всієї організації (цілком неможливо працювати, не порушуючи інструкцій) через низку додаткових незручностей, пов’язаних з великим обсягом рутинної формальної діяльності.

Адміністративні заходи треба скрізь, де тільки можливо, замінювати надійнішими сучасними фізичними та технічними засобами. Вони повинні забезпечувати ефективне застосування інших, більш надійних методів та засобів захисту у частині, що стосується регламентації дій людей.

Відомо не так багато загальних (універсальних) способів захисту АСУ від різних впливів на неї. Ними є:

- ідентифікація та автентифікація суб’єктів (користувачів, процесів тощо) АСУ;

- контроль доступу до ресурсів АСУ;

- реєстрація та аналіз подій, що відбуваються в АСУ;

- контроль цілісності об’єктів АСУ;

- шифрування даних;

- резервування ресурсів та компонентів АСУ.

Ці універсальні способи захисту можуть застосовуватись у різних варіаціях та сукупностях у конкретних методах та засобах захисту.

Результатом етапу планування є план захисту - документ, що містить перелік компонентів АСУ, що захищаються, і можливих впливів на них, мета захисту інформації в АСУ, правила обробки інформації в АСУ, що забезпечують її захист від різних впливів, а також опис розробленої системи захисту інформації.

При необхідності, крім плану захисту, на етапі планування може бути розроблений план забезпечення безперервної роботи та відновлення функціонування АСУ, що передбачає діяльність персоналу та користувачів системи відновлення процесу обробки інформації у разі різних стихійних лих та інших критичних ситуацій.

Сутність етапу реалізації системи захисту полягає в установці та налаштуванні засобів захисту, необхідних для реалізації зафіксованих у плані захисту правил обробки інформації. Зміст етапу залежить від способу реалізації механізмів захисту засобів захисту.

На цей час сформувалися два основних способи реалізації механізмів захисту.

При першому з них механізми захисту не реалізовані в програмному та апаратному забезпеченні АСУ або реалізована лише частина їх, необхідна для забезпечення працездатності всієї АСУ (наприклад, механізми захисту пам’яті в мультикористувальних системах). Захист інформації при зберіганні, обробці або передачі забезпечується додатковими програмними або апаратними засобами, що

входять до складу самої АСУ. При цьому засоби захисту підтримуються внутрішніми механізмами АСУ.

Такий спосіб отримав назву «доданої» (add-on) захисту, оскільки засоби захисту є доповненням до основних програмних та апаратних засобів АСУ. Такий підхід у забезпеченні безпеки дотримується, наприклад, фірма IBM майже всі моделі се комп'ютерів і ОС, від персональних до великих машин (крім AS/400), використовують доданий захист (наприклад, пакет RACF).

Інший спосіб носить назву «вбудованого» (built-in) захисту. Він полягає в тому, що механізми захисту є невід'ємною частиною АСУ, розробленою та реалізованою з урахуванням певних вимог безпеки. Механізми захисту можуть бути реалізовані у вигляді окремих компонентів АСУ, розподілені за іншими компонентами системи (тобто, у деякому компоненті АСУ є частина, яка відповідає за підтримку його захисту). При цьому засоби захисту складають єдиний механізм, який відповідає за безпеку всієї АСУ. Цей спосіб використовувався компанією DEC для розробки системи VAX/VMS.

Обидва способи - доданого та вбудованого захисту - мають свої переваги та недоліки. Доданий захист є більш гнучким, його механізми можна додавати або видаляти за необхідності. Це не складе великої праці, тому що всі вони реалізовані окремо з інших процедур системи. У тому випадку, якщо засоби захисту, що додаються, не підтримуються вбудованими механізмами АСУ, то вони не забезпечать необхідного рівня безпеки.

Проблемою може стати поєднання вбудованих механізмів з програмними засобами, що додаються - досить складно розробити конфігурацію механізмів захисту, їх інтерфейс з програмними засобами, що додаються, щоб захист охоплював всю систему повністю.

Інша проблема – оптимальність захисту. За будь-якої перевірки прав, призначення повноважень, дозволів доступу необхідно викликати процедуру, що позначається на продуктивності системи. Важлива проблема сумісності захисту з наявними програмними засобами. Мрі доданої захисту вносяться зміни до логіки

роботи системи. Зміни можуть бути неприйнятними для прикладних програм. Така плата за гнучкість та полегшення обслуговування засобів захисту.

Основна перевага вбудованого захисту – надійність та оптимальність. Засоби захисту та механізми їх підтримки розроблялися та реалізовувалися одночасно із самою системою обробки інформації, тому взаємозв’язок засобів захисту з різними компонентами системи вже, ніж при доданому захисті. Однак вбудований захист має жорстко фіксований набір функцій, не дозволяючи розширювати або скорочувати їх. Деякі функції можна лише вимкнути. Обидва види захисту у чистому вигляді зустрічаються рідко. Як правило, використовуються їх комбінації, що дозволяє поєднувати переваги та компенсувати недоліки кожного з них.

Комплексний захист АСУ може бути реалізований як за допомогою доданого, так і вбудованого захисту.

Етап супроводу полягає у контролі роботи системи, реєстрації подій, що відбуваються в ній, їх аналізі з метою виявити порушення безпеки.

У тому випадку, коли склад системи зазнав суттєвих змін (зміна обчислювальної техніки, переїзд в іншу будівлю, додавання нових пристроїв або програмних засобів), потрібне повторення описаної вище послідовності дій. Варто відзначити важливий факт, що забезпечення захисту АСУ - це інтерактивний процес, що закінчується лише з завершенням життєвого циклу системи.

1.2. Розбір можливих загроз для АСУ КІ

Автоматизовані системи управління критичної інфраструктури можуть бути піддані різноманітним видам кібератак через їхню зв'язаність з мережами та залежність від інформаційних технологій.

1.2.1. Віруси та черв’яки

Віруси та черв'яки можуть становити серйозну загрозу для автоматизованих систем управління в контексті критичної інфраструктури. Ці шкідливі програми можуть швидко розповсюджуватися через мережі АСУ, використовуючи

потенційні вразливості та слабкі точки безпеки. Їхні впливи на роботу систем управління та контролю можуть бути надзвичайно серйозними, порушуючи ключові процеси критичної інфраструктури. Наприклад, зміни в логіці роботи програм та систем можуть викликати неправильні команди та призвести до виникнення аварійних ситуацій. Безпека внутрішніх мереж також є важливою, оскільки віруси можуть експлуатувати вразливості для незаконного доступу та виявлення внутрішніх вразливостей мережі АСУ. Тому важливо приділяти велику увагу заходам кібербезпеки для захисту від цих загроз в критичних системах управління.

1.2.2. DDoS-атаки (атаки з великою кількістю запитів)

DDoS-атаки можуть представляти значну небезпеку для автоматизованих систем управління в контексті критичної інфраструктури. Ці атаки, спрямовані на перевантаження мережевих ресурсів, можуть призвести до тимчасового або повного відмову в обслуговуванні систем управління та контролю. Внаслідок цього можуть виникнути серйозні наслідки, включаючи зниження ефективності роботи та ризик виникнення аварій.

DDoS-атаки можуть бути спрямовані на різноманітні складові АСУ, включаючи сервери, комунікаційні канали та інші ключові елементи. Їхній вплив може виявити серйозний тиск на мережеві інфраструктури та заважати звичайному функціонуванню систем управління. Захист від DDoS-атак вимагає ефективних технічних та організаційних заходів, таких як використання систем фільтрації трафіку, застосування CDN (мережі доставки контенту) та моніторинг мережі для виявлення та реагування на надмірний трафік. Передбачення та підготовка до можливих DDoS-атак є важливою частиною стратегії кібербезпеки для АСУ в критичній інфраструктурі.

1.2.3. Фішинг та соціальна інженерія

Фішинг та соціальна інженерія представляють суттєву загрозу для автоматизованих систем управління в контексті критичної інфраструктури. Атаки цього типу спрямовані на здобуття конфіденційної інформації чи доступу до

систем шляхом маніпулювання співробітників та отримання їхньої авторизаційної інформації.

Зловмисники можуть використовувати електронні листи, які здаються легітимними, для введення співробітників у оману та отримання їхніх облікових даних або інших конфіденційних відомостей. Сприймаючи листи або повідомлення як внутрішні чи офіційні, співробітники можуть випадково надати зловмисникам доступ до системи.

Соціальна інженерія може включати в себе використання маніпуляційних технік, таких як встановлення довіри чи викликання емоційних реакцій, щоб отримати від співробітників інформацію або навіть отримати фізичний доступ до систем.

Враховуючи ризики фішингу та соціальної інженерії, необхідно вдосконалювати освіту персоналу, навчати їх визначати підозрілі ситуації та вчасно повідомляти про можливі загрози. Також важливо встановлювати механізми перевірки ідентифікації та вдосконалювати процеси аутентифікації для зменшення вразливості АСУ перед атаками цього типу.

1.2.4. Атаки на комунікаційні канали

Атаки на комунікаційні канали в автоматизованих системах управління представляють суттєвий ризик для безпеки критичної інфраструктури. Ці атаки можуть включати в себе перехоплення, модифікацію або блокування інформації, яка передається між різними компонентами системи.

Зловмисники можуть використовувати атаки на комунікаційні канали для впливу на обмін даними між контролерами, програмованими логічними контролерами (ПЛК), сенсорами та іншими пристроями, порушуючи нормальну роботу систем управління та контролю.

