МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ДЕРЖАВНЕ НЕКОМЕРЦІЙНЕ ПІДПРИЄМСТВО
«ДЕРЖАВНИЙ УНІВЕРСИТЕТ
«КИЇВСЬКИЙ АВІАЦІЙНИЙ ІНСТИТУТ»
ФАКУЛЬТЕТ КОМП'ЮТЕРНИХ НАУК ТА ТЕХНОЛОГІЙ
КАФЕДРА КІБЕРБЕЗПЕКИ

ДОПУСТИТИ ДО ЗАХИСТУ
Завідувач кафедри кібербезпеки

_____Анна ІЛЬЄНКО
"_____"_____ 20__ р.

КВАЛІФІКАЦІЙНА РОБОТА

(ПОЯСНЮВАЛЬНА ЗАПИСКА)

ЗДОБУВАЧА ОСВІТНЬОГО СТУПЕНЯ "МАГІСТР"

Тема: Модуль кіберзахисту системи керування розумного будинку

Виконавець: Вячеслав ДЗЕЦІНА

Керівник: к.т.н. Олена ВИСОЦЬКА

Нормоконтролер: к.т.н., доцент Андрій ПЕТРЕНКО

Київ 2024

ДЕРЖАВНЕ НЕКОМЕРЦІЙНЕ ПІДПРИЄМСТВО
«ДЕРЖАВНИЙ УНІВЕРСИТЕТ
«КИЇВСЬКИЙ АВІАЦІЙНИЙ ІНСТИТУТ»

Факультет комп'ютерних наук та технологій
Кафедра кібербезпеки
Освітній ступінь магістр
Спеціальність 125 «Кібербезпека та захист інформації»
Освітньо-професійна програма «Безпека інформаційних і комунікаційних систем»

ЗАТВЕРДЖУЮ
Завідувач кафедри кібербезпеки

_____Анна ІЛЬЄНКО
«30»___08_____2024 р.

ЗАВДАННЯ
на виконання кваліфікаційної роботи

Дзеціни Вячеслава Анатолійовича

1. Тема кваліфікаційної роботи: Модуль кіберзахисту системи керування розумного будинку
затверджена наказом ректора від 30.08.2024 р. №1696/ст.
2. Термін виконання роботи: з 30.08.2024 по 15.12.2024
3. Вихідні дані до роботи: Проаналізувати існуючі інформаційні технології в інтелектуальному будинку та на основі результату проведеного аналізу визначити функціональні вимоги щодо модулю кіберзахисту системи керування розумного будинку. Розробити архітектуру модулю кіберзахисту в аспектах топології, формату та політик підключення. Розробити апаратну та програмну складові модуля кібербезпеки. Провести тестування розробленого модуля.
4. Зміст пояснювальної записки: Аналіз інформаційних технології в інтелектуальному будинку, побудова системи кіберзахисту при керування розумним будинком, визначення вимог та апаратна реалізація модулю кіберзахисту, програмна реалізація та тестування модулю кіберзахисту.
5. Перелік обов'язкового графічного (ілюстративного) матеріалу: презентація.

6. Календарний план-графік

№ з/п	Завдання	Термін виконання	Підпис керівника
1.	Провести аналіз літературних джерел.	30.08.2024 – 05.09.2024	*Виконано*
2.	Обґрунтувати вибір рішення.	06.09.2024 – 15.09.2024	*Виконано*
3.	Розробити апаратну частину модулю.	16.09.2024 – 25.09.2024	*Виконано*
4.	Розробити програмне забезпечення модулю кіберзахисту.	26.09.2024 – 02.10.2024	*Виконано*
5.	Протестувати роботу модулю кіберзахисту.	03.10.2024 – 25.10.2024	*Виконано*

6. Дата видачі завдання: «30» __08__ 2024 р.

Керівник кваліфікаційної роботи: _____ Олена ВИСОЦЬКА
(підпис керівника) (П.І.Б.)

Завдання прийняв до виконання: _____ Вячеслав ДЗЕЦІНА
(підпис здобувача вищої освіти) (П.І.Б.)

РЕФЕРАТ

Пояснювальна записка до кваліфікаційної роботи «Модуль кіберзахисту системи керування розумного будинку»: 109 с., 33 рис., 6 табл., 29 літературних джерела.

Об’єкт дослідження: процес кіберзахисту інформаційної технології в розумному будинку.

Предмет дослідження: методи аудиту інформаційної технології в розумному будинку.

Мета кваліфікаційної роботи: розробити модуль кібербезпеки розумного будинку, який за рахунок забезпечення аудиту підвищує загальний рівень безпеки використання інформаційних технологій розумного будинку.

Методи дослідження: статистичний та багатофакторний аналіз, теорія вірогідності, комп’ютерне та натурне моделювання.

Практична цінність: розроблено модуль кібербезпеки розумного будинку, який рекомендується використовувати під час аудиту кібербезпеки розумного будинку, що за рахунок виявлень слабкости налаштувань шляхом перевірок дозволить покращити загальний рівень безпеки.

Наукова новизна: Запропонована модель аудиту, яка за рахунок контролю імовірність реалізації критичних вразливостей, дозволяє підвищити рівень безпеки використання інформаційних технологій в розумному будинку.

Результати кваліфікаційної роботи рекомендується використовувати при аудиту кібербезпеки розумного будинку що працює на платформі Home Assistans.
КІБЕРБЕЗПЕКА, АУДИТ КІБЕРБЕЗПЕКИ,

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ ТА СКОРОЧЕНЬ

DAC – Дискреційний контроль доступу

MAC – Мандатний контроль доступу

ПК – Персональний комп’ютер

МК – Модуль кібербезпеки

ВСТУП

Актуальність теми. Інформаційні технології навколо нас. Це твердження має багато аспектів. В першу чергу прогрес та радість нових можливостей. Це холодильник, який знає, що і де придбати тобі на вечерю. Але, це і зовнішнє спостережене за нашими діями, це і добре, як медик спостерігає за хворою людиною, і не зовсім, коли хтось бачить наші маленькі таємниці. Це можливість миттєвого переказу грошей за нашими рахунками і миттєва втрата накопичень, як внаслідок атаки хакера. Інформаційні технології - це тільки інструмент, корисне використане якого залежить тільки від навичок користувача. Якщо кібергігієні та кіберзахисту приділяється достатньо уваги, то вірогідність шкоди від інформаційних технологій мінімальна. Виконання функцій кіберзахисту теж можна доручити комп'ютеру. Але важливо знати, які функції і як він має виконувати саме в том середовищі, де він функціонує. Тому тема кваліфікаційної роботи «Модуль кіберзахисту системи керування розумного будинку» є актуальною.

Метою кваліфікаційної роботи є розробка модуля кібербезпеки розумного будинку, який за рахунок забезпечення аудиту, підвищує загальний рівень безпеки використання інформаційних технологій розумного будинку.

Для досягнення поставленої мети необхідно розв'язати наступні задачі:

1. Проаналізувати існуючиі інформаційні технології в інтелектуальному будинку, та на основі результату проведеного аналізу визначити функціональні вимоги щодо модулю кіберзахисту системи керування розумного будинку.

2. Розробити архітектуру модулю кіберзахисту в аспектах топології, формату та політик підключення.

3. Розробити апаратну та програмну складові модуля кіберзахисту для реализації визначених на етапі аналізу функціональних вимог щодо аудиту налаштування профілів механізмів захисту розумного будинку.

4. Провести тестування розробленого модуля кібербезпеки розумного будинку, що дасть змогу дослідити коректність його роботи.

Об'єкт та предмет дослідження:

Об’єкт дослідження: процес кіберзахисту інформаційної технології в розумному будинку.

Предмет дослідження: методи аудиту інформаційної технології в розумному будинку.

Методи дослідження. Статистичний та багатофакторний аналіз, теорія вірогідності, комп’ютерне та натурне моделювання.

Наукова новизна отриманих результатів. Запропонована модель аудиту, яка за рахунок контролю імовірності реалізації критичних вразливостей дозволяє підвищити рівень безпеки використання інформаційних технологій в розумному будинку.

Практичне значення отриманих результатів. Розроблено модуль кібербезпеки розумного будинку, який рекомендується використовувати під час аудиту кібербезпеки розумного будинку, що, за рахунок виявлень слабкості налаштувань шляхом перевірок, дозволить покращити загальний рівень безпеки.

Особистий внесок здобувача вищої освіти. Результати кваліфікаційної роботи отримані автором особисто.

Апробація отриманих результатів. Основні положення роботи доповідалися та обговорювалися на міжнародній науково-практичній конференції: Innovations and New Directions in Scientific Research: Proceedings of the International Scientific Conference (2024, October 14). Manchester, UK: Bookmundo.

Публікації. Vjatcheslav Dzecina, Olena Vysotska. Inspection of critical properties of subsystems of an smart home. Innovations and New Directions in Scientific Research: Proceedings of the International Scientific Conference (2024, October 14). Manchester, UK: Bookmundo...p.p. 111-112.

В першому розділі проведено аналіз інформаційних технологій, які застосуються в розумному будинку. А саме визначені базові поняття інформаційних технології в інтелектуальному будинку. Виконано аналіз ключових виробників систем розумних будинків. Розглянуто основні напрями

автоматизації розумного будинку. Визначено функціональні вимоги щодо модулю кіберзахисту системи керування розумного будинку.

В другому розділі проведено аналіз і побудована архітектура топології, формату та політик підключення модулю кіберзахисту. А саме, здійснено вибір технології та топології підключення модулю кіберзахисту. Виконано вибір варіанту підключення модулю кіберзахисту. Зроблено аналіз архітектури безпеки при керуванні розумним будинком - в другому розділі.

В третьому розділі, визначено вимоги та апаратна реалізація модулю кіберзахисту, аналіз вимог щодо модулю кіберзахисту. Зроблено аналіз атак на розумній будинок. Здійснена реалізація апаратної частині модулю кіберзахисту.

В четвертому розділі, здійснена програмна реалізація та тестування модулю кіберзахисту. А саме, обрано середовище реалізації. Виконана програмна реалізація модуля. Протестовано функції перехоплення трафіку.

В цілому в роботі :

- проаналізовані існуючі інформаційні технології в інтелектуальному будинку та, на основі результату проведеного аналізу, визначено функціональні вимоги щодо модулю кіберзахисту системи керування розумного будинку.
- розроблено архітектуру модулю кіберзахисту в аспектах топології, формату та політик підключення.
- розроблено апаратну та програмну складові модуля кібербезпеки.
- проведено тестування розробленого модуля кібербезпеки розумного будинку, що дало змогу довести коректність його роботи.

РОЗДІЛ 1
АНАЛІЗ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЇ В ІНТЕЛЕКТУАЛЬНОМУ БУДИНКУ.

1.1 Базові поняття інформаційних технології в інтелектуальному будинку

Інтелектуальний будинок – це система домашніх пристроїв, здатних виконувати дії та вирішувати певні повсякденні завдання без участі людини. Основна ідея полягає в автоматизації та спрощенні життя мешканців.

Беззаперечна перевага інтелектуального будинку полягає у тому, що його можливості обмежуються виключно Вашою фантазією

Ось декілька варіантів використання підсистем керування:
1. Керування освітленням:

Автоматичне вмикання/вимикання світла

Регулювання яскравості

Створення сценаріїв освітлення

Взаємодія з датчиками руху та освітленості

Змоделюємо декілька варіантів використання, почнемо з першого пункту. Мабуть у всіх бувало таке, що забували вимкнути світло коли йдете по справах, з системою розумного будинку можливе автоматичне вимикання / вмикання світла при виході / вході в квартиру чи будинок. Завдяки тонким можливостями налаштування, можна досягти регулювання яскравості, яке буде повністю відповідати Вашим вимогам. Створення сценаріїв освітлення дозволить, наприклад, робити світло теплим коли сідаєте читати книгу або приглушення світла коли дивитеся телевізор. Завдяки великій кількості датчиків руху чи освітленості можна налаштувати ввімкнення / вимкнення світла при настанні темряви чи відповідно — ранку та зміни яскравості для отримання стабільного освітлення. А завдяки датчикам руху — можливо взагалі

відмовитися від вимикачів світла, воно просто буде вмикатися коли ви пройдете повз чи відкриєте двері і вимикатися через деякий час або відсутності руху.

2. Керування кліматом:

-Управління термостатами

-Контроль кондиціонерів та опалення

-Створення енергозберігаючих сценаріїв

Регулювання температури - ви можете використовувати систему розумного будинку, щоб регулювати температуру у вашому будинку, щоб економити енергію та забезпечити комфорт.

Створення графіка - ви можете встановити графік для вашого термостата, щоб він автоматично змінював температуру протягом дня.

Віддалений контроль - ви можете використовувати смартфон, щоб змінювати температуру у вашому будинку, коли вас немає.

Вмикання/вимикання кондиціонера - ви можете використовувати голосові команди або смартфон, щоб вмикати та вимикати кондиціонер.

Голосове регулювання температури - ви можете використовувати голосові команди або смартфон, щоб регулювати температуру кондиціонера.

Економія енергії - ви можете використовувати систему розумного будинку, щоб економити енергію, автоматично вимикаючи кондиціонер або опалення, коли вас немає.

Сценарій "Економія енергії" - вимкнення кондиціонера або опалення, коли вас немає, зниження температури вночі, автоматичне закриття штор.

Сценарій "Відпочинок" - встановлення комфортної температури, приглушення світла, включення фонового шуму.

Сценарій "Вечірка" - вимкнення кондиціонера, відкриття вікон, включення вентиляторів.

3. Керування безпекою:

-Охоронна сигналізація

-Відеонагляд

-Контроль доступу

-Пожежна сигналізація

Оповіщення про проникнення - ви можете отримувати повідомлення про проникнення у ваш будинок, коли вас немає.

Віддалений контроль - ви можете використовувати смартфон, щоб увімкнути або вимкнути охоронну сигналізацію.