Ці атаки можуть бути спрямовані на внедрення шкідливого коду у передачу даних або викликання невірного розпізнавання сигналів між компонентами системи. Наприклад, модифікація команд, які передаються від контролера до ПЛК, може призвести до неправильної реакції об'єкта управління, що може мати серйозні наслідки для процесів критичної інфраструктури.

Захист від атак на комунікаційні канали вимагає ефективних механізмів шифрування та автентифікації для забезпечення конфіденційності та цілісності даних. Регулярні перевірки на виявлення аномального трафіку та вдосконалення заходів моніторингу також є ключовими аспектами забезпечення кібербезпеки комунікаційних каналів в АСУ.

1.2.5. Зламані аутентифікаційні дані

Зламані аутентифікаційні дані у автоматизованих системах управління можуть призвести до серйозних загроз для безпеки критичної інфраструктури. Ці атаки передбачають отримання незаконного доступу до систем шляхом отримання облікових даних або використання методів введення у систему з імперсонацією користувача.

Зловмисники можуть використовувати різні методи для зламування аутентифікаційних даних, такі як перехоплення паролів, використання слабких паролів, атаки на методи аутентифікації, або використання розгалужених методів, таких як фішинг та соціальна інженерія, для отримання конфіденційної інформації від користувачів.

Отримання несанкціонованого доступу до облікових записів управління системами АСУ може дозволити зловмисникам змінювати налаштування систем, передавати невірні команди, або навіть блокувати доступ до системи. У контексті критичної інфраструктури, це може призвести до неправильного функціонування систем управління та серйозних наслідків для безпеки об'єктів управління.

Для захисту від зламання аутентифікаційних даних в АСУ, важливо використовувати сильні паролі, застосовувати методи двофакторної аутентифікації, регулярно оновлювати облікові записи та вдосконалювати механізми виявлення та реагування на незвичайну активність. Також важливо підвищувати осведомленість користувачів щодо потенційних загроз та застосовувати найновіші методи кіберзахисту.

1.2.6. Атаки на Інфраструктуру Інтернету Речей (IoT)

Атаки на інфраструктуру Інтернету Речей (IoT) в автоматизованих системах управління можуть представляти значний ризик для безпеки критичної

інфраструктури. Зловмисники можуть використовувати вразливості в пристроях IoT, які використовуються в АСУ, для незаконного доступу, введення шкідливого коду або виконання атак на мережеві пристрої.

Ці атаки можуть призвести до зміни параметрів роботи пристроїв IoT, перехоплення інформації, що передається через мережу, або навіть блокування роботи окремих пристроїв. З урахуванням того, що пристрої IoT можуть бути пов'язані з системами управління важливою інфраструктурою, такими як енергетичні мережі чи транспортні системи, компрометація цих пристроїв може мати серйозні наслідки.

Для захисту від атак на Інтернет Речей в АСУ, необхідно вживати заходів з кібербезпеки для пристроїв IoT, таких як встановлення оновлень та виправлень вразливостей, застосування сильних методів аутентифікації та шифрування, а також впровадження механізмів моніторингу та виявлення надмірної активності. Освіта та тренінг персоналу щодо безпеки використання IoT-пристроїв також є ключовими елементами стратегії забезпечення кібербезпеки в АСУ.

1.3. Огляд існуючих програмних застосунків для захисту АСУ

Програмні засоби для захисту автоматизованих систем управління є важливою складовою сфери кібербезпеки. Ці засоби призначені для забезпечення безпеки та надійності функціонування систем, упередження та виявлення можливих загроз та атак, а також для захисту конфіденційності, цілісності та доступності інформації.

Програмне забезпечення для захисту АСУ може включати в себе різноманітні рішення, такі як антивіруси, системи виявлення та запобігання вторгненням, системи керування доступом, механізми шифрування, фаєрволи, системи резервного копіювання та відновлення, а також інші засоби, що використовуються для захисту інформаційних ресурсів та інфраструктури.

Ці програмні засоби допомагають створити комплексний підхід до безпеки, охоплюючи як технічні, так і організаційні заходи. Вони дозволяють ефективно

виявляти та запобігати загрозам, вдосконалювати політику доступу, керувати конфігурацією систем, а також реагувати на події безпеки за допомогою моніторингу та аналізу.

Обраний комплекс програмних заходів повинен відповідати конкретним потребам та характеристикам АСУ, забезпечуючи баланс між безпекою, продуктивністю та зручністю в управлінні.

1.3.1. Система виявлення та запобігання вторгнень (IDS/IPS) Snort

Snort - це система виявлення та запобігання вторгнень (IDS/IPS), яка виявляє аномалії в мережевому трафіку та реагує на можливі загрози. Розроблена як відкрите програмне забезпечення, Snort ставить своїм завданням виявлення підозрілих або зловмисних дій, використовуючи набір правил і сигнатур, які описують відомі типи атак.

Рисунок 1.1 – логотип програмного застосунку «Snort»

Його режим виявлення вторгнень (IDS) дозволяє лише аналізувати та реєструвати події без прямого блокування трафіку. У режимі запобігання вторгнень (IPS) Snort може приймати додаткові заходи для блокування атак та захисту мережі.

Snort виявляє атаки, аналізуючи заголовки та дані пакетів, порівнюючи їх із заданими правилами та сигнатурами. Він може використовувати різні методи аналізу, включаючи розпізнавання сигнатур, аналіз протоколів та виявлення аномалій.

Адміністратори можуть налаштовувати Snort, визначаючи, які правила застосовувати, як реагувати на певні події та як збирати дані для аналізу. Також, для оптимізації роботи, важливо регулярно оновлювати базу сигнатур та використовувати актуальні дані для ефективного виявлення нових загроз.

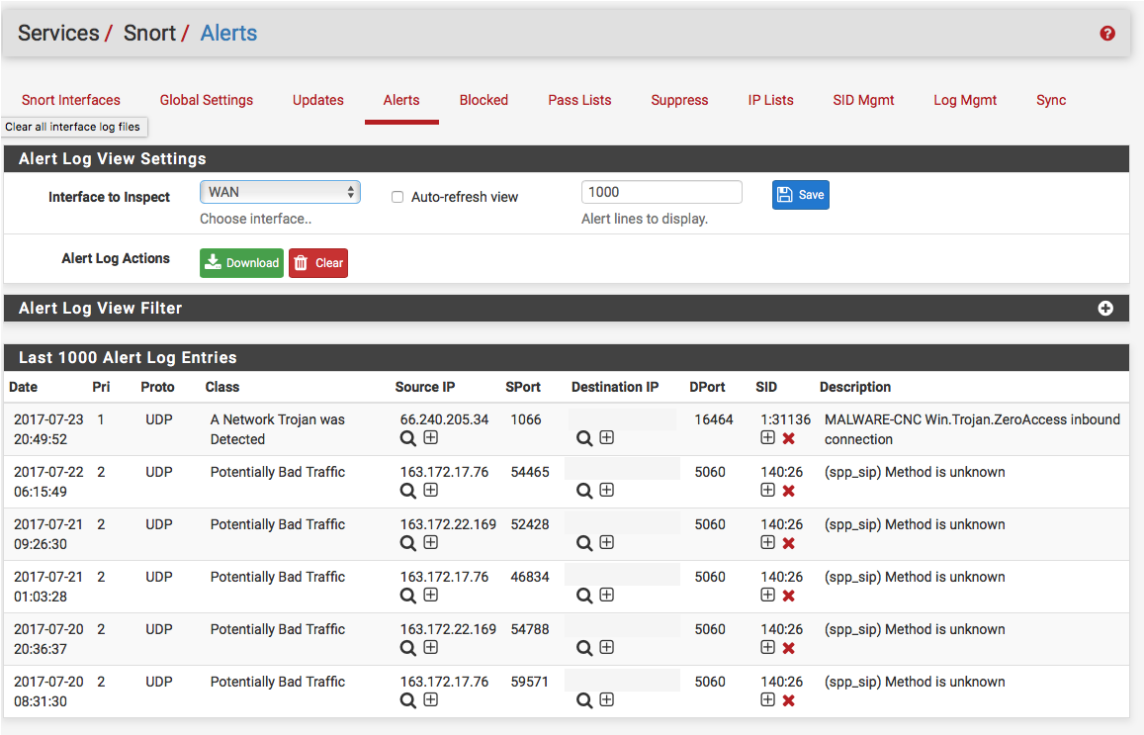

Рисунок 1.2 – приклад роботи застосунку

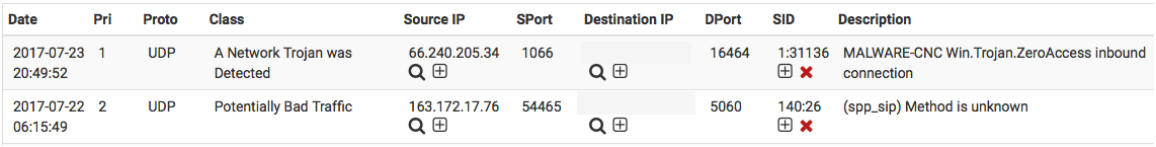

Рисунок 1.3 – деталі сповіщень щодо вторгнень

Snort використовується у багатьох організаціях як частина комплексної стратегії кібербезпеки для захисту мережі та даних від потенційних атак і зловмисних дій.

Переваги Snort:

1. Ефективність виявлення атак: Snort відомий своєю здатністю ефективно виявляти атаки та ненормальну активність в мережі, особливо за допомогою підписів.