Підключення до служби охорони - ви можете підключити вашу систему охоронної сигналізації до служби охорони, щоб вони могли негайно відреагувати на проникнення.

Спостереження за вашим будинком - ви можете використовувати камери відеоспостереження, щоб спостерігати за вашим будинком, коли вас немає.

Віддалений перегляд - ви можете використовувати смартфон, щоб переглядати камери відеоспостереження в будь-який час.

Запис подій - ви можете використовувати камери відеоспостереження для запису подій, які відбуваються у вашому будинку.

Віддалене відкривання дверей - ви можете використовувати смартфон, щоб віддалено відкривати двері для гостей або кур'єра.

Блокування дверей - ви можете використовувати смартфон, щоб заблокувати двері, коли вас немає.

Журнал доступу - ви можете отримувати повідомлення про те, хто і коли приходив до вашого будинку.

Оповіщення про пожежу - ви можете отримувати повідомлення про пожежу у вашому будинку, коли вас немає.

Віддалений контроль - ви можете використовувати смартфон, щоб увімкнути або вимкнути пожежну сигналізацію.

Підключення до служби порятунку - ви можете підключити вашу систему пожежної сигналізації до служби порятунку, щоб вони могли негайно відреагувати на пожежу.

4. Керування енергоспоживанням:

-Моніторинг енергоспоживання

-Вимкнення приладів у режимі очікування

-Оптимізація використання енергії

Відстеження споживання енергії - ви можете використовувати систему розумного будинку, щоб відстежувати споживання енергії вашим будинком або окремими приладами.

Отримання звітів - ви можете отримувати звіти про ваше енергоспоживання, щоб бачити, де ви можете економити.

Порівняння з іншими будинками - ви можете порівнювати ваше енергоспоживання з енергоспоживанням інших будинків.

Автоматичне вимкнення - ви можете використовувати систему розумного будинку, щоб автоматично вимикати прилади, коли вони не використовуються.

Вимкнення одним натисканням - ви можете використовувати смартфон, щоб вимкнути всі прилади в вашому будинку одним натисканням кнопки.

Економія енергії - ви можете економити енергію, вимикаючи прилади в режимі очікування.

Регулювання температури - ви можете використовувати систему розумного будинку, щоб регулювати температуру у вашому будинку, щоб економити енергію.

Використання енергоефективних приладів - ви можете використовувати систему розумного будинку, щоб отримувати рекомендації щодо енергоефективних приладів.

Використання відновлюваних джерел енергії - ви можете використовувати систему розумного будинку, щоб інтегрувати сонячні панелі або інші відновлювані джерела енергії у вашу систему.

5. Керування мультимедіа:

-Управління телевізором, музичною системою

-Доступ до контенту

-Створення мультимедійних сценаріїв

Вмикання/вимикання телевізора - ви можете використовувати голосові команди або пульт дистанційного керування, щоб вмикати та вимикати телевізор.

Перемикання каналів - ви можете використовувати голосові команди або пульт дистанційного керування, щоб перемикати канали.

Регулювання гучності - ви можете використовувати голосові команди або пульт дистанційного керування, щоб регулювати гучність.

Відтворення музики - ви можете використовувати голосові команди або смартфон, щоб відтворювати музику з вашої бібліотеки або онлайн-сервісів.

Створення плейлистів - ви можете використовувати смартфон, щоб створювати плейлисти з вашої улюбленої музики.

Перегляд фільмів і серіалів - ви можете використовувати онлайн-сервіси, такі як Netflix, HBO Max або Amazon Prime Video, щоб переглядати фільми та серіали.

Прослуховування подкастів - ви можете використовувати онлайн-сервіси, такі як Spotify або Apple Podcasts, щоб прослуховувати подкасти.

Перегляд фотографій - ви можете використовувати смартфон або планшет, щоб переглядати фотографії, які зберігаються на вашому домашньому сервері.

Сценарій "Вечір" - приглушення світла, увімкнення телевізора, запуск вашого улюбленого плейлиста.

Сценарій "Вечірка" - вимкнення освітлення, увімкнення музики, запуск світлового шоу.

Сценарій "Відпочинок" - увімкнення фонового шуму, наприклад, звуку дощу або шуму лісу, приглушення світла.

6. Керування домашніми приладами:

-Управління пральною машиною, посудомийною машиною, холодильником

-Автоматизація домашніх завдань

-Створення сценаріїв для роботи приладів

Автоматичний запуск прання - ви можете встановити час початку прання, щоб воно закінчилося до вашого повернення додому.

Моніторинг ходу прання - ви можете отримувати повідомлення про стан прання, наприклад, коли воно закінчиться.

Автоматичний запуск миття - ви можете встановити час початку миття посуду, щоб воно закінчилося до вашого повернення додому.

Контроль температури - ви можете дистанційно регулювати температуру в холодильнику.

Моніторинг продуктів - ви можете отримувати повідомлення про те, коли закінчуються продукти харчування.

Створення рецептів - ви можете використовувати рецепти, які пропонуються системою розумного будинку, з урахуванням продуктів, які є у вас в холодильнику.

Підготовка вечері - ви можете запрограмувати мультиварку, щоб вона почала готувати вечерю до вашого повернення додому.

Прибирання - ви можете використовувати робот-пилосос для автоматичного прибирання вашого будинку.

Полив рослин - ви можете встановити систему автоматичного поливу, щоб ваші рослини завжди отримували необхідну кількість води. А ось декілька сценаріїв для роботи приладів.

Сценарій "Ранок" - вмикання світла, кавоварки, підігрів рушників у ванній кімнаті.

Сценарій "Вечір" - приглушення світла, увімкнення телевізора, активація охоронної сигналізації.

Сценарій "Відпустка" - **в**имикання всіх приладів, крім холодильника, активація режиму економії енергії.

7. Моніторинг довкілля:

-Контроль якості повітря

-Моніторинг температури та вологості

-Отримання інформації про погодні умови

Вимірювання рівня забруднення - ви можете використовувати датчики для вимірювання рівня забруднення повітря в вашому будинку або на вулиці.

Отримання повідомлень про шкідливі речовини - ви можете отримувати повідомлення про те, коли рівень забруднення повітря перевищує допустимі норми.

Очищення повітря - ви можете використовувати очищувач повітря, щоб поліпшити якість повітря у вашому будинку.

Вимірювання температури та вологості - ви можете використовувати датчики для вимірювання температури та вологості у вашому будинку.

Створення комфортних умов - ви можете використовувати систему кондиціювання або опалення, щоб створити комфортні умови у вашому будинку.

Запобігання плісняві - ви можете використовувати осушувач повітря, щоб запобігти появі плісняви у вашому будинку.

Прогноз погоди - ви можете отримувати прогноз погоди на день, тиждень або місяць.

Попередження про негоду - ви можете отримувати попередження про сильні вітри, зливи, град або інші небезпечні погодні явища.

Планування вашого дня - ви можете використовувати інформацію про погоду, щоб спланувати свій день або тиждень.

8. Зв'язок та комунікація:

-Внутрішній домофон

-Відеозв'язок з відвідувачами

-Оповіщення та повідомлення

Зв'язок з членами сім'ї - ви можете використовувати домофон, щоб зв'язатися з членами сім'ї, які знаходяться в інших кімнатах вашого будинку.

Відповідь на дверний дзвінок - ви можете використовувати домофон, щоб відповісти на дверний дзвінок і побачити, хто прийшов.

Віддалений доступ - ви можете використовувати домофон, щоб віддалено відкрити двері для гостей або кур'єра.

Спілкування з відвідувачами - ви можете спілкуватися з відвідувачами, які знаходяться біля вашого будинку, не виходячи з нього.

Віддалена ідентифікація - ви можете використовувати відеозв'язок, щоб ідентифікувати відвідувачів.

Запис розмови - ви можете записати розмову з відвідувачами для вашої безпеки.

Нагадування - ви можете встановити нагадування про важливі події або завдання.

Повідомлення про події - ви можете отримувати повідомлення про події, які відбуваються у вашому будинку, наприклад, про спрацювання охоронної сигналізації.

Спілкування з друзями та близькими - ви можете використовувати систему розумного будинку для спілкування з друзями та близькими, які не знаходяться у вашому будинку.

9. Додаткові підсистеми:

-Розумні полиці

-Роботи-пилососи

-Системи поливу

-Системи розумного саду

Контроль за терміном придатності продуктів - ви можете використовувати датчики, щоб відстежувати термін придатності продуктів харчування на ваших полицях.

Створення списку покупок - ви можете отримувати повідомлення про те, які продукти харчування вам потрібно купити.

Рекомендації рецептів - ви можете отримувати рекомендації рецептів на основі продуктів, які є у вас на полицях.

Автоматичне прибирання - ви можете використовувати робот-пилосос для автоматичного прибирання вашого будинку.

Створення графіка прибирання - ви можете встановити графік прибирання для вашого робота-пилососа.

Віддалене керування - ви можете використовувати смартфон, щоб керувати роботом-пилососом.

Автоматичний полив - ви можете використовувати систему автоматичного поливу, щоб ваші рослини завжди отримували необхідну кількість води.

Моніторинг вологості ґрунту - ви можете використовувати датчики, щоб відстежувати вологість ґрунту у вашому саду.

Економія води - ви можете використовувати систему автоматичного поливу, щоб економити воду.

Контроль за освітленням - ви можете використовувати систему розумного саду, щоб регулювати освітлення ваших рослин.

Контроль за температурою - ви можете використовувати систему розумного саду, щоб регулювати температуру у вашому саду.

Захист від шкідників - ви можете використовувати систему розумного саду, щоб захистити ваші рослини від шкідників.

1.2 Аналіз ключових виробників систем розумних будинків

На ринку існує багато виробників розумного будинку, які пропонують різноманітні системи та пристрої для автоматизації житла. Ось декілька з них:

1) Ajax Systems [1] - це українська компанія, заснована в 2011 році, яка розробляє та виробляє бездротові системи безпеки та розумного будинку. Їхні продукти використовуються в житлових, комерційних та промислових приміщеннях по всьому світу.

Ajax пропонує широкий спектр систем розумного будинку, які можна розділити на дві основні категорії:

- Системи безпеки - ці системи призначені для захисту вашого будинку від крадіжок, пожеж та інших загроз. Вони включають в себе датчики руху, відкриття дверей/вікон, датчики затоплення, пожежні датчики та багато іншого.

- Системи автоматизації - ці системи призначені для автоматизації завдань у вашому домі, таких як увімкнення/вимкнення освітлення, регулювання температури, керування розумними пристроями тощо. Вони включають в себе розумні розетки, термостати, реле, кнопки та багато іншого.

Переваги систем Ajax:

1. Бездротові - системи Ajax є бездротовими, тому вам не потрібно прокладати кабелі по вашому дому.

2. Прості у встановленні - системи Ajax прості у встановленні та налаштуванні. Ви можете зробити це самостійно за лічені хвилини.

3. Надійні - системи Ajax надійні та безпечні. Вони використовують передові технології для захисту вашого дому.

4. Масштабовані - системи Ajax можна масштабувати, щоб відповідати вашим потребам. Ви можете додати більше датчиків, пристроїв та користувачів у будь-який час.

5. Сучасні - системи Ajax мають сучасний дизайн та зручний інтерфейс.

Деякі з найпопулярніших систем Ajax:

Ajax Hub - це центральний елемент системи Ajax. Він керує всіма датчиками та пристроями Ajax.

Ajax KeyPad - це клавіатура, яка використовується для керування системою Ajax.

Ajax SpaceControl - це пульт дистанційного керування, який використовується для керування системою Ajax.

Ajax MotionProtect - це датчик руху, який використовується для виявлення вторгнення.

Ajax DoorProtect - це датчик відкриття дверей/вікон, який використовується для виявлення проникнення.

Ajax FireProtect - це пожежний датчик, який використовується для виявлення пожежі.

2) BroadLink - це китайська компанія [2], заснована в 2009 році, яка розробляє та виробляє продукти розумного дому. Їхні продукти використовуються в житлових та комерційних приміщеннях по всьому світу.

BroadLink пропонує широкий спектр систем розумного будинку, які можна розділити на дві основні категорії:

- Універсальні пульти дистанційного керування - ці пульти дистанційного керування можуть використовуватися для керування широким спектром пристроїв, таких як телевізори, кондиціонери, вентилятори, лампи тощо.

- Системи автоматизації - ці системи призначені для автоматизації завдань у вашому домі, таких як увімкнення/вимкнення освітлення, регулювання температури, керування розумними пристроями тощо. Вони включають в себе розумні розетки, термостати, датчики та багато іншого.

Переваги систем BroadLink:

1. Доступні - системи BroadLink є одними з найдоступніших на ринку.

2. Прості у використанні - системи BroadLink прості у налаштуванні та використанні.

3. Сумісні - системи BroadLink сумісні з широким спектром пристроїв.

4. Надійні - системи BroadLink надійні та безпечні.

5. Масштабовані - системи BroadLink можна масштабувати, щоб відповідати вашим потребам.

Деякі з найпопулярніших систем BroadLink:

BroadLink RM Pro - це універсальний пульт дистанційного керування, який може використовуватися для керування широким спектром пристроїв.

BroadLink RM Mini3 - це компактний універсальний пульт дистанційного керування, який може використовуватися для керування телевізорами, кондиціонерами та іншими пристроями.

BroadLink S1C - це розумна розетка, яка використовується для керування електроприладами.

BroadLink HTS2 - це термостат, який використовується для регулювання температури у вашому домі.

BroadLink SP3S - це розумний датчик, який використовується для виявлення руху, температури та вологості.

3) Fibaro - це польська компанія, заснована в 2008 році, яка розробляє та виробляє продукти розумного дому. Їхні продукти використовуються в житлових та комерційних приміщеннях по всьому світу [3].

Fibaro пропонує широкий спектр систем розумного будинку, які можна розділити на дві основні категорії:

- Z-Wave - ці продукти використовують Z-Wave, бездротовий протокол, який спеціально розроблений для автоматизації дому.

- Zigbee - ці продукти використовують Zigbee, бездротовий протокол, який також використовується для автоматизації дому.

Переваги систем Fibaro:

1. Інноваційні - Fibaro постійно вдосконалює свої продукти та пропонує інноваційні рішення для розумного дому.

2. Якість - продукти Fibaro виготовлені з високоякісних матеріалів і мають сучасний дизайн.

3. Безпека - системи Fibaro безпечні та надійні.

4. Простота використання - системи Fibaro прості у налаштуванні та використанні.

5. Сумісність - системи Fibaro сумісні з широким спектром пристроїв.

Деякі з найпопулярніших систем Fibaro:

Fibaro Home Center 3 - це центральний елемент системи Fibaro. Він керує всіма датчиками, пристроями та сценаріями Fibaro.

Fibaro Dimmer 2 - це розумний димер, який використовується для регулювання освітлення.

Fibaro Roller Shutter 3 - це розумний ролет, який використовується для керування жалюзі та шторами.

Fibaro Motion Sensor - це датчик руху, який використовується для виявлення вторгнення.

Fibaro Flood Sensor - це датчик затоплення, який використовується для виявлення протікання води.

4) Orvibo - це китайська компанія, заснована в 2011 році, яка розробляє та виробляє продукти розумного дому. Їхні продукти використовуються в житлових та комерційних приміщеннях по всьому світу [4].

Orvibo пропонує широкий спектр систем розумного будинку, які можна розділити на дві основні категорії:

- Wi-Fi - ці продукти підключаються до вашої домашньої мережі Wi-Fi.

Zigbee - ці продукти використовують Zigbee, бездротовий протокол, який спеціально розроблений для автоматизації дому.

Переваги систем Orvibo:

1. Доступні - системи Orvibo є одними з найдоступніших на ринку.

2. Прості у використанні - системи Orvibo прості у налаштуванні та використанні.

3. Сумісні - системи Orvibo сумісні з широким спектром пристроїв.

4. Надійні - системи Orvibo надійні та безпечні.

5. Масштабовані - системи Orvibo можна масштабувати, щоб відповідати вашим потребам.

Деякі з найпопулярніших систем Orvibo:

Orvibo AllOne - це універсальний пульт дистанційного керування, який може використовуватися для керування широким спектром пристроїв.

Orvibo Home Mate - це мобільний додаток, який використовується для керування системами Orvibo.

Orvibo Smart Dimmer - це розумний димер, який використовується для регулювання освітлення.

Orvibo Smart Plug - це розумна розетка, яка використовується для керування електроприладами.

Orvibo Motion Sensor - це датчик руху, який використовується для виявлення вторгнення.

5) Xiaomi - це китайська компанія [5], заснована в 2010 році, яка розробляє та виробляє широкий спектр електроніки, включаючи смартфони, планшети, ноутбуки, телевізори, роутери та багато іншого. Їхні продукти розумного будинку є одними з найпопулярніших у світі.

Xiaomi пропонує широкий спектр систем розумного будинку, які можна розділити на дві основні категорії:

- Mi Home - ця платформа використовується для керування всіма пристроями Xiaomi розумного будинку.

- Aqara - це суббренд Xiaomi, який пропонує більш широкий спектр продуктів розумного будинку, включаючи датчики, розумні замки, камери та багато іншого.

Переваги систем Xiaomi:

1. Доступні - системи Xiaomi є одними з найдоступніших на ринку.

2. Прості у використанні - системи Xiaomi прості у налаштуванні та використанні.

3. Сумісні - системи Xiaomi сумісні з широким спектром пристроїв.

4. Надійні - системи Xiaomi надійні та безпечні.

5. Масштабовані - системи Xiaomi можна масштабувати, щоб відповідати вашим потребам.

Деякі з найпопулярніших систем Xiaomi:

Mi Home Hub - це центральний елемент системи Mi Home. Він керує всіма пристроями Xiaomi розумного будинку.

Aqara Hub - це центральний елемент системи Aqara. Він керує всіма пристроями Aqara розумного будинку.

Mi Smart LED Bulb - це розумна лампочка, яка використовується для регулювання освітлення.

Mi Smart Plug - це розумна розетка, яка використовується для керування електроприладами.

Aqara Motion Sensor - це датчик руху, який використовується для виявлення вторгнення.

1.3 Аналіз основних напрямів автоматизації розумного будинку

За останні 20-30 років автоматизовані системи управління [6] перестали бути модною екзотикою. Незалежно від застосування, будь то будівля, складальний цех або поїзд метро, метою впровадження таких систем є зниження

експлуатаційних витрат, надання важливої інформації, підвищення безпеки та комфорту. Але, незважаючи на те, що модні журналісти зараз більше цікавляться досягненнями традиційних IT-компаній, досягнення в області автоматичного управління в найближчому майбутньому можуть мати не менше впливу на наше світогляд, ніж поява мобільних телефонів і Інтернету.

Щоб зрозуміти, наскільки змінилися можливості в галузі автоматизації за останні роки і як вони будуть змінюватися, важливо зрозуміти значення деяких технологічних проривів, які відбулися за останні роки.

Власність окремих компаній, відповідні продукти та технології автоматизації було важко інтегрувати один з одним. Для вирішення цієї проблеми були потрібні дорогі технічні рішення, пов’язані з написанням нового програмного забезпечення, зміною топології мережі та придбанням додаткових компонентів.

Таким чином, на певному етапі ринку склалися об'єктивні передумови для успішного впровадження нових підходів у сфері автоматизації.

1.3.1 Напрями автоматизації систем забезпечення послуг

У системі захисту можуть бути використані кілька типів датчиків. Ці датчики можуть бути:

- датчики руху;
- датчики чадного газу;
- датчики розбиття скла;
- датчики відкритих дверей та вікон;
- датчики диму [7]

Слід звернути увагу на спосіб встановлення цих детекторів і датчиків і місце їх розташування в будинку. Наприклад, чи мають бути датчики диму на кожному поверсі, чи мають бути датчики руху біля кожних дверей, чи достатньо використовувати датчики відкритих дверей?

Потім потрібно вибрати спосіб сповіщення при спрацьовуванні будь-якого з датчиків. Чи достатньо буде лише сповістити компанію, яка контролює будинок, коли виникає тривога.

Плануючи систему освітлення розумного будинку, слід пам’ятати, що існує два способи встановлення та налаштування освітлювальних приладів. Перший полягає в установці нової виділеної проводки для системи керування освітленням, а другий – у використанні існуючої проводки (за допомогою, наприклад, протоколу X10). Використання спеціальної проводки забезпечить більш надійний результат, але обійдеться вдвічі дорожче. XI0 дозволяє легко встановлювати та налаштовувати пристрої та координувати їх з іншими пристроями у системі розумного дому.

Можна сказати, що схема освітлення певною мірою добре поєднується з системою захисту. Це не означає, що коли в будинок зайдуть непрохані гості, яскравість світла відразу зменшиться, щоб забезпечити їм більший комфорт. Але це означає, що певні освітлювальні прилади пов’язані з датчиками руху і вмикаються, коли ці датчики спрацьовують. Наприклад, якимось популярним освітлювальним приладом можна підсвічувати підходи до будинку, які при виявленні руху будуть наповнюватися світлом. Його не обов'язково підключати до системи захисту (інакше поліція буде приходити кожного разу, коли сусідський кіт буде гуляти опівночі). У цьому випадку заходи захисту будуть зведені до розпізнавання будь-якого руху та ввімкнення світла. Це також може стати в нагоді, коли вам потрібно вийти з машини вночі, і трохи світла не завадить, або коли ви хочете перевірити, чи хтось ходить по дому.

1.3.2 Структурні особливості системи

Основою справді інтегрованої системи розумного будинку [8] є локальна мережа (LAN). Ця локальна мережа потрібна не лише для керування пристроями розумного дому – вона також стане в нагоді під час виконання простих обчислень. Майже в кожному домі є комп’ютер (це твердження справедливо для більшості

людей, особливо для тих, у кого вдома повно проводів і хто хоче побудувати «розумний будинок»). Крім того, якщо в будинку кілька комп'ютерів, то напевно є необхідність їх підключення. Домашня локальна мережа має багато переваг, від спільного доступу до Інтернету до файлів і принтерів. Ви можете не тільки спільно використовувати ті самі ресурси, але й використовувати комп'ютер для виконання спеціальних завдань, наприклад, для контролю роботи системи безпеки або обслуговування музичного автомата. Крім того, якщо проект розумного будинку досить масштабний, то, начебто, має сенс придбати комп’ютер або спеціальний пристрій, призначений виключно для управління обладнанням.

Звичайно, не обов'язково встановлювати локальну мережу - будь-яку кількість функцій можна реалізувати без наявності локальної мережі. Однак, якщо ви хочете досягти високого рівня автоматизації цих функцій і їх інтеграції, локальна мережа буде відмінним помічником.

З будь-якою домашньою локальною мережею вам доведеться працювати з багатьма її компонентами, наприклад мережевими клієнтами. Будь-який пристрій, який використовується для введення або виведення даних, тобто це пристрої, які підключаються до мережі і отримують інформацію або надають її для подальшої обробки. Це такі пристрої, як клієнтські комп’ютери та принтери.

Рис. 1.1. Домашня локальна мережа

Щоб почати обговорення, розглянемо схему, показану на рисунку 1.1. Це схема простої домашньої локальної мережі. Комп’ютер, який найчастіше використовується, є клієнтським пристроєм. Домашня локальна мережа, показана на рисунку 1.1, містить три клієнтські комп’ютери: один у домашньому офісі, один у кімнаті відпочинку, а третій є ноутбуком.

Це комп’ютери, якими члени родини користуються під час обчислень. Наприклад, їх можна використовувати для виконання домашнього завдання, доступу до Інтернету та вирішення обчислювальних завдань. Як правило, комп’ютери використовують Windows як операційну систему.

У цьому прикладі два настільні комп’ютери підключено до мережі за допомогою кабелю категорії 5e, а ноутбук під’єднано за допомогою бездротового з’єднання, але насправді дозволено будь-яке поєднання. Бездротову мережеву карту можна легко використовувати на всіх трьох комп’ютерах.

Сервери - це спеціальні комп'ютери, які призначені для обслуговування клієнтських комп'ютерів. Сервери забезпечують зв'язок клієнтських комп'ютерів між собою та з Інтернетом. У великих компаніях нерідко можна зустріти ситуацію, коли кілька серверів використовуються для управління сотнями або навіть тисячами клієнтських комп'ютерів. Однак для системи, що розробляється, потрібен лише один сервер. Крім того, виправданим буде використання програмного забезпечення розумного будинку на одному комп’ютері, який виконує подвійну роль, а саме роль і клієнтської машини, і сервера.

1.3.3 Протоколи передачі для автоматизації будівель

На сьогоднішній день існує більше трьохсот різних протоколів [9] передачі даних в системах автоматизації. Всі вони повинні відповідати певним вимогам.

У системах автоматизації помилка в даних, що передаються від контролера або до нього, означає вихід з ладу виконавчого механізму. Ціна такої помилки може бути дуже високою. Тому основними вимогами до протоколу передачі даних є надійність протоколу, його стійкість до помилок і можливих розривів

лінії. У таблиці 1.1 наведені можливі варіанти інженерного вирішення даної проблеми.

Системи контролю та управління будівлею зазнають розширення кілька разів протягом свого життєвого циклу. Як правило, якщо підприємство розробляє нові продукти або розширює виробництво, існуючі датчики або замінюють, або доповнюють більш точними. У той же час, розтягуючи лінії зв'язку до нових контролерів або інтелектуальних адресних датчиків, часто доводиться мати справу з жорсткими вимогами до топології використовуваного протоколу. Тому в даному випадку ідеальним буде протокол, який має мінімальні вимоги до топології ліній. Такий протокол зазвичай називають протоколом вільної топології.

Таблиця 1.1

Рішення, що використовуються для підвищення стійкості протоколів передачі даних до помилок

Проблему вирішує система передачі даних	Рішення, що застосовується
Надійний обмін повідомленнями, перевірка цілісності	Надійність протоколу, його стійкість до помилок і можливих розривів вузлів
Захист від відмови	Резервування через дублювання вузлів, ліній, мереж. Кільцева топологія, яка дозволяє підтримувати зв'язок під час локалізованого розриву
Ізоляція несправності та відновлення.	Автоматична ідентифікація несправного вузла. Дистанційне керування за допомогою дистанційних команд процесу ізоляції та відключення вийшли з ладу вузлів.

Основні параметри протоколу, доступні зараз і в майбутньому:

- рішення на основі CAN, такі як автоматизація CAN;
- DeviceNet, J1850 і SDS;
- шини простих датчиків Seriplex і Bitbus;
- технологія LonWorks;
- CEBus;
- BACnet;
- промисловий автобус EtherCat;
- TCP/IP з бездротовими мережами [10].

Є й інші схеми, призначені для вирішення конкретних завдань. Компанії, які розробляли протоколи, не мали наміру продавати їх третім організаціям, а планували використовувати у своїй роботі. Таблиці 1.2 і 1.3 підсумовують деякі характеристики вищезазначених протоколів.

Промисловий автобус EtherCAT розроблений німецькою компанією Beckhoff. EtherCAT — це рішення для автоматизації Ethernet, яке є одночасно високопродуктивним і простим у використанні. За допомогою EtherCAT ви можете завершити топологію Ethernet типу «зірка» за допомогою простої лінійної структури.

Таблиця 1.2

Основні характеристики протоколів передачі BACnet, CAN-based, CEBus

Характеристики	BACnet	CAN-based (SDS, DeviceNet)	CEBus
Галузь застосування	Автоматизація будівель	Транспорт (J1850, J1939) Дискретна автоматизація (SDS, DeviceNet)	

Закінчення таблиці 1.2

Характеристики	BACnet	CAN-based (SDS, DeviceNet)	CEBus
Рівні OSI/ISO	1,2,3,7		1,2,3,7
Підтримувані середовища передачі		Віта пара (SDS, DeviceNet) Альтернативні рішення на основі оптоволокна для мереж CAN	Силові електричні лінії (FCC) Коаксіальний кабель RF
Швидкість передачі даних	10 Mbps	1 Mbps (CAN) 1 Mbps (SDS) 500 Kbps (DeviceNet)	6.666 kbps (або 10 kbps)
Максимальний адресний простір	248		216
Підтримка маршрутизаторів мережного рівня	Є	Ні	Ні
Автентифікація	Є	Ні	На рівні програми

Основи передачі інформації за протоколами TCP/IP.