2. Підтримка плагінів: Є можливість розширення функціоналу за допомогою плагінів, що дозволяє користувачам налаштовувати систему з урахуванням конкретних вимог.

3. Активна спільнота користувачів: Існує широка спільнота користувачів та розробників, що сприяє обміну досвідом та розробці нових можливостей.

Недоліки Snort:

1. Залежність від підписів: Snort ґрунтується на системі підписів, що може призвести до пропусків у виявленні нових атак, які не мають відповідного підпису.

2. Можливість фальшивих спрацювань: Існує ризик отримання фальшивих позитивів, коли система помічає нормальну активність як потенційну загрозу.

3. Система обробки трафіку: У великих мережах може виникати проблема з обробкою великої кількості трафіку, що може вплинути на продуктивність.

4. Слабка підтримка для деяких протоколів: Деякі протоколи можуть бути менш добре підтримані, що може призвести до обмежень у виявленні атак для конкретних типів трафіку.

1.3.2. Система шифрування Symantec

Symantec Endpoint Encryption - це комплексне рішення для шифрування даних в корпоративних середовищах, розроблене компанією Symantec. Його головною метою є забезпечення безпеки конфіденційної інформації та виконання вимог щодо захисту даних.

Продукт дозволяє організаціям шифрувати дані на різних пристроях, таких як комп’ютери, ноутбуки, портативні носії інформації тощо. Шифрування даних забезпечує захист від несанкціонованого доступу і зберігає конфіденційні інформаційні ресурси в безпеці.

Рисунок 1.3 – логотип Symantec

Symantec Endpoint Encryption також надає централізоване управління політикою шифрування, дозволяючи адміністраторам встановлювати правила, контролювати ключі шифрування та забезпечувати дотримання внутрішніх та регуляторних вимог щодо безпеки.

Продукт включає механізми моніторингу та аудиту, що дозволяють відслідковувати дії користувачів та виявляти можливі аномалії в системі. Він також може підтримувати відновлення даних та втручання в разі проблем, пов’язаних із шифруванням.

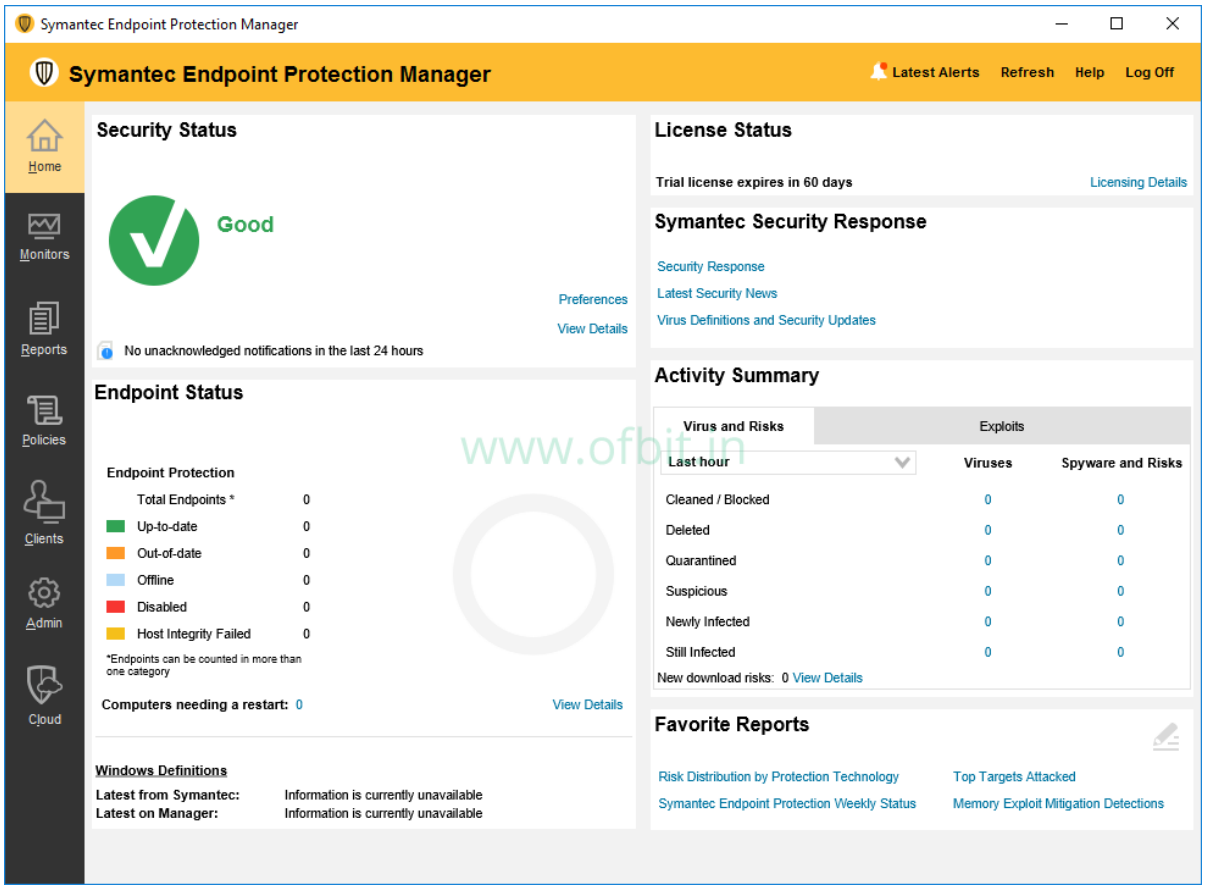

Рисунок 1.4 – робоча область програмного застосунку

Symantec Endpoint Encryption враховує необхідність захисту від втрати даних (DLP) та інтегрується з іншими системами управління ключами та сертифікатами. Це робить продукт універсальним і ефективним засобом для забезпечення конфіденційності та цілісності даних у корпоративних інфраструктурах.

1.3.3. Система моніторингу та аудиту безпеки Splunk

Splunk - це платформа для обробки та аналізу великих обсягів даних, зокрема журналів подій, що генеруються в реальному часі. Вона використовується для моніторингу, виявлення вторгнень, аналізу даних безпеки, оптимізації додатків та багато іншого.

Переваги Splunk:

- Реалізація Великих Даних: Splunk дозволяє обробляти, збирати та аналізувати великі обсяги даних в реальному часі.

- Універсальність: Splunk може аналізувати дані з різних джерел, включаючи журнали подій, дані безпеки, дані мережі, додатки та інше.

- Пошук та кореляція даних: Можливості потужного пошуку та кореляції дозволяють знаходити зв’язки між різними подіями та даними.

- Реалізація в реальному Часі: Splunk надає можливість аналізувати дані в реальному часі, що важливо для виявлення вторгнень та інших подій в мережі в момент їхнього виникнення.

- Легко налаштовується та спосіб використання: Інтуїтивний інтерфейс користувача спрощує налаштування та використання Splunk.

Розширюваність: Можливість додавання різних додатків та розширень для покращення функціональності.

Недоліки Splunk:

- Вартість: Splunk може бути високою за великих обсягів даних та розширені функціональності, що робить його менш доступним для деяких компаній.

- Складність для новачків: Налаштування та оптимізація Splunk може бути складною для новачків.

- Вимоги до ресурсів: Великі обсяги даних вимагають значних ресурсів для ефективної роботи Splunk.Обмежена аналітика машинного навчання: У порівнянні з деякими іншими платформами, Splunk може мати обмежені засоби для реалізації аналітики машинного навчання.

Вертикальна Масштабованість: Деякі користувачі вказують на обмежену вертикальну масштабованість у роботі з великими обсягами даних.

Необхідно враховувати, що переваги та недоліки Splunk можуть змінюватися з часом залежно від оновлень, нових версій та інших факторів.

1.4. Висновки до розділу 1

Ефективний захист автоматизованих систем управління критичною інфраструктурою є критично важливим аспектом забезпечення безпеки ключових об’єктів. Це вимагає централізованого управління та моніторингу для оперативного реагування на потенційні загрози в реальному часі. Застосування шифрування, систем виявлення вторгнень та антивірусного захисту є обов’язковими елементами для запобігання несанкціонованому доступу та забезпечення інтегритету даних. Автоматизоване виявлення аномалій, системи резервного копіювання та відновлення дозволяють підтримувати безперебійну роботу системи. Крім того, важливо надавати персоналу високий рівень освіти та стажування для ефективного реагування на нові кіберзагрози. Співпраця з іншими структурами та дотримання стандартів безпеки є важливими аспектами для забезпечення взаємодії та високого рівня захисту. Захист АСУ КІ повинен бути постійно оцінюваним та оновлюваним, враховуючи змінюючіся технологічні та кібербезпекові виклики. Загалом, цей підхід допомагає створити комплексну систему безпеки, яка гарантує надійність та функціональність АСУ КІ у змінюючихся умовах кіберзагроз.

2. РОЗРОБКА МЕТОДУ ЗАБЕЗПЕЧЕННЯ КОНФІДЕНЦІЙНОСТІ ІНФОРМАЦІЇ В АСУ КРИТИЧНОЇ ІНФРАСТРУКТУРИ

Реалізуємо захист за допомогою програмного застосунку Snort.

Snort використовується для моніторингу мережевого трафіку в реальному часі з метою виявлення аномальної чи підозрілої активності. Це досягається завдяки використанню правил, які описують сигнатури зловмисних дій або відомих атак. Кожен пакет даних, що проходить через систему, порівнюється з цими правилами, і якщо виявляється відповідність, спрацьовує виявлення вторгнення.

Однією з ключових переваг Snort є його відкритий вихідний код, що дозволяє спеціалістам у галузі безпеки переглядати та модифікувати його код з метою вдосконалення або адаптації до конкретних потреб АСУ. Це також сприяє швидкому виявленню та виправленню можливих уразливостей.

Snort використовує архітектуру з модулями, що дозволяє розширювати його можливості. За допомогою цієї архітектури можна додавати різні модулі, такі як розширені фільтри, системи виявлення і запобігання вторгненням (IPS), або інші функції, що роблять Snort більш гнучким та налаштовуваним під конкретні потреби АСУ.

Важливим аспектом є існування активної спільноти користувачів Snort, яка регулярно оновлює правила виявлення для врахування нових загроз та атак. Це забезпечує актуальний рівень захисту для АСУ і дозволяє ефективно протидіяти постійно змінюючимся кіберзагрозам.

Загалом, Snort володіє солідною репутацією як ефективний інструмент для виявлення вторгнень у мережах, і його застосування в області захисту АСУ відображає важливість надійного та постійного моніторингу безпеки для забезпечення стабільності та безпеки систем управління.

2.1. Встановлення та налаштування Snort

Для початку ми оновимо систему Ubuntu:

```
sudo apt update
sudo apt upgrade
```

У репозиторії Ubuntu є пакет snort. Доступний пакет snort – це стара версія. Щоб встановити Snort 3, нам потрібно виконати збірку з вихідних джерел. Перед встановленням Snort 3 нам потрібно встановити необхідні компоненти та бібліотеки.

Встановимо пакети залежностей Snort 3 за допомогою наступної команди:

```
sudo apt install build-essential libpcap-dev libpcre3-dev libnet1-dev zlib1g-dev luajit hwloc libdnet-dev libdumbnet-dev bison flex liblzma-dev openssl libssl-dev pkg-config libhwloc-dev cmake cpputest libsqlite3-dev uuid-dev libcmocka-dev libnetfilter-queue-dev libmnl-dev autotools-dev libluajit-5.1-dev libunwind-dev
```

Після встановлення залежностей створюємо каталог, у якому ми компілюємо та зберігаємо вихідні файли для Snort, за допомогою наступної команди:

```
mkdir snort-source-files cd snort-source-files
```

Потім завантажимо та встановимо останню версію бібліотеки збору даних Snort (LibDAQ). Для встановлення LibDAQ нам необхідно зібрати та встановити його з вихідного коду за допомогою наступної команди.

```
git clone https://github.com/snort3/libdaq.git cd libdaq ./bootstrap ./configure make
make install
```

Наступна залежність - Tcmalloc, яка оптимізує розподіл пам’яті та забезпечує краще використання пам’яті.

Встановимо Tcmalloc за допомогою наступної команди.

```
cd ../
wget
https://github.com/gperftools/gperftools/releases/download/gperftools-
2.9/gperftools-2.9.tar.gz tar xzf gperftools-2.9.tar.gz
cd gperftools-2.9/ ./configure make
make install
```

Після налаштування залежностей ми збираємося завантажити та встановити Snort 3 на Ubuntu 20.04.

Офіційний репозиторій Clone Snort 3 на GitHub:

```
cd ../
git clone git://github.com/snortadmin/snort3.git
```

Змінюємо каталог Snort3:

```
cd snort3/
```

Звідти налаштовуємо та вмикаємо tcmalloc за допомогою наступної команди:

```
./configure_cmake.sh --prefix=/usr/local --enable-tcmalloc
```

Перейдемо до каталогу збірки, скомпілюючи та встановивши Snort 3 за допомогою make та make install:

```
cd build
make
make install
```

Після завершення встановлення оновимо бібліотеки, що розділяються:

```
sudo ldconfig
```

За замовчуванням Snort встановлюється в каталог /usr/local/bin/snort, рекомендується створити символічне посилання для /usr/sbin/snort.

```
sudo ln -s /usr/local/bin/snort /usr/sbin/snort
```

Перевіримо налаштування Snort 3:

```
snort –V
```

Результат:

```
,,_     -> Snort++ <-
  o"  )~   Version 3.1.10.0

   ‘‘‘‘    By Martin Roesch & The Snort Team
           http://snort.org/contact#team
           Copyright (C) 2014-2021 Cisco and/or its affiliates. All
rights reserved.
           Copyright (C) 1998-2013 Sourcefire, Inc., et al.
           Using DAQ version 3.0.4
           Using LuaJIT version 2.1.0-beta3
           Using OpenSSL 1.1.1f  31 Mar 2020
           Using libpcap version 1.9.1 (with TPACKET_V3)
           Using PCRE version 8.39 2016-06-14
```

```
Using ZLIB version 1.2.11
Using LZMA version 5.2.4
```

Налаштуємо карти мережного інтерфейсу. Знайдемо інтерфейс, на якому Snort прослуховує мережевий трафік, і вмикаємо нерозбірливий режим, щоб мати можливість бачити весь мережний трафік, що відправляється йому.

```
ip link set dev eh0 promisc on
```

Перевіримо за допомогою наступної команди:

```
ip add sh eth0
```

Результат:

```
2: eth0: <BROADCAST,MULTICAST,PROMISC,UP,LOWER UP> mtu 1500 qdisc mq state UP group default qlen 1000
    link/ether f2:3c:92:ed:7e:d8 brd ff:ff:ff:ff:ff:ff
    inet 74.207.230.186/24 brd 74.207.230.255 scope global dynamic eth0
       valid_lft 72073sec preferred_lft 72073sec
    inet6 2600:3c02::f03c:92ff:feed:7ed8/64 scope global dynamic mngtmpaddr noprefixroute
       valid_lft 60sec preferred_lft 20sec
    inet6 fe80::f03c:92ff:feed:7ed8/64 scope link
       valid_lft forever preferred_lft forever
```

Потім відключимо розвантаження інтерфейсу, щоб Snort 3 не урізав великі пакети, не більше 1518 байт. Перевіримо, чи активовано цю функцію за допомогою наступної команди:

```
ethtool -k eth0 | grep receive-offload
```

Результат:

```
generic-receive-offload: on
large-receive-offload: on
```

Вимкнемо його за допомогою наступної команди:

```
ethtool -K eth0 gro off lro off
```

Зміни збережуться під час перезавантаження системи, нам потрібно створити та включити службовий модуль systemd для реалізації змін:

```
sudo nano /etc/systemd/system/snort3-nic.service
```

Вставимо наступну конфігурацію, яка вказує на мережний інтерфейс:

```
[Unit] Description=Set Snort 3 NIC in promiscuous mode and Disable GRO, LRO on boot After=network.target

[Service]
Type=oneshot
ExecStart=/usr/sbin/ip link set dev eth0 promisc on
ExecStart=/usr/sbin/ethtool -K eth0 gro off lro off
TimeoutStartSec=0
RemainAfterExit=yes

[Install]
WantedBy=default.target
```

Перезавантажуємо конфігураційні установки systemd:

```
sudo systemctl daemon-reload
```

Запускаємо і вмикаємо службу під час завантаження за допомогою наступної команди:

```
sudo systemctl enable --now snort3-nic.service
```

Результат:

```
Created symlink /etc/systemd/system/default.target.wants/snort3-nic.service → /etc/systemd/system/snort3-nic.service
```

Перевіримо snort3-nic.service за допомогою:

```
sudo systemctl status snort3-nic.service
```

Результат:

```
● snort3-nic.service - Set Snort 3 NIC in promiscuous mode and Disable GRO, LRO on boot
     Loaded: loaded (/etc/systemd/system/snort3-nic.service; enabled; vendor preset: enabled)
     Active: active (exited) since Sat 2021-09-18 12:35:17 UTC; 4min 59s ago
    Process: 182782 ExecStart=/usr/sbin/ip link set dev eth0 promisc on (code=exited, status=0>
    Process: 182783 ExecStart=/usr/sbin/ethtool -K eth0 gro off lro off (code=exited, status=0>
   Main PID: 182783 (code=exited, status=0/SUCCESS)
Sep 18 12:35:17 li72-186 systemd[1]: Starting Set Snort 3 NIC in promiscuous mode and Disable >
Sep 18 12:35:17 li72-186 systemd[1]: Finished Set Snort 3 NIC in promiscuous mode and Disable >
```

У Snort набори правил є основною перевагою механізму виявлення вторгнень. Існує три типи правил Snort: правила спільноти, зареєстровані правила, правила передплатника.

Правила спільноти представлені спільнотою розробників програмного забезпечення з відкритим вихідним кодом або інтеграторами snort.

Спочатку створимо каталог для правил у /usr/local/etc/snort:

```
mkdir /usr/local/etc/rules
```

Завантажимо правила спільноти Snort 3:

```
wget https://www.snort.org/downloads/community/snort3-community-rules.tar.gz
```

Виймемо завантажені правила та помістимо їх у раніше створений каталог /usr/local/etc/rules/:

```
tar xzf snort3-community-rules.tar.gz -C /usr/local/etc/rules/
```

Snort 3 включає два основні файли конфігурації, snort_defaults.lua та snort.lua.

Snort.lua файл містить основну конфігурацію пирхання, що дозволяє реалізацію та налаштування Snort препроцесорами, правила включення файлів, реєстрацію фільтрів подій, вихід і т.д.

У snort_defaults.lua файли містять значення за промовчанням , такі як шляхи до правил, AppID, списків розвідки та мережних змінних.