TCP/IP є двома основними мережевими протоколами Інтернету. Часто ця назва також використовується для позначення мереж, що працюють на їх основі. Протокол IP забезпечує маршрутизацію (доставку за адресою) мережевих пакетів. Протокол TCP забезпечує встановлення надійного з'єднання між двома машинами та фактичну передачу даних шляхом контролю оптимального розміру пакета даних, що передається, і повторної відправки в разі збою. Кількість одночасно встановлених з'єднань між абонентами мережі не обмежена, тобто будь-яка машина може обмінюватися даними з будь-якою кількістю інших машин по одній фізичній лінії за певний проміжок часу.

Таблиця 1.3

Основні характеристики протоколів передачі EtherCat,

Характеристики	EtherCat	LONWORKS
Галузь застосування	Автоматизація будівлі	Автоматизація будівель Керування виробництвом Автоматизація фабрик Транспорт Автоматизація житла, Автоматизація підстанцій, Управління вуличним освітленням
Рівні OSI/ISO	1,2,3,4,5,6,7	1,2,3,4,5,6,7
Підтримувані середовища передачі	Кручена пара. Оптоволокно	Віта пара з вільною топологією Кручена пара Лінії електроживлення (рішення, сумісне зі стандартами FCC та CENELEC) Оптоволокно Коаксіальний кабель RF (кілька діапазонів)
Швидкість передачі даних	10 Mbps	5Mbps
Максимальний адресний простір	65535 вузлів у мережі	248 доменів, 32000 вузлів у домені
Підтримка маршрутизаторів мережевого рівня	Є	Є
Аутентифікація	Є	Є

Ще однією важливою перевагою мережі з протоколами TCP/IP є те, що через неї можна підключати машини з різними архітектурами та різними операційними системами, такими як Unix, MacOS, MS-DOS, MS Windows тощо. Більше того, машини однієї системи за допомогою мережевої файлової системи NFS (NetFileSystem) можуть підключати диски з файловою системою абсолютно іншої собі ОС і оперувати «чужими» файлами як своїми.

Протоколи TCP/IP (TransmissionControlProtocol/InternetProtocol) є основними транспортними та мережевими протоколами в ОС UNIX. Заголовок TCP/IP пакета вказує:

1. IP-адреса відправника.
2. IP-адреса одержувача.
3. Номер порту.

Пакети TCP / IP мають унікальну властивість потрапляти до адресата, проходячи через різнорідні, в тому числі локальні мережі, з використанням різноманітних фізичних носіїв. Маршрутизація IP-пакету (передача його в потрібну мережу) здійснюється на добровільній основі комп’ютерами, що входять до складу мережі TCP/IP.

Протокол IP – це протокол, який описує формат пакета даних, що передається через мережу.

Створення системи домашньої автоматизації передбачає інтеграцію даного проекту з основними елементами систем домашнього зв'язку, опалення, водопостачання, медіа-пристроїв і т. д. Створена система має кілька варіантів реалізації, що розширює сферу її застосування.

Принцип побудови системи «розумний дім» з бездротовим маршрутизатором або модемом наведено на рис. 1.2 та рис.1.3

Рис. 1.2 Принцип побудови системи «розумний дім» з бездротовим маршрутизатором або модемом

Модулі «Розумний дім» можна використовувати як в будинках, підключених до мережі Інтернет, так і в приміщеннях без такої можливості [11]. У варіанті, зображеному на рисунку 1.2, в якості точки доступу виступає цілком звичайне домашнє обладнання таких компаній, як TP-link, D-link, Cisco, Zyxel, Asus та ін.

Переваги такого варіанту реалізації:

– можливість дистанційного керування;
– можливість дистанційного моніторингу;
– підключення веб-сервісів: погода, час тощо.

Нижче наведено другий варіант реалізації системи (рис. 1.3).

Рис. 1.3 Принцип побудови системи «розумний дім» із сервером у режимі програмної точки доступу

Ця опція створена без доступу до мережі, вся система має власну, незалежну мережу з шифруванням, тобто має повністю автономний, власний канал зв'язку. Ця опція захищена від несанкціонованих зовнішніх впливів і не залежить від ступеня завантаження мережі, як це буває в домашніх мережах, але позбавлена можливості віддаленої взаємодії.

Переваги:

– охорона;

– ширша сфера застосування;

– відсутність ризику, пов’язаного з перевантаженням мережі.

Технологія передачі ІЧ команд:

Як правило, пульти дистанційного керування використовують одну несучу частоту модуляції (тобто частоту випромінювання ІЧ-світлодіода) - на неї налаштовані і пульт, і приймач. Частоти модуляції зазвичай стандартні - це 36 кГц, 38 кГц, 40 кГц (Panasonic, Sony). Частоти 56 кГц (Sharp) вважаються рідкісними. Фірма Використання приймача з частотою модуляції, яка не зовсім

відповідає частоті передавача, не означає, що він не буде приймати - прийом залишиться, але його чутливість може сильно впасти [12].

Передача сигналу здійснюється випромінюванням ІЧ світлодіоду з відповідною частотою модуляції. Для частот від 30 до 50 кГц зазвичай використовують світлодіоди з довжиною хвилі 950 нм, а для 455 кГц — спеціальні світлодіоди з довжиною хвилі 870 нм (спеціальні приймачі TSOP5700 і TSOP7000 орієнтовані на цю довжину хвилі і високу частоту модуляції).

Кілька з цих модульованих передач і гасінь (імпульсних спалахів) утворюють кодоване повідомлення. Приймач ІЧ-сигналу складається з декількох каскадів підсилювачів і демодулятора (детектора частоти) і чутливий до сигналу до -90 дБ (більшість аматорських радіосхем мають чутливість до -60 дБ). Крім того, майже всі ІЧ-приймачі серійного виробництва мають ІЧ-світлофільтр (темно-червоні лінзи або пластини). Команда у форматі HEX виглядає так: 0000 FREQ CNT1 CNT2 ON_1 OFF1 ON_2 OFF2 ON_nOFFn.

0000 – це завжди чотири нулі, але для наших цілей ми використаємо перші два нулі та замінимо їх кількістю повторень коду, а два других нулі будуть вихідним числом на arduino. Таким чином, код 010D двічі повторить команду і відправить її на вихід 13 (той, що має вбудований світлодіод); - FREQ - опорна частота сигналу. Зазвичай в діапазоні 35-40 кГц і пишеться хитро - 36 кГц = 0073 = 115, не дуже зрозуміло чому десяткова 115 дає частоту 36 кГц, можу тільки сказати, що це можна порахувати так 4145 / FREQ; - CNT1 - допоміжна частина; - CNT2 - допоміжна частина. - ON - дані. Це кількість періодів мерехтіння ІЧ діода; - OFF - кількість періодів, коли на нозі зберігається логічний нуль. Це виглядає так (рис. 1.4):

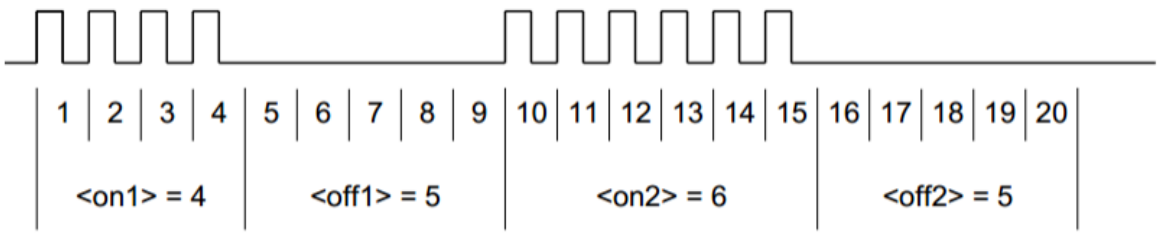


Рис. 1.4 Командні імпульси

У складі середовища arduino [13-15] є спеціальна функція tone (пін, частота, тривалість), вона генерує сигнал на порт введення / виводу - прямокутну «хвилю», заданої частоти і з шпаруватістю 50%, використання переривань таймера. Це дозволяє використовувати генерацію ІЧ-команд для дистанційного керування обладнанням.

Основні переваги адресних аналогових систем полягають у:

– використання кільцевої структури шлейфу, що значно підвищує надійність всієї системи (у разі обриву шлейф продовжує повноцінно працювати);

– можливість подальшого нарощування системи без додаткових витрат;

– проста організація мережі панелей і повторювачів з організацією взаємодії між усіма вузлами мережі;

– можливість використання волоконно-оптичних ліній зв’язку між панелями;

– велика кількість сервісних функцій, що полегшують обслуговування системи (реєстрація подій, ручне/автоматичне відключення датчиків/зон, автоматичне попередження про необхідність очищення датчиків);

– використання спеціальних ізоляторних модулів для ізоляції короткого замикання в шлейфі, що також підвищує надійність шлейфу;

– алгоритми, що запобігають хибним спрацьовуванням (режим день/ніч, змінний еталонний рівень чутливості датчиків з автоматичною компенсацією забруднення, перевірка збігу тривог);

– використовуйте менше кабелю для підключення датчиків;

– можливість простої інтеграції з системами автоматизації будівлі (димовидалення, надлишковий тиск повітря, вентиляція, ліфти);

– потужні програмні графічні засоби з гнучкою архітектурою. Як зазначалося вище, важлива перевага адресного аналога.

1.4 Визначення функціональних вимог щодо модулю кіберзахисту системи керування розумного будинку

На основі попереднього аналізу визначмо функціональні вимоги:

1. Доступність — можливости використання.

2. Інтерфейс — наявність інтуїтивного інтерфейсу.

3. Масштабованість -властивість пов’язана з зручністю розширення.

4.Автономність — властивість пов’язана з наявністю працездатности без звязку.

5.Гнучкість — властивість пов’язана з адаптацією.

6.Сумісність — властивість пов’язана з підтримкою широкого спектру протоколів обміну.

7. Аудит — наявність засобів контролю коректности роботи системі.

Таблиця 1.4

Порівняльна характеристика засобів оцінювання, що вже існують

Характеристика	Ajax Systems	BroadLink	Fibaro	Orvibo	Xiaomi
Доступність	+	+	+	+	+
Інтерфейс	+	-	-	-	+

Закінчення таблиці 1.4

Характеристика	Ajax Systems	BroadLink	Fibaro	Orvibo	Xiaomi
Масштабов-ність	+	+	+	+	+
Автономність	+	-	+	-	-
Гнучкість	+	-	-	-	-
Сумісність	-	+	-	-	+
Аудит	-	-	-	-	-

Висновок до розділу 1

В результаті виконання розділу 1 отримані наступні результати:

Проаналізовані існуючі інформаційні технології в інтелектуальному будинку.

На основі результату проведеного аналізу визначено функціональні вимоги щодо модулю кіберзахисту системи керування розумного будинку

РОЗДІЛ 2
ПОБУДОВА АРХІТЕКТУРИ МОДУЛЮ КІБЕРЗАХИСТУ В АСПЕКТАХ ТОПОЛОГІЇ, ФОРМАТУ ТА ПОЛІТИК ПІДКЛЮЧЕННЯ

Основним елементом системи керування розумного будинку є фаєрвол.

Визначимося з вимогами до фаєрволу та його налаштувань.

Вимоги до фаєрволу:

1. Безпека - забезпечення надійного захисту мережі від несанкціонованого доступу та атак зовнішніх зловмисників.

2. Керованість - можливість налаштування правил фільтрації трафіку для дозволу або блокування конкретних портів та протоколів.

3. Масштабованість - здатність фаєрволу працювати ефективно в мережах будь-якого розміру, від невеликих домашніх мереж до корпоративних інфраструктур.

4. Відмовостійкість - забезпечення неперервної роботи фаєрволу навіть у випадку відмови деяких його компонентів або атаки.

2.1 Вибір технології та топології підключення модулю кіберзахисту.

Для внутрішнього фаєрволу в інтелектуальному будинку можна розглянути використання мікроконтролерів, таких як Arduino, як основи для реалізації функціоналу фаєрволу. Однак, слід врахувати, що Arduino може бути не зовсім належним варіантом для таких завдань через обмежені можливості та обробку великої кількості даних. Рекомендується розглянути більш потужні мікроконтролери або використання спеціалізованих пристроїв або програмного забезпечення для реалізації фаєрволу в розумному будинку, наприклад, Raspberry Pi або спеціалізовані маршрутизатори з підтримкою функцій захисту мережі.

Розглянемо варіанти побудові інформаційного обміну в системі керування розумним будинком. Варіант з прямим підключенням показано рис. 2.1.

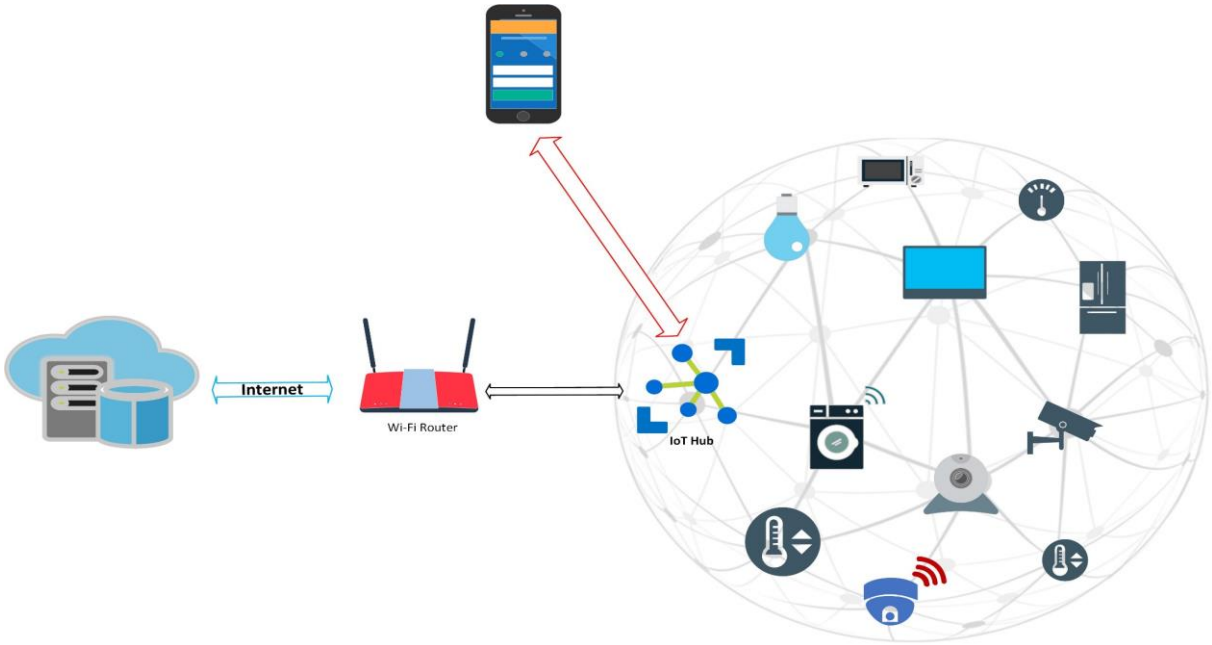


Рис 2.1 Пряме підключення абонента що керує до фаєрволу

При прямому керуванні існує переваги більш надійно підключення абонента що здійснює керування але недоліком є обмеження дальності керування відповідно абонент має знаходиться в зоні прямого зв'язку. Відповідно кількість можливих атак зменшується, що дозволяє використовувати більш простий механізми захисту що спрощує його реалізацію.

Варіант інтернет підключення абонента, що керує, до фаєрволу див. рис. 2.2. Більш складний але не залежить від місця знаходження абоненту. Більш складні механізми захисту вимагають більших системних ресурсів та часу їх реалізації.

Не зважаючи на це він є найбільш привабливим для реалізації. В першу чергу це обумовлено зручністю керування та доступністю інформації о стані елементів розумного будинку на будь якої відстані. Тому рішення о виборі варіанту топології підключення необхідно приймати на основі аналізу можливих атак. Попереднє зупинімся на цьому варіанти але розглянемо для повноти аналізу альтернативний варіант. Альтернативнім варіантом є пряме підключення компонентів системи розумного будинку до хмари. Цей варіант показано на рис.

2.3. Основою для нього можуть бути пропозиції світових бренд компаній які пропонують окремі елементи з прямим підключенням до хмари.

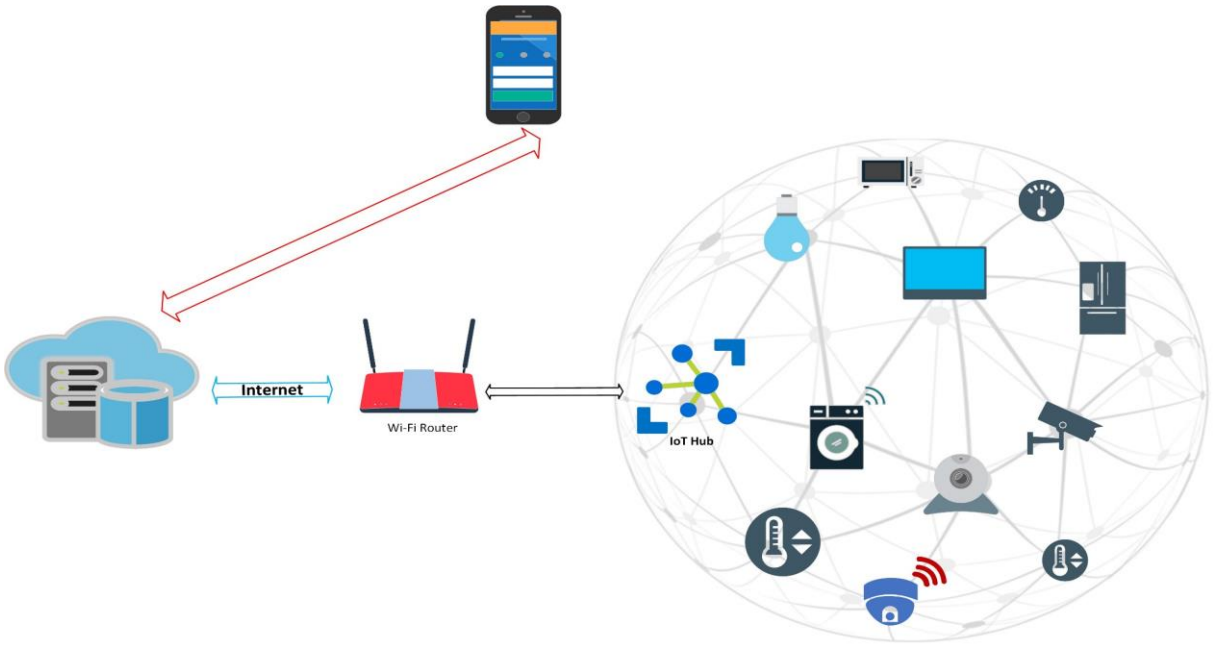


Рис 2.2 Інтернет підключення абонента що керує до фаєрволу

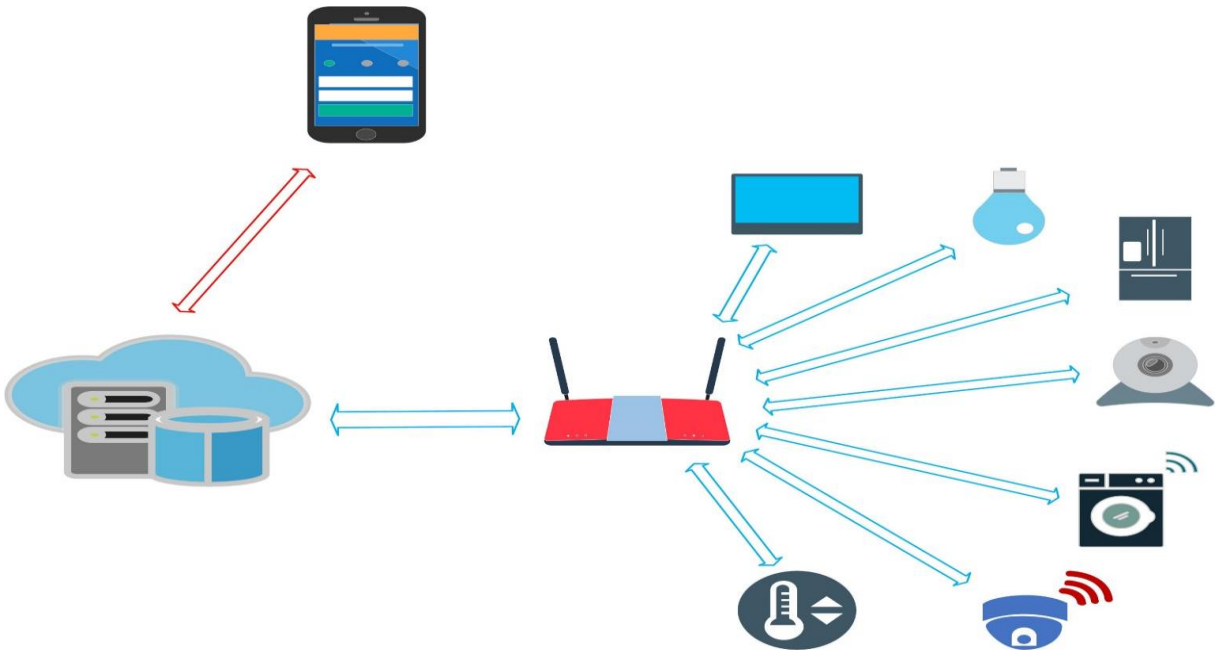


Рис 2.3 Пряме підключення абонента що керує до пристроїв розумного будинку

Наприклад - BroadLink пропонує широкий спектр систем розумного будинку, в том числі - системи автоматизації - для увімкнення/вимкнення освітлення, регулювання температури, керування розумними пристроями тощо.

Альтернативний варіант хорош в випадку необхідності автоматизації одної або пари функцій розумного будинку. Недоліком цього варіанту необхідність довірі до виробника. Нажаль завжди є вірогідність компрометації реалізованих механізмів захисту, яка прямо пропорційна популярності розробника.

В підсумку відзначимося з першими кроками вибору при побудові модулю кіберзахисту системи керування розумного будинку :

- Топологія інтернет підключення з використанням фаєрволу;
- Для реалізації фаєрволу в розумному будинку буде розроблене програмне забезпечення для мікроконтролеру Raspberry Pi.

2.2 Вибір варіанту підключення модулю кіберзахисту.

Розглянемо чотири варіанта підключення модулю кіберзахисту.

А саме:

- Підключення фаєрволу між роутером та мережею розумного будинку (див. рис 2.4);
- Підключення фаєрволу в якості роутеру (див. рис 2.5);
- Підключення фаєрволу в якості IoT контролеру мережі розумного будинку (див. рис 2.6);
- Підключення фаєрволу до роутеру (див. рис 2.7).

Перший варіант найбільш очевидній але має два суттєвих недоліків.

Додає зайвий пристрій та залишає відкритим для нападу роутер.

Варіант три віршує перше, а четвертий друге. З цієї точки зору привабливим є другій варіант якій вирішує всі питання. Але існує ще низка факторів які потрібно врахувати. По перше це складність функціоналу, по друге час розробки, по третє вартість розробки.

Потрібно враховувати що середній час розробки якісного, в том числі модифікованого під наші вимоги роутера, приблизно півтора два роки, а атестаційну роботу потрібно написати за два місяці маємо приймати складні рішення.

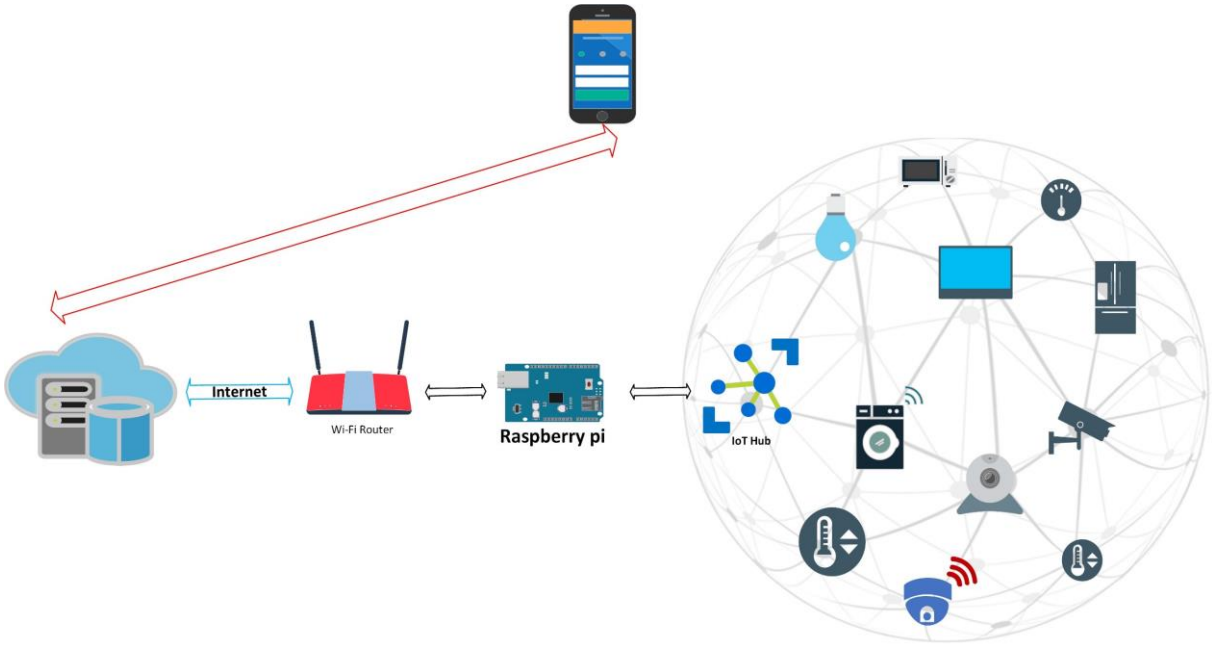


Рис 2.4 Підключення фаєрволу між роутером та мережею розумного будинку

Визначив привабливість другого варіанту (рекомендувавши його реалізацію в майбутньому) обмежити вибір першим варіантом.