Коли файли правил будуть вилучені та розміщені, ми збираємось налаштувати один із цих файлів конфігурації під назвою snort.lua.

Відкриємо файл редактором та побачимо щось схоже на конфіг.

```
... -- HOME_NET and EXTERNAL_NET must be set now -- setup the network addresses you are protecting HOME_NET = ‘server_public_IP/32’ -- set up the external network addresses. -- (leave as "any" in most situations) EXTERNAL_NET = ‘any’ EXTERNAL_NET = ‘!$HOME_NET’ ...
```

Задаємо мережу, яку ми хочемо захистити від атак, як значення змінної HOME_NET і вкажіть змінну EXTERNAL_NET на змінну HOME_NET.

Ми також можемо відредагувати налаштування Snort за умовчанням у /usr/local/etc/snort/snort_defaults.lua, а в розділі IPS ми можемо визначити місцезнаходження ваших правил:

```
ips = {
-- use this to enable decoder and inspector alerts
--enable_builtin_rules = true,

-- use include for rules files; be sure to set your path
-- note that rules files can include other rules files
include = ‘/usr/local/etc/rules/snort3-community-rules/snort3-community.rules’ } ...
```

Ми збираємося запускати Snort як сервісний демон у фоновому режимі, можна створити сервісний модуль systemd для Snort. Розумно запускати його від імені непривілейованого користувача системи.

Створимо обліковий запис користувача без входу до системи:

```
sudo useradd -r -s /usr/sbin/nologin -M -c SNORT_IDS snort
```

Потім створимо службовий модуль systemd для Snort, який запускатиметься від імені користувача snort. Відрегулюємо та зіставимо з нашим мережевим інтерфейсом:

```
sudonano /etc/systemd/system/snort3.service
```

Вставимо наступну конфігурацію:

```
[Unit] Description=Snort 3 NIDS Daemon After=syslog.target network.target [Service] Type=simple ExecStart=/usr/local/bin/snort -c /usr/local/etc/snort/snort.lua -s 65535 -k none -l /var/log/snort -D -i eht0 -m 0x1b -u snort -g snort [Install] WantedBy=multi-user.target
```

Перезавантажимо конфігурацію systemd:

```
sudo systemctl daemon-reload
```

Встановимо власника та дозволи для файлу журналу:

```
sudo chmod -R 5775 /var/log/snort
sudo chown -R snort:snort /var/log/snort
```

Запустимо та увімкнимо Snort для запуску під час завантаження системи:

```
sudo systemctl enable --now snort3
```

Перевіримо статус служби, щоб переконатися, що її запущено:

```
sudo systemctl status snort3
```

Результат:

```
snort3.service - Snort 3 NIDS Daemon
      Loaded: loaded (/etc/systemd/system/snort3.service; enabled; vendor preset: enabled)
      Active: active (running) since Sat 2021-09-18 12:44:32 UTC; 6s ago
    Main PID: 182886 (snort)
```

```
             Tasks: 2 (limit: 1071)
            Memory: 62.6M
            CGroup: /system.slice/snort3.service
                    └─182886             /usr/local/bin/snort            -c
/usr/local/etc/snort/snort.lua -s 65535 -k none >
      Sep 18 12:44:32 li72-186 systemd[1]: Started Snort 3 NIDS Daemon.
```

2.2. Інтеграція Barnyard2

Barnyard2 - це програма, яка використовується у співпраці з системою виявлення вторгнень Snort з метою обробки та передачі журналів подій в інші системи, такі як системи управління журналами подій (SIEM) чи бази даних. Вона розширює функціональність Snort, допомагаючи витягати дані про виявлені події та атаки з журналів Snort для подальшої обробки та аналізу.

Основна роль Barnyard2 полягає в ефективній обробці журналів, зокрема журналів Snort, для подальшої передачі цих даних до централізованих систем аналізу та моніторингу. Вона дозволяє передавати оброблені дані до різних систем, таких як SIEM-платформи, бази даних чи інші зберігальні системи.

Barnyard2 оптимізований для роботи з великими обсягами даних, розділяючи функції обробки подій та системи виявлення вторгнень. Це допомагає покращити продуктивність та ефективність в обробці журналів Snort.

Додатково, Barnyard2 підтримує різні формати журналів, включаючи файли у форматі Unified2 та ASCII, що робить його гнучким у використанні з різними конфігураціями Snort.

Використання Barnyard2 з Snort дозволяє підвищити здатність системи виявлення вторгнень співпрацювати з іншими системами та інфраструктурою безпеки, що є важливим у сучасних умовах зростаючого обсягу кіберзагроз та потреб у централізованому моніторингу подій.

Для його встановлення спершу ставимо необхідні компоненти:

```
sudo apt-get install -y mysql-server libmysqlclient-dev mysql-client autoconf libtool
```

Звернемо увагу, що під час встановлення MySQL попросить ввести пароль для root. Потім ставимо сам Barnyard2:

```
cd ~/snort_src
wget https://github.com/firnsy/barnyard2/archive/master.tar.gz -O barnyard2-Master.tar.gz
tar zxvf barnyard2-Master.tar.gz
cd barnyard2-master
autoreconf -fvi -I ./m4
sudo ln -s /usr/include/dumbnet.h /usr/include/dnet.h
sudo ldconfig
./configure --with-mysql --with-mysql-libraries=/usr/lib/i386-linux-gnu
make
sudo make install
```

Перевіряємо:

```
/usr/local/bin/barnyard2 –V
```

Знову копіюємо файли з вихідного пакета, створюємо файли та роздаємо права:

```
sudo cp ~/snort/barnyard2-master/etc/barnyard2.conf /etc/snort/
sudo mkdir /var/log/barnyard2
sudo chown snort.snort /var/log/barnyard2
sudo touch /var/log/snort/barnyard2.waldo
sudo chown snort.snort /var/log/snort/barnyard2.waldo
```

Тепер доведеться попрацювати з MySQL:

```
mysql -u root -p
mysql> create database snort;
mysql> use snort;
mysql> source ~/snort/barnyard2-master/schemas/create_mysql
mysql> CREATE USER ‘snort’@’localhost’ IDENTIFIED BY ‘snortpass’;
mysql> grant create, insert, select, delete, update on snort.* to ‘snort’@’localhost’;
mysql> exit
```

Додаємо в кінець файлу /etc/snort/barnyard2.confрядок output database: `log, mysql, user=snort password=snortpass dbname=snort host=localhost sensor name=sensor01`. І знову маніпуляція із правами:

```
sudo chmod o-r /etc/snort/barnyard2.conf
```

Наразі перевіримо все, що зробили раніше. У файл /etc/snort/rules/local.rulesдодаємо рядок

```
alert icmp any any -> $HOME_NET any (msg:"ICMP test detected"; GID:1; sid:10000001; rev:001; classtype:icmp-event;)
```

Це буде наше правило для перевірки пінгу (ICMP-пакети). Дописуємо два рядки у файл /etc/snort/sid-msg.map:

```
`#v2`
1 || 10000001 || 001 || icmp-event || 0 || ICMP Test detected || url,tools.ietf.org/html/rfc792
```

Запустимо Snort у режимі демона (він так і працюватиме завжди).

```
sudo /usr/local/bin/snort -q -u snort -g snort -c /etc/snort/snort.conf -i enp0s3 –D
```

Потім запустимо Barnyard2:

```
sudo barnyard2 -c /etc/snort/barnyard2.conf -d /var/log/snort -f snort.u2 -w /var/log/snort/barnyard2.waldo -g snort -u snort
```

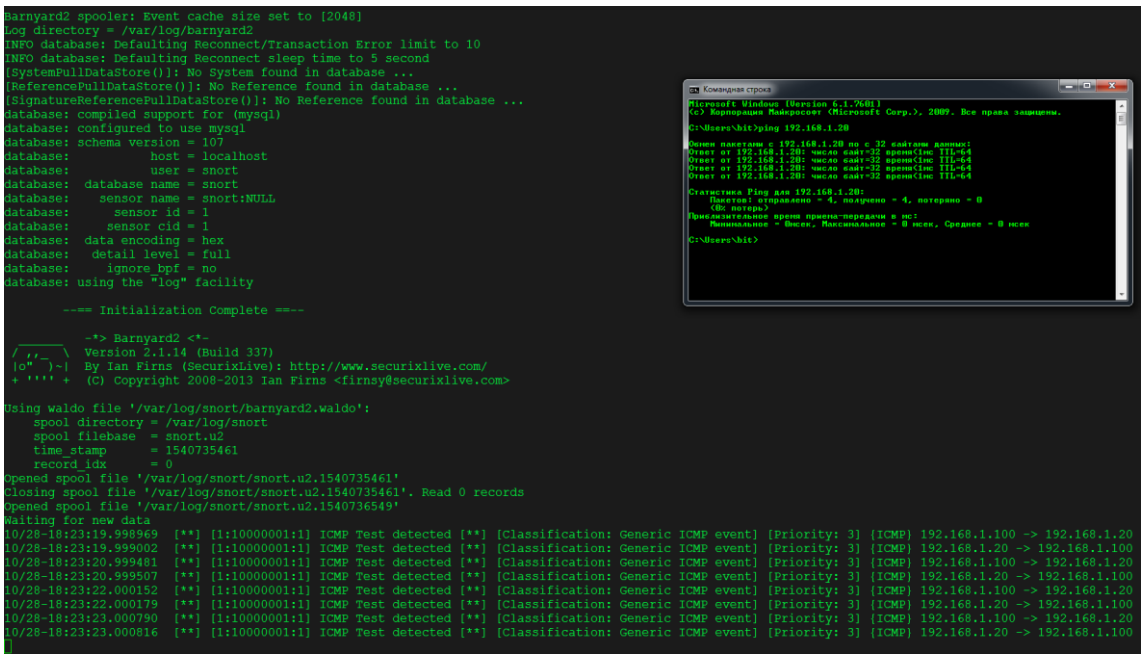

Рисунок 3.1 – робота Barnyard2 та Snort

2.3. Інтеграція PulledPork

PulledPork - це інструмент, розроблений для автоматизації процесу управління правилами Snort та Suricata, систем виявлення вторгнень. Він спрощує завдання оновлення та впровадження сигнатур, що використовуються для виявлення кіберзагроз та атак.