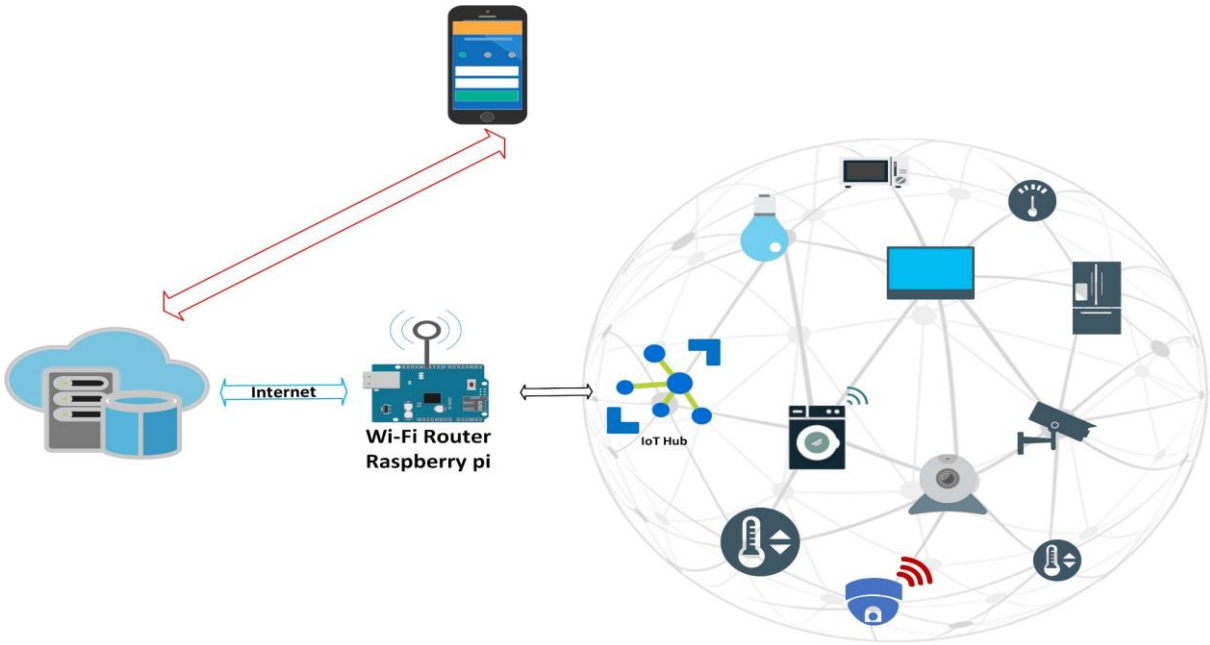


Рис 2.5 Підключення фаєрволу в якості роутеру.

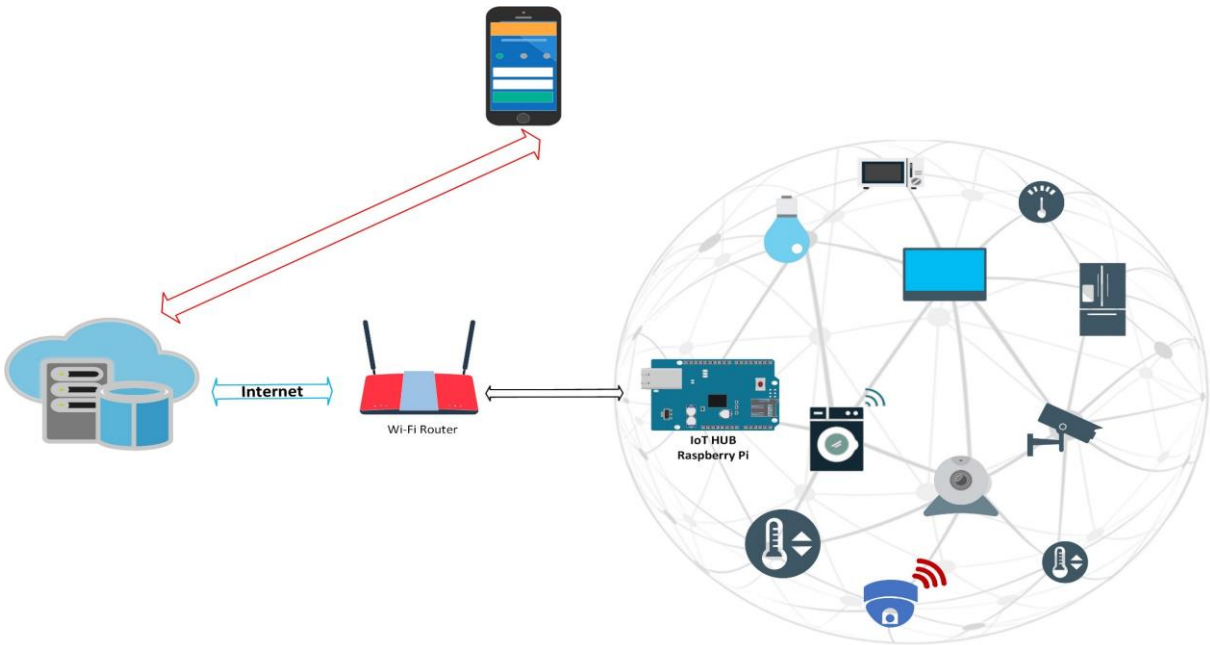


Рис 2.6 Підключення фаєрволу в якості IoT контролеру мережі розумного будинку

Також доцільне провести аналіз можливих атак для визначення функціоналу модуля захисту.

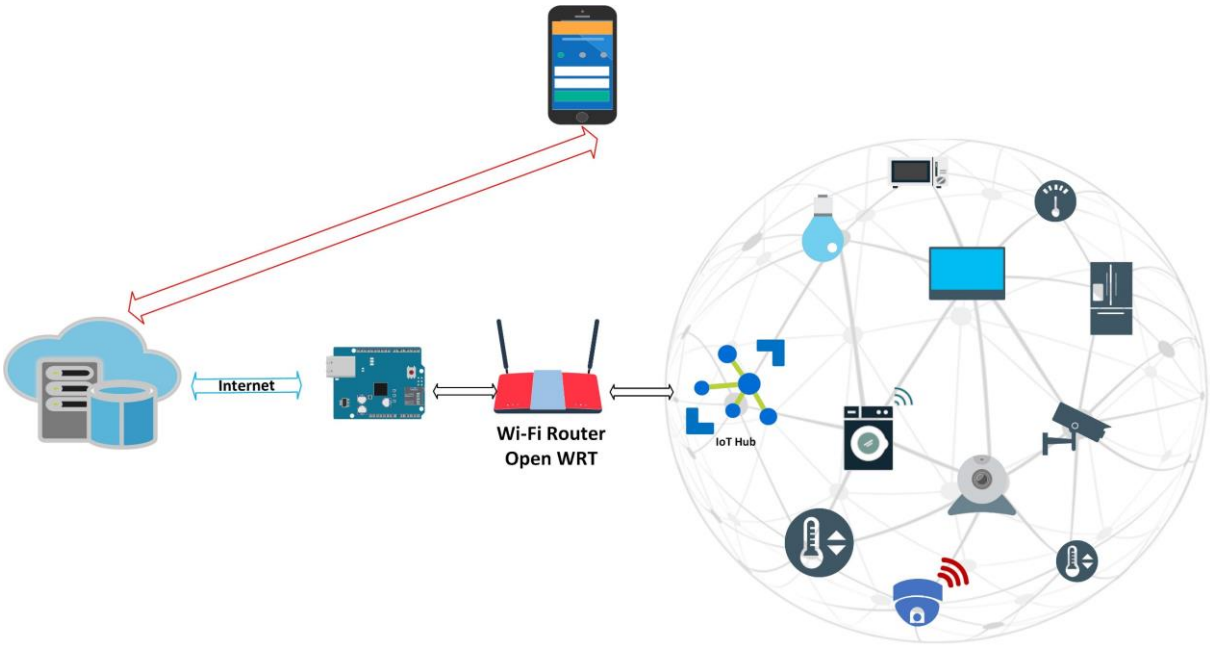


Рис 2.7 Підключення фаєрволу до роутеру

2.3 Аналіз архітектури безпеки при керування розумним будинком

Попередньо було прийнято рішення про використання платформи Home Assistant в якості операційного середовища для модуля кіберзахисту. Розглянемо її архітектуру, як вже було сказане вище Home Assistant це платформа для управління розумним будинком і домашньої автоматизації. Home Assistant — це вбудована система, яка забезпечує інтеграцію прикладних застосунків, додатково забезпечує: адаптацію, конфігурацію та оновлення які виконуються через простий у використанні інтерфейс.

В мінімальної конфігурації операційна система надає мінімальне середовище Linux, налаштоване для запуску основних компонентів Home Assistant. Це середовище включає лише необхідні елементи, забезпечуючи ефективне використання ресурсів.

Окремо маємо згадати Supervisor. Supervisor — це важливий компонент, який керує операційною системою та стежить за загальним станом системи. Він відповідає за такі завдання, як оновлення програмного забезпечення, створення резервних копій та забезпечення нормальної роботи компонентів системи, включаючи основний компонент Home Assistant. Завдяки такому підходу значно спрощується структура модулю кіберзахисту тому що ці функції виконуються компонентом Supervisor.

Також важливим є Core. Home Assistant Core — це основний елемент, який взаємодіє з користувачем, керує IoT-пристроями та сервісами, а також виконує автоматизовані сценарії. Відповідно також виконує задачі автоматизації, інтеграції та взаємодії з користувачем через інтерфейс. Core розроблений для інтеграції з сторонніми IoT-пристроями та сервісами, створюючи єдине сумісне рішення для автоматизації будинку.

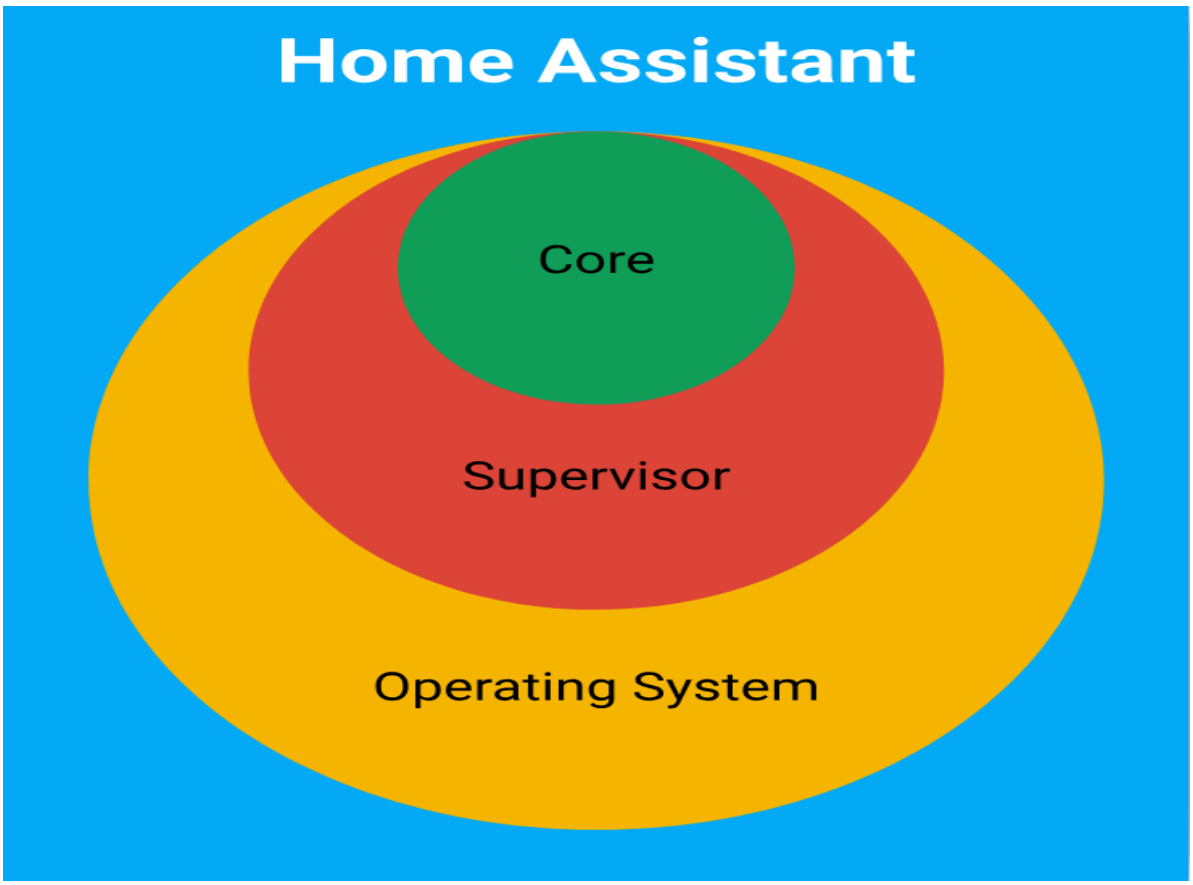


Рис 2.8 Загальна структура операційної системи Home Assistant (HAOS)

Операційна система Home Assistant (HAOS) — це спеціально розроблена ОС для запуску Home Assistant на одноплатних комп'ютерах . Її основне завдання — забезпечити стабільну та безперебійну роботу Home Assistant, гарантуючи надійність функціонування .

Ключові компоненти HAOS: Завантажувач, Операційна система, Файлові системи, Платформа контейнерів, Оновлення, Безпека.

Завантажувач є два типа GRUB та U-Boot. Перший GRUB - використовується для пристроїв, що підтримують UEFI (Універсальний розширюваний інтерфейс прошивки). Другий U-Boot - використовується для пристроїв, які не підтримують EFI.

Операційна система використає Buildroot. Buildroot, це система збірки, призначена для створення компактних та оптимізованих Linux-дистрибутивів. Buildroot не являється класичним Linux-дистрибутивом, а є набором інструментів для створення налаштованого середовища Linux. Buildroot

використовується для створення кастомізованих Linux-систем, придатних для вбудованих пристроїв з обмеженими ресурсами, таких як одноплатні комп'ютери, маршрутизатори, пристрої IoT і т.д.

Файлові системи має декілька модифікації. SquashFS: це файлова система з підтримкою тільки для читання (read-only), яка використовується для зберігання даних у вигляді стиснутого образу. Вона часто застосовується для створення компактних та ефективних файлових систем у вбудованих пристроях. ZRAM: Використовується для тимчасових директорій, таких як /tmp, /var і swap, забезпечуючи стиснуті в оперативній пам'яті блоки даних для підвищення продуктивності, також із використанням LZ4 стиснення.

Наступною є платформа контейнерів Docker Engine. Docker використовує контейнеризацію для запуску компонентів Home Assistant, забезпечуючи ізоляцію кожного компонента та його незалежну роботу. Такий підхід додає гнучкості та надійності системі.

Модуль оновлення RAUC: Забезпечує механізми оновлення по повітрю (OTA) та через USB. Це дає змогу доставляти оновлення дистанційно або через USB-накопичувач, що спрощує оновлення HAOS.