Головною функцією PulledPork є завантаження та оновлення сигнатурних баз для Snort та Suricata, які визначають підозрілі чи шкідливі пакети в мережі. Він отримує оновлення з офіційних репозиторіїв, забезпечуючи актуальність бази даних сигнатур та підтримуючи захист від нових загроз.

PulledPork дозволяє зручно керувати конфігурацією правил, включаючи вибір конкретних категорій сигнатур, визначення правил для виключення чи включення, а також налаштування параметрів застосування сигнатур.

Інструмент також може автоматично перезапускати систему виявлення вторгнень після оновлення, щоб забезпечити негайне застосування нових правил.

Використання PulledPork дозволяє адміністраторам мереж зосередитися на ефективному виявленні та реагуванні на кіберзагрози, замість того, щоб витрачати час на ручне оновлення та конфігурування сигнатурних баз. Такий підхід сприяє підвищенню безпеки мереж та зменшенню ризиків забруднення систем зловмисними атаками.

Завантажуємо правила для Snort:

```
sudo apt-get install -y libcrypt-ssleay-perl liblwp-useragent-determined-perl
cd ~/snort
wget https://github.com/shirkdog/pulledpork/archive/master.tar.gz -O pulledpork-master.tar.gz
tar xzvf pulledpork-master.tar.gz
cd pulledpork-master/

sudo cp pulledpork.pl /usr/local/bin
sudo chmod +x /usr/local/bin/pulledpork.pl
sudo cp etc/*.conf /etc/snort
```

Запускаємо PulledPork:

```
sudo /usr/local/bin/pulledpork.pl -c /etc/snort/pulledpork.conf –l
```

Спостерігаємо, як завантажуються правила. Перевіряємо, як відпрацює наш Snort після зміни конфігу.

```
sudo snort -T -c /etc/snort/snort.conf -i enp0s3
```

PulledPork сам перевірятиме наявність оновлень та завантажуватиме правила. Необхідно лише додати команду на його запуск у планувальник:

```
sudo crontab -e
03 02 * * * /usr/local/bin/pulledpork.pl -c /etc/snort/pulledpork.conf –l
```

Важливо, що команда розробників Snort просить рандомно вказувати час звернення завантажувача, щоб розподілити навантаження на канал.

2.4. Інтеграція Basic Analysis and Security Engine

Basic Analysis and Security Engine (BASE) - це вільно поширюване веб-середовище, створене для аналізу та візуалізації журналів подій систем виявлення вторгнень (IDS) та систем безпеки. BASE спроектовано для спрощення завдань адміністрування та аналізу подій в мережі, надаючи зручний інтерфейс для взаємодії з великим обсягом журнальних даних.

Основна мета BASE полягає в тому, щоб надати адміністраторам та аналітикам засіб для ефективного вивчення, аналізу та візуалізації подій, що стосуються безпеки мережі. Інтерфейс BASE інтегрується з різними системами виявлення вторгнень, такими як Snort та Suricata, забезпечуючи зручну взаємодію з різноманітними джерелами журналів подій.

Важливим елементом BASE є гнучкість у виборі часових інтервалів для аналізу, що дозволяє користувачам досліджувати події в різні періоди часу. Система також надає графічні візуалізації, діаграми та графіки для легкості розуміння статистики та забезпечення швидкого виявлення аномалій.

Додатково, BASE дозволяє здійснювати ефективний пошук та фільтрацію даних, щоб користувачі могли зосередитися на конкретних аспектах безпеки

мережі. Забезпечується можливість експорту даних для подальшого аналізу та збереження.

Система підтримує автентифікацію та авторизацію, дозволяючи обмежувати доступ до даних лише авторизованим користувачам. BASE є інструментом, який сприяє покращенню безпеки мережі, роблячи процес аналізу та візуалізації подій більш доступним та ефективним.

Приступаємо до встановлення і налаштування:

```
sudo add-apt-repository ppa:ondrej/php
sudo apt-get update
sudo apt-get install -y apache2 libapache2-mod-php5.6 php5.6-mysql php5.6-cli php5.6 php5.6-common php5.6-gd php5.6-cli php-pear php5.6-xml
sudo pear install -f --alldeps Image_Graph
```

Завантажуємо бібліотеку ADODB:

```
cd ~/snort
wget https://sourceforge.net/projects/adodb/files/adodb-php5-only/adodb-520-for-php5/adodb-5.20.8.tar.gz
tar -xvzf adodb-5.20.8.tar.gz
sudo mv adodb5 /var/adodb
sudo chmod -R 755 /var/adodb
```

Завантажуємо BASE:

```
cd ~/snort
wget http://sourceforge.net/projects/secureideas/files/BASE/base-1.4.5/base-1.4.5.tar.gz
tar xzvf base-1.4.5.tar.gz
sudo mv base-1.4.5 /var/www/html/base/
```

Копіюємо конфігураційний файл:

```
cd /var/www/html/base
sudo cp base_conf.php.dist base_conf.php
```

І наводимо деякі рядки у файлі /var/www/html/base/base_conf.phpдо зразків:

```
$BASE_urlpath   = '/base';
$DBlib_path     = '/var/adodb/';
$alert_dbname   = 'snort';
$alert_host     = 'localhost';
$alert_port     = '';
$alert_user     = 'snort';
$alert_password = 'snortpass';
```

Звичайно, необхідно змінити права, щоб ніхто не побачив пароль у файлі:

```
sudo chown -R www-data:www-data /var/www/html/base
sudo chown -R www-data:www-data /var/www/html/base
```

Перезапускаємо Apache. Відкриваємо браузер і йдемо за адресою (айпишник вказуй свій) http://192.168.1.20/base/index.php. Натискаємо кнопку Create BASE AG у верхньому правому кутку.

2.5. Тестування працездатності методу

Для перевірки працездатності я взяв трохи неординарне завдання, щоб показати трохи більше можливостей Snort. Наприклад, нам необхідно відстежити відвідування сайту якимось користувачем (будь-яким з локальної мережі). Для цього нам потрібно створити правило, яке відслідковувало б цю дію. Воно буде виглядати так:

```
alert tcp any any -> any any  (content: "www.xakep.ru"; msg: "Someone is visiting site now"; sid:1000008; rev:1)
```

Запишемо його /etc/snort/rules/local.rulesi перезапустимо моніторинг. Так як Snort у мене відстежує лише мій сервер, я вийшов на сайт за допомогою Links із сервера.

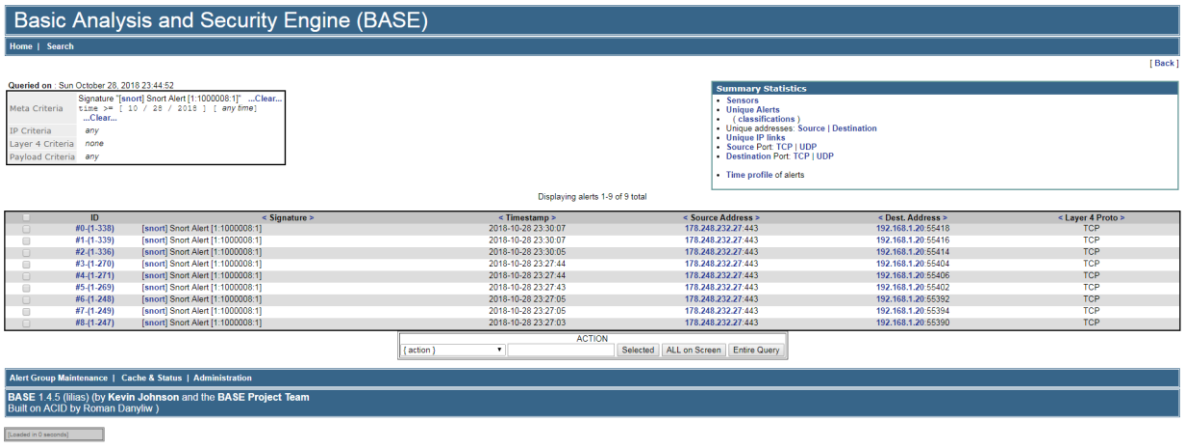

Рисунок 3.2 – графічне сповіщення про відвідування сайту

В результаті ми побачили кілька оповіщень - стільки, скільки разів я підключався до цього домену. Якщо неправильно написати правило, Snort не запуститься, видавши помилку та опис, чому він не зможе запуститися.