І наприкінці найбільш цікавій нам AppArmor. AppArmor це модуль безпеки для ядра Linux, який забезпечує контроль доступу до ресурсів на основі політик. AppArmor дозволяє обмежувати поведінку програм, визначаючи, до яких ресурсів (файли, мережі, пристрої тощо) вони можуть мати доступ і які операції можуть виконувати.. AppArmor допомагає ізолювати та обмежувати потенційні безпекові ризики, забезпечуючи безпечне середовище для роботи компонентів Home Assistant.

Розглянемо більш детально Home Assistant Supervisor.

Home Assistant Supervisor — це ключовий компонент системи, що забезпечує централізоване керування та взаємодію з установкою Home Assistant. Він відповідає за автоматичне оновлення, створення резервних копій, моніторинг системи та управління додатками, забезпечуючи ефективну та безпечну роботу. Завдяки Supervisor користувачі можуть легко керувати своєю системою без необхідності вручну налаштовувати кожну функцію.

Основні задачі Supervisor:

-Запуск Home Assistant Core:

Supervisor керує роботою Home Assistant Core, центральної платформи автоматизації, яка є основою системи.

-Оновлення Home Assistant Core:

Він забезпечує актуальність Home Assistant Core, оновлюючи систему новими функціями, удосконаленнями та безпековими патчами. Якщо оновлення не вдалося, Supervisor автоматично повертає попередню версію, щоб уникнути перебоїв у роботі системи.

-Створення та відновлення резервних копій:

Supervisor відповідає за створення резервних копій всієї установки Home Assistant, включаючи конфігурації, налаштування та інтеграції. Він також дозволяє користувачам відновити систему з цих копій за необхідності.

-Додатки (Add-ons):

Supervisor дозволяє користувачам встановлювати та керувати додатками, які є додатковими застосунками або сервісами, що розширюють функціональність Home Assistant. Ці додатки можуть включати такі сервіси, як бази даних, медіасервери або MQTT брокери.

-Єдина аудіосистема:

Supervisor допомагає керувати аудіофункціями, дозволяючи як Home Assistant Core, так і додаткам відтворювати аудіо по всій системі. Ця функція корисна для голосових асистентів, медіаплеєрів та систем сповіщень.

-Оновлення операційної системи Home Assistant:

У деяких типах установки Supervisor відповідає за оновлення основної операційної системи Home Assistant (OS), забезпечуючи актуальність та безпеку всіх компонентів системи.

Архітектура Home Assistant є модульною та масштабованою, де кожна частина системи виконує свою конкретну роль. Ось розподіл основних архітектурних компонентів

:

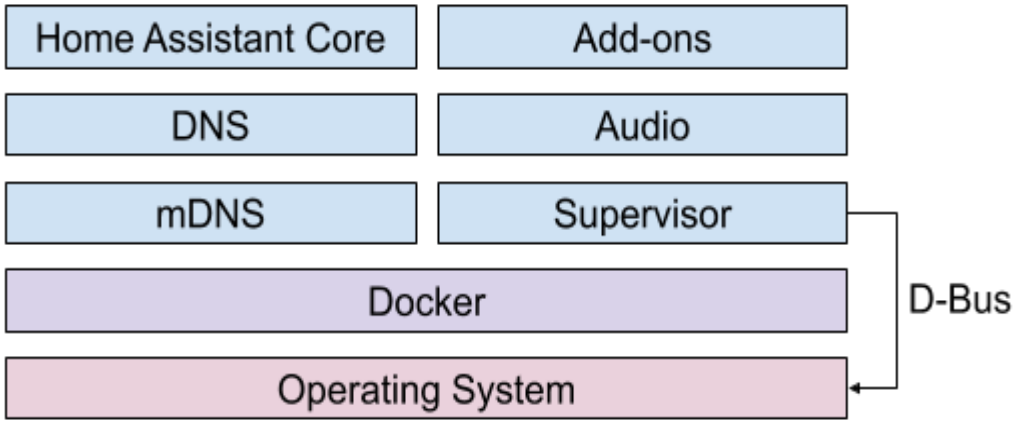


Рис 2.9 Архітектура Home Assistant

Home Assistant Core: Home Assistant Core — центральна платформа автоматизації, відповідальна за керування пристроями, створення автоматизацій та взаємодію з користувачем.

Додатки (Add-ons): Це додаткові застосунки, які запускаються на установці Home Assistant для розширення її можливостей. Додатки можуть включати такі сервіси, як бази даних, медіасервери, MQTT брокери та інші. Кожен додаток працює в окремому контейнері Docker.

DNS (Система доменних імен): DNS дозволяє Home Assistant Core та додаткам взаємодіяти між собою. Він допомагає розв'язувати імена хостів у IP-адреси, що дозволяє компонентам взаємодіяти один з одним по мережі.

Аудіо: Цей компонент дозволяє як Home Assistant Core, так і додаткам відтворювати та керувати аудіо. Це корисно для голосових сповіщень, відтворення медіа чи інших аудіо функцій.

mDNS (Multicast DNS): mDNS допомагає Home Assistant Core та додаткам знаходити та підключатися до пристроїв і сервісів у локальній мережі. Це особливо важливо для безконфігураційної мережевої взаємодії, дозволяючи пристроям, таким як розумні лампочки, колонки чи камери, бути швидко виявленими.

Supervisor: Supervisor відповідає за управління всією системою, включаючи забезпечення актуальності та стабільної роботи. Він контролює роботу Home Assistant Core, додатків та інших компонентів, забезпечуючи їх правильне функціонування.

Docker: Docker використовується для запуску застосунків у контейнерах. Як Home Assistant Core, так і додатки працюють у контейнерах Docker, що забезпечує ізоляцію, простоту оновлення та стабільність системи.

Операційна система: Home Assistant Operating System — це легка операційна система на базі Linux, оптимізована для роботи з Home Assistant. Вона забезпечує необхідне середовище для запуску Home Assistant Core, додатків та інших сервісів.

D-Bus: D-Bus — це система комунікації, яка дозволяє Home Assistant Core, додаткам та операційній системі взаємодіяти. Вона надає механізм для керування компонентами системи, такими як мережевий менеджер, налаштування звуку та апаратні пристрої.

Архітектура Home Assistant Core

Home Assistant Core складається з чотирьох основних частин і численних допоміжних класів для роботи з поширеними сценаріями, такими як створення сутностей або робота з локаціями.

Event Bus: забезпечує механізм ініціації та прослуховування подій — це основа системи Home Assistant.

State Machine: відстежує стан різних елементів і генерує подію state_changed, коли стан змінюється.

Service Registry: прослуховує події call_service на event bus та дозволяє іншим частинам коду реєструвати дії сервісів.

Timer: кожну секунду генерує подію time_changed, що передається через event bus.

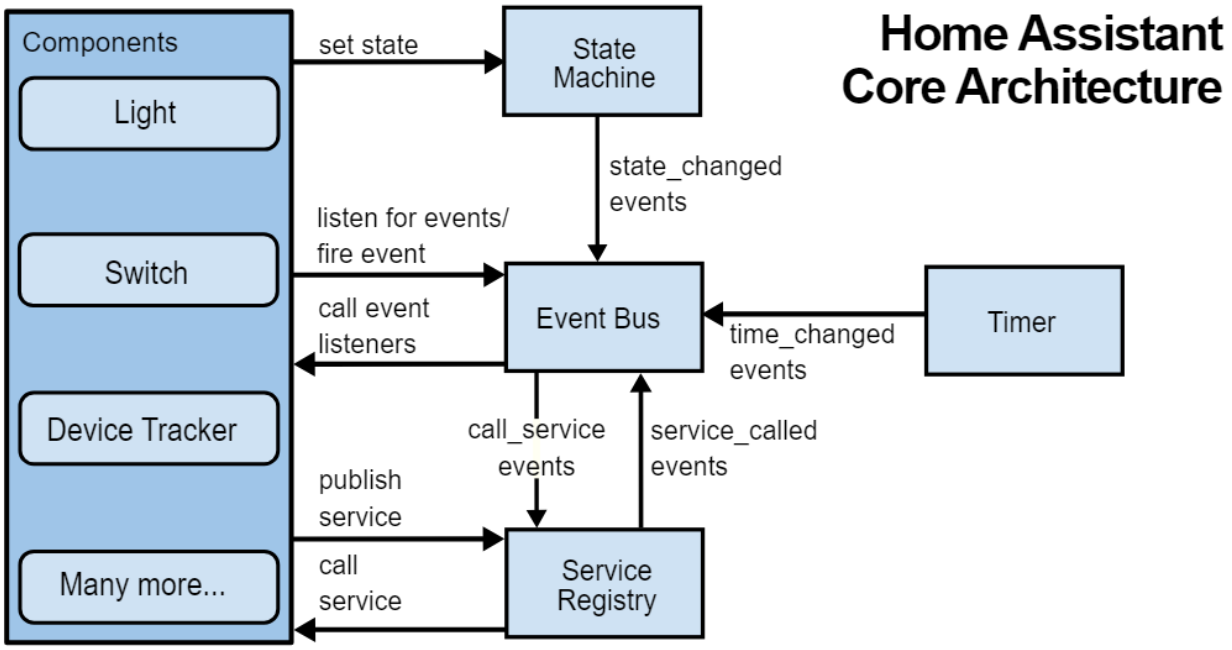


Рис 2.10 Архітектура Home Assistant Core

AppArmor — це встановлена технологія, яка вперше була помічена в Immunix і пізніше інтегрована в Ubuntu, Novell/SUSE і Mandriva. Основна функціональність AppArmor знаходиться в основному ядрі Linux, починаючи з версії 2.6.36; Тривають роботи AppArmor, Ubuntu та іншими розробниками для злиття додаткового функціоналу AppArmor в основне ядро.

AppArmor є реалізацій системі Mandatory Access Control (MAC), яка базується на архітектури Linux Security Modules (LSM). Модель безпеки Apparmor базується на прив’язі атрибутів контролю доступу не к користувачам , а к застосункам. AppArmor здійснює ізоляцію за допомогою профілів, які додаються в ядро, при завантаженні.

AppArmor відрізнятися от інших реалізацій MAC в Linux принципом дії на основі шляхів, а також він дозволяє змішувати профілі примусового виконання та режим попередження. Архітектура дискреційного контролю доступу (DAC) обмежує доступ до критичних ресурсів на основі атрибутів суб'єктів або групи,

до якої вони належать. Ці атрибути визначають права доступу до ресурсів файлової системи. Значення привілеїв Read, Write та eXecute добре відомі.

Ці атрибути застосовуються до трьох категорій користувачів: власник, група та інші. Категорія власника відноситься до одного користувача ОС, тоді як група може містити багато користувачів ОС. У категорію решти входять ті користувачі, які не належать до перших двох.

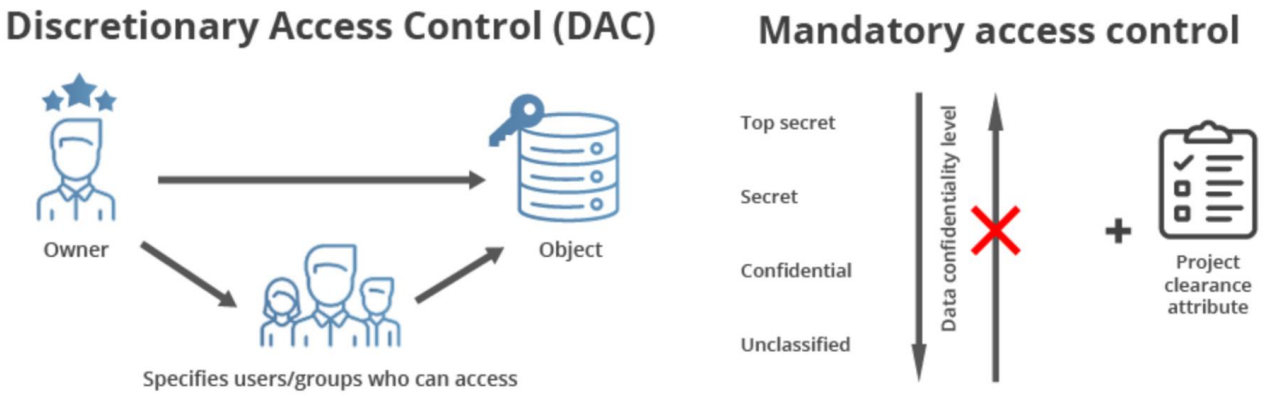
Модель DAC надає власнику ресурсу право визначати тип доступу для заданих категорій користувачів. Таке розмежування доступу підходить для захисту від ненавмисних дій користувача і дозволяє відповісти на наступні питання:

• Які ресурси ФС доступні даному користувачеві ОС для читання, запису та виконання?

• Які ресурси ФС читаються, записуються і виконуються доступні цій групі?

• Які ресурси ФС доступні іншим користувачам для читання, запису та виконання?

• Хто з користувачів має достатні права для початку цього процесу?



•

Рис 2.11 Системи безпеки DAC та MAC

Система безпеки Mandatory Access Control (MAC) основана на централізованому контролю за правилами політики доступу, при якому звичайні користувачі не мають можливості вносити в них будь-які зміни.

Розробник політик визначає, які програми або процеси можуть виконувати певні дії над системними ресурсами. MAC більше орієнтується на програми, ніж на користувачів і вирішує проблему розмежування доступу процесів до ресурсів ОС.

По суті, дизайн MAC намагається відтворити розмежування привілеїв доступу до документації у фізичному світі. Якщо певний співробітник має право читати документи з грифом «цілком таємно», то йому також доступні стандартні конфіденційні і внутрішні документи. Однак це не так. Те ж саме справедливо і в контексті привілеїв доступу процесів ОС в архітектурі MAC. Наприклад, якщо програма може прочитати файл /etc/sudoers, то вона також має доступ до /etc/hosts, але протилежне не вірно.