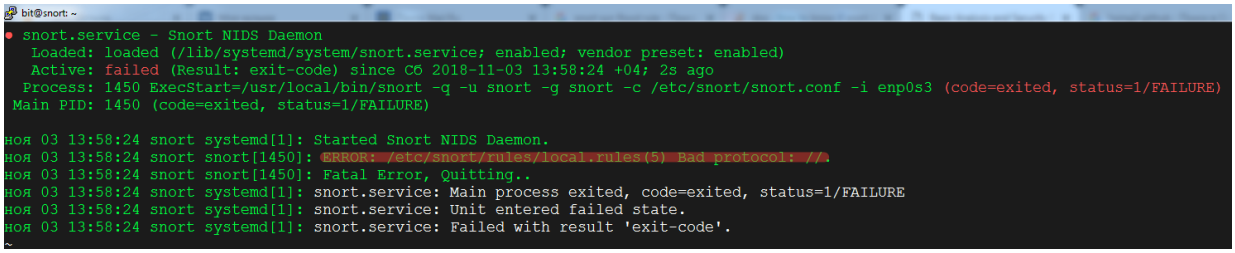

Рисунок 3.3 – помилка запуску Snort

Встановити та перевірити на працездатність – цього мало. Потрібно його ще й переглядати. Ось давай і перевіримо, на що здатна ця IDS із заводськими правилами. У нас є SSH, Apache та FTP. Значить, мінімальний набір правил, які треба мати - це виявляти сканування портів, атаку на облікові записи (brute force), DoS та SQL-ін’єкції.

Припустимо, що на цьому етапі з’явився хакер. Він зробив свою справу і зник, але відразу в логах відобразилася підозріла активність з однієї IP-адреси. У мене відображається 3964 події з п’ятьма різними діями. Подивившись уважніше, можна побачити незграбну картину.

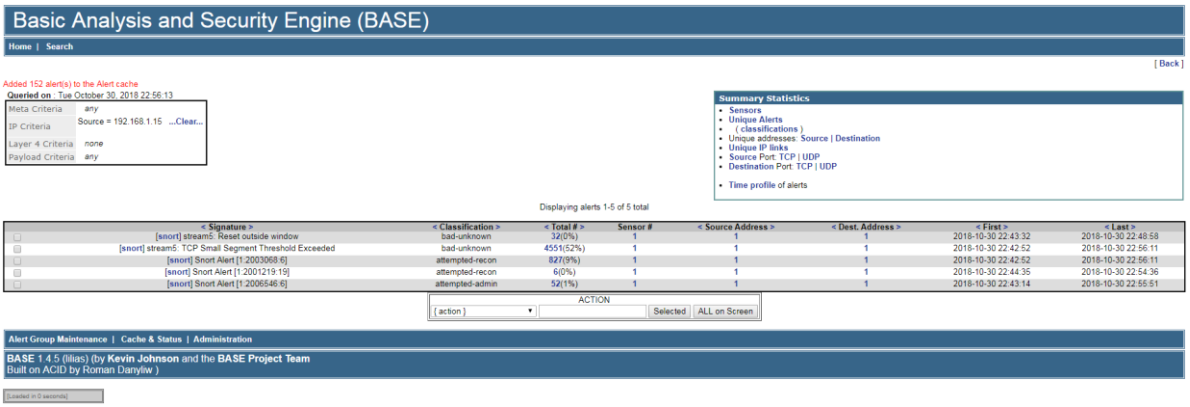

Рисунок 3.4 – відображення дій підозрілої IP-адреси

Неозброєним оком видно сканування портів, або, як це розробники назвали, «розвідувальну діяльність». Дивимося далі:

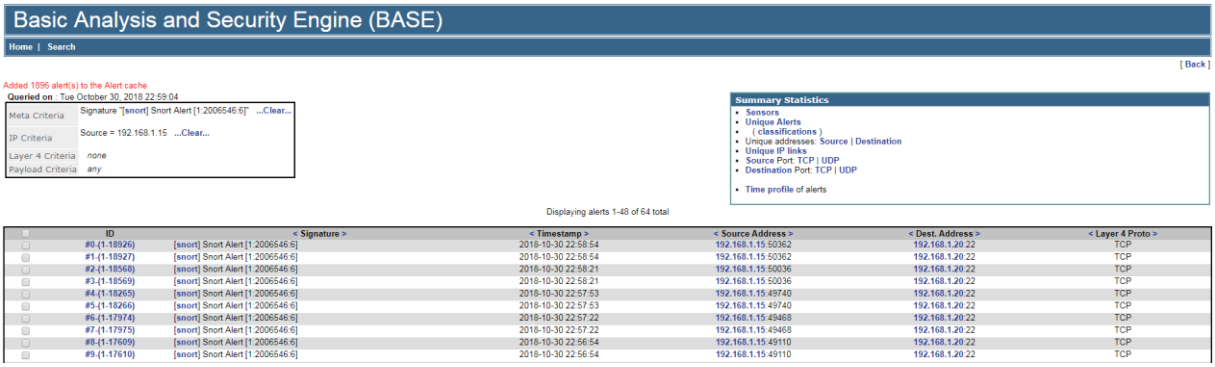

Рисунок 3.5 – відображення атаки на 22-ий порт

Видно брутфорс SSH-сервісу на 22-му порту. Так само справи і з FTP-портом (21-м).

Далі ведеться розслідування, переглядаємо логи самої ОС, виявляємо слабку ланку захисту, звідки вона прийшла і таке інше. Головне, що оповіщення спрацювали і причетність даного IP до протиправних дій встановлено.

Піднімаємо Kali Linux і намагаємося провести DoS-атаку на наш Apache. Так як BASE працює по HTTP, найлегший спосіб його «покласти», як на мене, –

це атака Slowloris . Тому скачуємо на нашу Kali відповідний скрипт та запускаємо атаку.

```
root@kali:~/x# git clone https://github.com/gkbrk/slowloris.git
Клонирование в «slowloris»…
remote: Enumerating objects: 6, done.
remote: Counting objects: 100% (6/6), done.
remote: Compressing objects: 100% (6/6), done.
remote: Total 78 (delta 0), reused 3 (delta 0), pack-reused 72
Распаковка объектов: 100% (78/78), готово.
root@kali:~/x# cd slowloris/
root@kali:~/x/slowloris# python3 slowloris.py 192.168.1.20
[02-11-2018 22:06:26] Attacking 192.168.1.20 with 150 sockets.
[02-11-2018 22:06:26] Creating sockets...
[02-11-2018 22:06:30] Sending keep-alive headers... Socket count: 150
[02-11-2018 22:06:45] Sending keep-alive headers... Socket count: 150
[02-11-2018 22:07:00] Sending keep-alive headers... Socket count: 150
^C
Stopping Slowloris...
```

Рисунок 3.6 – DoS Slowloris на Kali Linux

За 15–30 секунд наш BASE перестає відповідати. І навіть після зупинки атаки довелося перезапустити веб-сервер. Сам він за короткий проміжок часу не зміг підвестися. Впав лише BASE. Snort, природно, продовжував працювати і записувати оповіщення до бази.

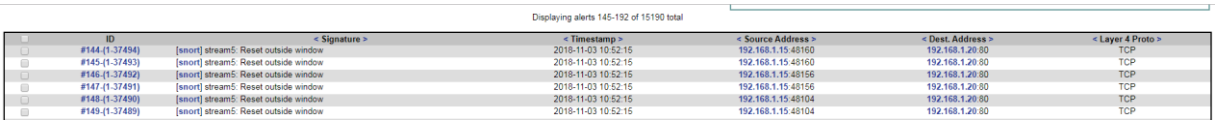

Рисунок 3.7 – Лог Snort після DoS

Ось так виглядає наш лог після атаки. Потім я спробував зафлудити SYN-пакетами свій Snort-сервер. Зроблено це було за допомогою програми hping3.

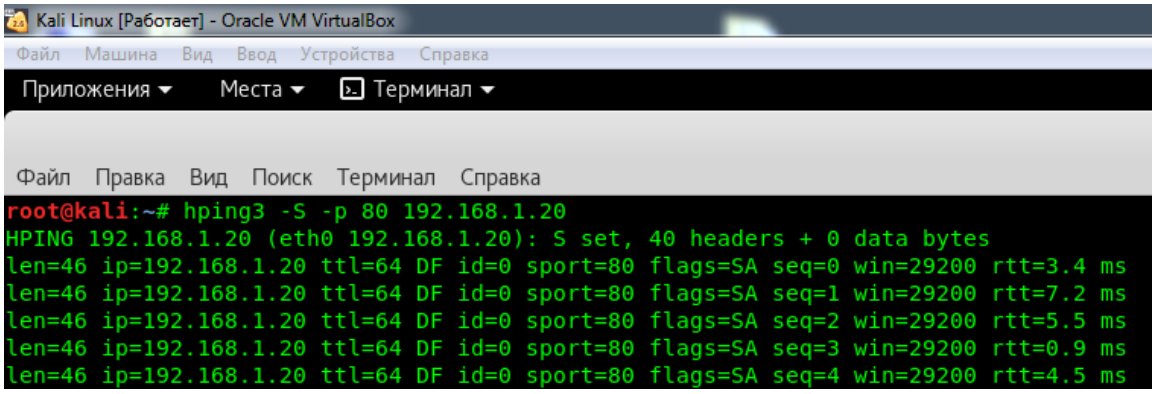

Рисунок 3.8 – Флуд сервера SYN-пакетами

На жаль, з поточними налаштуваннями Snort не в змозі було визначити таку атаку і в логах нічого не показав. Довелося дивитися через tcpdump, чи пакети йдуть на сервер взагалі.

Наступним етапом я нацькував sqlmap на BASE на запит http://192.168.1.20/base/base_stat_alerts.php?sensor=1. На жаль, це теж не дало результатів: відповідних правил визначення атак на базу даних у дефолтній добірці Snort немає. Отже, треба написати правило самому. У нашому прикладі воно виглядає так:

```
alert tcp any any -> any any (msg: "SQL Injection"; content: "GET"; http_method; uricontent: "and 1=1"; nocase; sid:3000001; rev:1;)
```

Тепер зайдемо на web-інтерфейс з комп'ютера атакуючого і спробуємо проексплуатувати цей метод, додавши наприкінці запиту `and 1=1`. У лозі відобразилося повідомлення з номером правила.

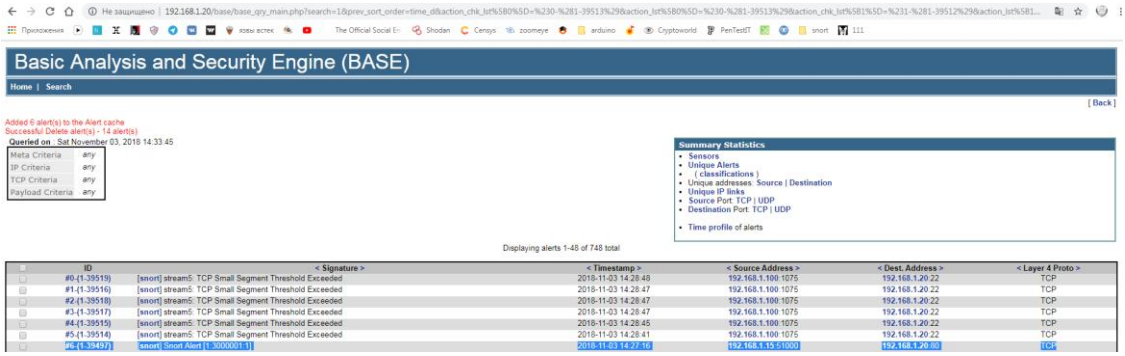

Рисунок 3.9 – Флуд сервера SYN-пакетами

2.6. Висновки до розділу 2

Snort - це потужний інструмент системи виявлення вторгнень та запобігання вторгненням (IDS/IPS), який визначається своєю здатністю виявляти та реагувати на потенційні атаки у мережі.

Його функціонал включає в себе виявлення вторгнень на основі сигнатур, що дозволяє ідентифікувати вже відомі атаки, а також аналіз та виявлення аномальностей у мережевому трафіку.

Snort пропонує гнучкий підхід до роботи, дозволяючи використовуватися як в режимі виявлення вторгнень, так і в режимі запобігання вторгненням, що дає можливість адаптувати його до конкретних потреб безпеки мережі.

Модульність та можливість розширення функціоналу дозволяють інтегрувати Snort з іншими системами та розробляти додаткові модулі для покращення його можливостей.

Інтеграція з активною спільнотою користувачів сприяє регулярним оновленням сигнатур та забезпечує підтримку, що важливо для ефективності системи виявлення вторгнень у змінних умовах загроз.

Враховуючи його відкритий вихідний код, Snort стає доступним інструментом для розробників та адміністраторів, які можуть адаптувати його до своїх конкретних потреб у кібербезпеці.

Узагальнюючи, Snort визначається своєю ефективністю, гнучкістю та здатністю пристосовуватися до зростаючих загроз у сфері кібербезпеки.

На мій погляд, NIDS необхідна на підприємстві. Мені зустрічалися великі компанії, які не обтяжували себе встановленням подібної ланки захисту.

Snort лише система виявлення мережевих атак. У зв’язку з недбайливим адміністратором вона просто марна. Відразу після установки Snort запускається неналаштованою, з великою кількістю включених правил, що викликає море помилкових спрацьовувань. Та й установка не з найпростіших.

Зате після всіх доналаштувань та написання своїх правил вона стає досить потужним інструментом.

ВИСНОВОК

В магістерській роботі, присвяченій розробці методу забезпечення конфіденційності інформації в автоматизованих системах управління критичної інфраструктури, виявлено та проаналізовано ключові аспекти кібербезпеки, зокрема в контексті важливості надійного функціонування систем, що керують критично важливими процесами.

Однією з ключових висновків роботи є те, що захист інформації в АСУ критичної інфраструктури є завданням високої важливості в умовах зростаючих кіберзагроз. Аналіз сучасних технічних та організаційних рішень вказує на необхідність комплексного підходу, що охоплює технічні заходи, політики безпеки, моніторинг та освіту персоналу.

Також виявлено, що вибір ефективних технічних засобів для виявлення та захисту від кіберзагроз, таких як системи виявлення вторгнень та засоби шифрування, має велике значення для забезпечення цілісності, конфіденційності та доступності інформації.

У роботі також враховано важливість взаємодії з організаційними структурами, нормативними вимогами та стандартами безпеки для підвищення рівня відповідності та готовності до вирішення можливих інцидентів.

Для реалізації нового методу щодо виявлення та протидії несанкціонованого доступу до АСУ було використано програмний застосунок Snort.

Snort використовує гнучкі правила для виявлення аномальної активності в мережі, дозволяючи аналізувати пакети на предмет підозрілих патернів та сигнатур. Це робить його ефективним інструментом для виявлення різноманітних кіберзагроз, включаючи атаки на вторгнення, атаки на викидання, та інші.

Також, можемо підкреслити гнучкість Snort у налаштуванні та розширенні, що дозволяє адаптувати його під конкретні потреби мережі та забезпечує можливість інтеграції з іншими системами безпеки.

Зазначимо, що, не дивлячись на свою ефективність, Snort має свої обмеження та вимоги, які слід враховувати при його використанні. Потрібно розглядати його як один із компонентів більшої стратегії кібербезпеки, і узгоджувати його роботу з іншими інструментами та заходами захисту, що й було виконано у ході налаштування його роботи і захисту АСУ в цілому.

В ході виконання кваліфікаційної роботи було проаналізовані сучасні рішення щодо забезпечення конфіденційності інформації в автоматизованих системах управління, досліджені потенційних загроз для АСУ критичної інфраструктури, за допомогою наявних методів було розроблено метод для відслідковування та протидії щодо втручання в АСУ, що надало змогу забезпечити конфіденційність та надійне зберігання інформації. В ході роботи було залучено кілька методів відслідковування потенційних загроз і спроб несанкціонованого втручання в систему.

Загалом, дана магістерська робота спрямована на вдосконалення розуміння та реалізацію заходів кібербезпеки в АСУ критичної інфраструктури, надаючи важливий внесок у сферу безпеки в цьому важливому контексті.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Промислові мережі та інтеграційні технології в автоматизованих системах: [навч. посіб.] / Пупена О.М. [та ін.] К. : Вид-во «Ліра-К», 2011. 552 с.

2. Правила для безпеки інформаційних систем та мереж: До культури безпеки 2016. [Електронний ресурс] – Режим доступу до ресурсу: http://www.ftc.gov/bcp/conline/edcams/infosecurity/popups/OECD guidelines.pdf.

3. Schjolberg S., Ghernaouti-Hlie S. Глобальний договір про кібербезпеку та кіберзлочинство, Друге видання. [Електронний ресурс] – Режим доступу до ресурсу: http://www.cybercrimelaw.net/documents/.

4. Автоматизація виробничих процесів: Підручник / І.В. Ельперін, О.М. Пупена, В.М. Сідлецький, С.М. Швед . К.: Ліра-К, 2015. 378 с.

5. Організація комп’ютерних мереж: Підручник / КПІ ім. Ігоря Сікорського ; Ю. А. Тарнавський, І. М. Кузьменко. – Київ: КПІ ім. Ігоря Сікорського, 2018. 259с.

6. Лукінюк М.В. Автоматизація типових технологічних процесів: технологічні об’єкти керування та схеми автоматизації: Навчальний посібник. – К.: НТУУ “КПІ”, 2008. 236 с.

7. Денисенко В.В. Комп'ютерне управління технологічним процесом, експериментом, обладнанням. М .: Гаряча лінія-Телеком, 2009. 608 с.

8. Автоматизовані системи керування технологічними процесами: Підручник / За редакцією І.О. Фурмана. Харків: Факт, 2006. 317 с.

9. Ігнатьєв А. А. Удосконалення управління якістю продукції на основі системи моніторінга з елементами штучного інтелекту / А. А. Ігнатьєв, Є. М. Самойлова // Вісник СГТУ. 2009. № 3(41). С. 207–209.

10. Островерхов М.Я., Сільвестров А. М., Скриннік О.М. Системи і методи ідентифікації електротехнічних об’єктів: Монографія. К.: НАУ, 2016. 324 с.

11. . Бобух А.О. Автоматизовані системи керування технологічними процесами: Навчальний посібник. Харків: ХНАМГ, 2006. 185 с.

12. Пупена О.М. Розроблення людино-машинних інтерфейсів та систем збирання даних з використанням програмних засобів SCADA/HMI: Посібник. Київ, Ліра-К, 2020. 594 с.

13. С.В. Бейцун. Основи комп’ютерно-інтегрованого управління: Конспект лекцій та методичні вказівки до індивідуального завдання. Днепропетровск : НМетАУ, 2009. 45с

14. Головко Д.Б., Рего К.Г., Скрипник Ю.О. Автоматика і автоматизація технологічних процесів: Підручник. К.: Либідь, 1997. 232 с

15. О.П. Єгоров, М.О. Рибальченко, І.О. Маначин. Цифрові методи дослідження та розрахунку регуляторів в системах автоматичного керування : навчальний посібник. Дніпро : УДУНТ, 2022 .122 с