Приклади профілів можна знайти в пакеті apparmor-profiles від universe, а за замовчуванням будуються профілі примусового виконання:

Таблиця 2.1

Приклади профілів

Source package/binary	**12.04 LTS**	**14.04 LTS**	**16.04 LTS**	**18.04 LTS**	**20.04 LTS**	**20.10**
Akonadi (mysqld)	yes	yes	yes	yes	yes	yes
Apache (apache2)	yes[1]	yes[1]	yes[1]	yes[1]	yes	yes
Bind (named)	yes	yes	yes	yes	yes	yes
ClamAV (clamd,freshclam)	yes	yes	yes	yes	yes	yes
Cups (cupsd)	yes	yes	yes	yes	yes	yes

Продовження таблиці 2.1

Source package/binary	**12.04 LTS**	**14.04 LTS**	**16.04 LTS**	**18.04 LTS**	**20.04 LTS**	**20.10**
Evince	yes	yes	yes	yes	yes	yes
Firefox (firefox-3.5/firefox)	yes[1]	yes[1]	yes[1]	yes[1]	yes	yes
gdm-guest-session	N/A	N/A	yes	yes	yes	yes
ISC Dhcpd (dhcpd3/dhcpd)	yes	yes	yes	yes	yes	yes
ISC Dhcp client (dhclient3/dhclient)	yes	yes	yes	yes	yes	yes
juju	yes[2]	yes[2]	yes[2]	yes[2]	yes	yes
Libvirt (libvirtd and kvm/qemu guests)	yes	yes	yes	yes	yes	yes
Lightdm guest session	yes	yes	yes	--	--	--
LXC	yes[3]	yes[3]	yes[3]	yes[3]	yes	yes
MAAS dhcpd (dhcpd)	yes	yes	yes	yes	--	--
MySQL (mysqld)	yes	yes	yes	yes	yes	yes
NTP (ntpd)	yes	yes	yes	--	--	--
OpenLDAP (slapd)	yes	yes	yes	yes	yes	yes
quassel-core	yes	yes	yes	yes	yes	yes
rsyslog	yes[1]	yes[1]	yes[1]	yes[1]	yes	yes
tcpdump	yes	yes	yes	yes	yes	yes

Продовження таблиці 2.1

Source package/binary	**12.04 LTS**	**14.04 LTS**	**16.04 LTS**	**18.04 LTS**	**20.04 LTS**	**20.10**
Telepathy	yes	yes	yes	--	--	--
AppStore apps (click)[4]	--	yes	yes	--	--	--
Cups filters (cups-browsed)	--	yes	yes	yes	yes	yes
lightdm-remote-session-freerdp	--	yes	yes	--	--	--
lightdm-remote-session-uccsconfigure	--	yes	yes	--	--	--
media-hub	--	yes	yes	--	--	--
mediascanner2	--	yes	yes	--	--	--
squid3	--	yes[1]	yes[1]	yes[1]	yes	yes
sssd	--	yes[1]	yes[1]	yes[1]	yes	yes
StrongSwan (stroke/lookip)	--	yes	yes	yes	yes	yes
Telepathy (ofono)	--	yes	yes	yes	yes	yes
AppStore apps (snappy)[5]	--	--	yes	yes	yes	yes
libvirt (libvirt-lxc containers)	--	--	yes	yes	yes	yes
LXD	--	--	yes	yes	yes	yes
snap-confine (aka ubuntu-core-launcher)	--	--	yes	yes	yes	yes
ubuntu-download-manager (extractor)	--	--	yes	--	--	--

Закінчення таблиці 2.1

Source package/binary	**12.04 LTS**	**14.04 LTS**	**16.04 LTS**	**18.04 LTS**	**20.04 LTS**	**20.10**
webbrowser-app	--	--	yes	--	--	--
chrony	--	--	--	yes	yes	yes
ippusbxd	--	--	--	yes	yes	yes
libreoffice[6]	--	--	--	yes	yes	yes
man-db	--	--	--	yes	yes	yes
mozc	--	--	--	yes	yes	yes
anope	--	--	--	--	--	yes

Ця таблиця розповсюдженості обраної технології підтверджує наш вибір.

Також перелічим вимоги що безпеки конфігурації.

Немає відкритих портів

Інсталяції Ubuntu за замовчуванням не повинні мати прослуховуваних мережевих служб після початкового встановлення. Винятки з цього правила в настільних системах включають служби мережевої інфраструктури, такі як клієнт DHCP і mDNS (Avahi/ZeroConf, див. ZeroConfSpec для деталей реалізації та обґрунтування). Для Ubuntu у хмарі виняток становлять служби мережевої інфраструктури для хмари та OpenSSH, що працюють із відкритим ключем клієнта та доступом до порту, налаштованим хмарним провайдером. При встановленні Ubuntu Server адміністратор, звичайно, може вибрати конкретні служби для встановлення за замовчуванням (наприклад, Apache).

Хешування паролів

Системний пароль, який використовується для входу в Ubuntu, зберігається в /etc/shadow. Дуже старі хеші паролів базувалися на DES і були видимі в

/etc/passwd. Сучасний Linux вже давно перейшов на /etc/shadow і вже деякий час використовує солоні хеші на основі MD5 для перевірки пароля (crypt id 1). Оскільки MD5 вважається «зламаним» для деяких застосувань, а обчислювальна потужність, доступна для виконання брутфорсингу MD5, зростає, Ubuntu 8.10 і пізніше активно перейшли до використання солоних хешів паролів на основі SHA-512 (crypt id 6), які на порядки складніше піддаються брутфорсу. Ubuntu 22.04 LTS і пізніші версії потім перейшли на yescrypt, щоб забезпечити підвищений захист від злому паролів в автономному режимі. Дивіться man-сторінку crypt для отримання додаткової інформації.

Файли cookie SYN

Коли система перевантажена новими мережевими з'єднаннями, активується використання файлів cookie SYN, що допомагає пом'якшити атаку SYN-флуду.

Автоматичні оновлення системи безпеки

Починаючи з Ubuntu 16.04 LTS, unattended-upgrades налаштований на автоматичне щоденне застосування оновлень безпеки. Більш ранні випуски Ubuntu можна налаштувати на автоматичне застосування оновлень безпеки.

Живі патчі ядра

Служба Canonical Livepatch забезпечує виправлення безпеки більшості основних проблем безпеки ядра без необхідності перезавантаження. Користувачі Ubuntu можуть безкоштовно скористатися послугою на трьох вузлах. Усі машини, на які поширюється підписка на підтримку Ubuntu Advantage, можуть отримувати виправлення в реальному часі.

Вимкніть застарілий протокол TLS

Застарілі версії протоколу Transport Layer Security, включаючи SSL 3.0, TLS 1.0 і TLS 1.1, мають кілька властивих вразливостей і не можуть забезпечити заявлений рівень безпеки. Для цього Ubuntu 20.04 і пізніші версії активно відключають ці версії, встановлюючи планку безпечного зв'язку для протоколів, які сьогодні вважаються безпечними.

Для зв'язку із застарілими системами можна повторно активувати протоколи.

Висновок до розділу 2

В результаті виконання розділу 2 отримані наступні результати:

Проаналізовані існуючі інформаційні технології в інтелектуальному будинку.

На основі аналізу здійснено вибір технології та топології підключення модулю кіберзахисту, визначено шляхи та стандарти його підключення та формати політик безпеки.

РОЗДІЛ 3
ВИЗНАЧЕННЯ ВИМОГ ТА АПАРАТНА РЕАЛІЗАЦІЯ МОДУЛЮ КІБЕРЗАХИСТУ

3.1. Аналіз вимог що до модулю кіберзахисту

Для аналізу будемо використовувати модель запропоновану в [15].

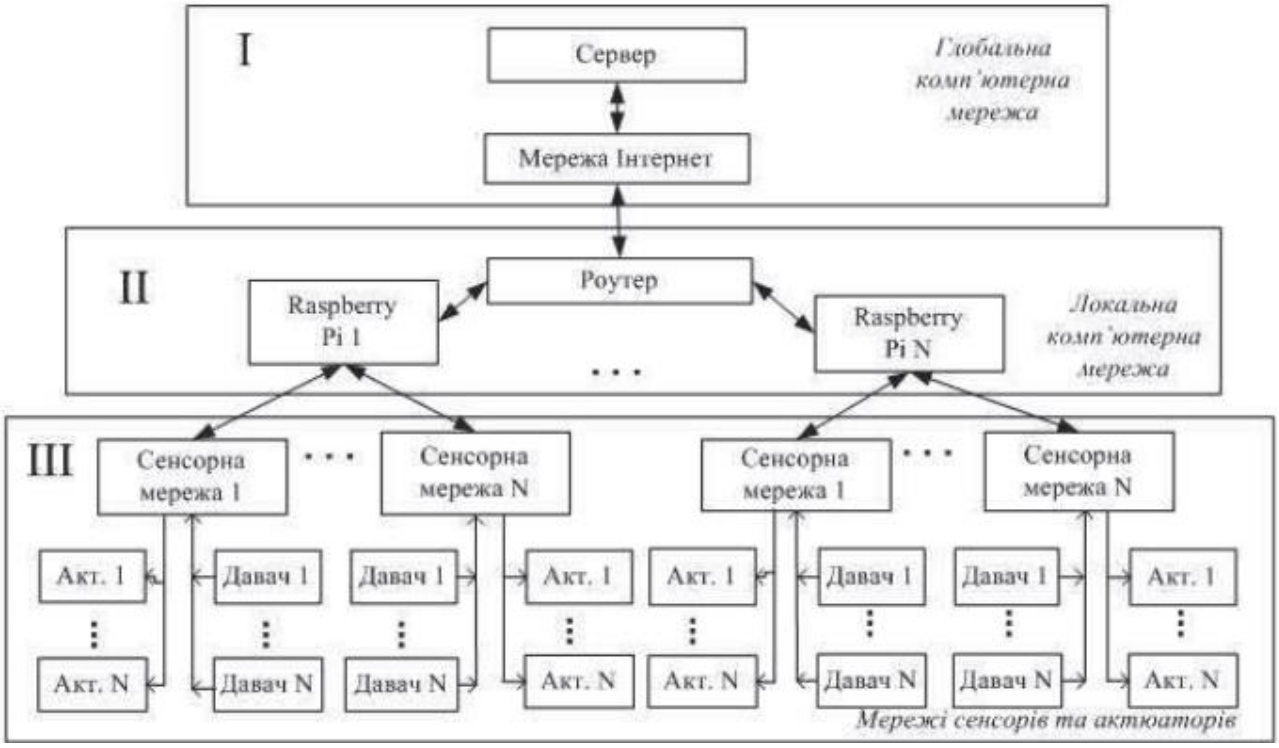


Рис 3.1 Ієрархічна структура телекомунікаційної мережі "інтелектуального будинку"

Ця модель інтерпретує інформаційній обмін як складно підпорядковану ієрархічну структуру. Сервер — Мережа — Роутер — Локальній сервер — Сенсорна або Виконавча або Гібридна мережа.

Пристрої Інтернету речей створюють проблеми з безпекою, і на це є багато причин. Наприклад вони можуть слугувати точками доступу до конфіденційних даних ширшої мережі, збільшуючи поле для атак. Згідно з нещодавнім звітом компанії Parks Associates, яка займається дослідженням споживчого ринку

Інтернету речей, 55% споживачів стурбовані безпекою своїх пристроїв "розумного дому". Якщо хакери зможуть проникнути в розумний пристрій, вони потенційно можуть залишити будинок беззахисним як кібернетичному так і в фізичному аспектах.

Стандартні небезпеки визначені в [9]:

1 Атаки на центральний вузол Підключення мережі «Розумного будинку» до Інтернету. Відсутність (неефективність) механізмів захисту периметра мережі. Порушення роботи або вихід з ладу центрального сервера і як наслідок, всієї системи. Порушення конфіденційності, цілісності та доступності інформації.

2 Вплив вірусних та троянських програм на роботу системи. Підключення мережі «Розумного будинку» до Інтернет. Відсутність (неефективність) механізмів захисту периметра мережі Збої в ПЗ системи, а, отже, порушення роботи або виведення з ладу апаратури системи. Порушення конфіденційності, цілісності та доступності інформації

3 Перехоплення інформації, переданої по дротових та бездротових каналах зв'язку Можливість доступу зловмисника до провідних каналів або до зони стійкого перехоплення радіосигналів мережі. Відсутність (неефективність) механізмів захисту трафіку. Порушення конфіденційності інформації, що передається по каналу. Можливе захоплення управління системою.

4 Доступ зловмисника з правами адміністратора на центральний вузол за допомогою крадіжки паролей та інших реквізитів розмежування доступу Відсутність (неефективність) механізмів аутентифікації та ідентифікації. Порушення конфіденційності, цілісності та доступності інформації, яка знаходиться в мережі.

5 Доступ до мережі неавторизованих користувачів Відсутність (неефективність) механізмів аутентифікації та ідентифікації. Порушення конфіденційності, цілісності та доступності інформації, яка знаходиться в мережі.

6 Помилки користувача Відсутність (неефективність) механізмів захисту системи від неправильних дій користувачів. Порушення конфіденційності,

цілісності та доступності інформації. Можливі збої у системі через неправильне використання обладнання.

7 Поломка апаратури системи Низька надійність обладнання, низька кваліфікація персоналу Порушення конфіденційності, цілісності та доступності інформації

8 Помилки програмного забезпечення Використання неліцензійного ПЗ, низька кваліфікація персоналу, відсутність (неефективність) тестування закупленого ПЗ. Порушення конфіденційності, цілісності та доступності інформації

Сьоме та восьме пов’язано с якістю системи та вирішується організаційними методами. Перше, четверте та п’яте може бути вирішено побудовою якісної системою доступу.

Друге за рахунок навчання користувача.

Сконцентруємось на третій проблемі перехоплення трафіку.

Для зменшення рівня небезпеки доцільно побудувати модуль кібербезпеки якій буде допомагати контролювати його шляхом моніторингу та аудиту.

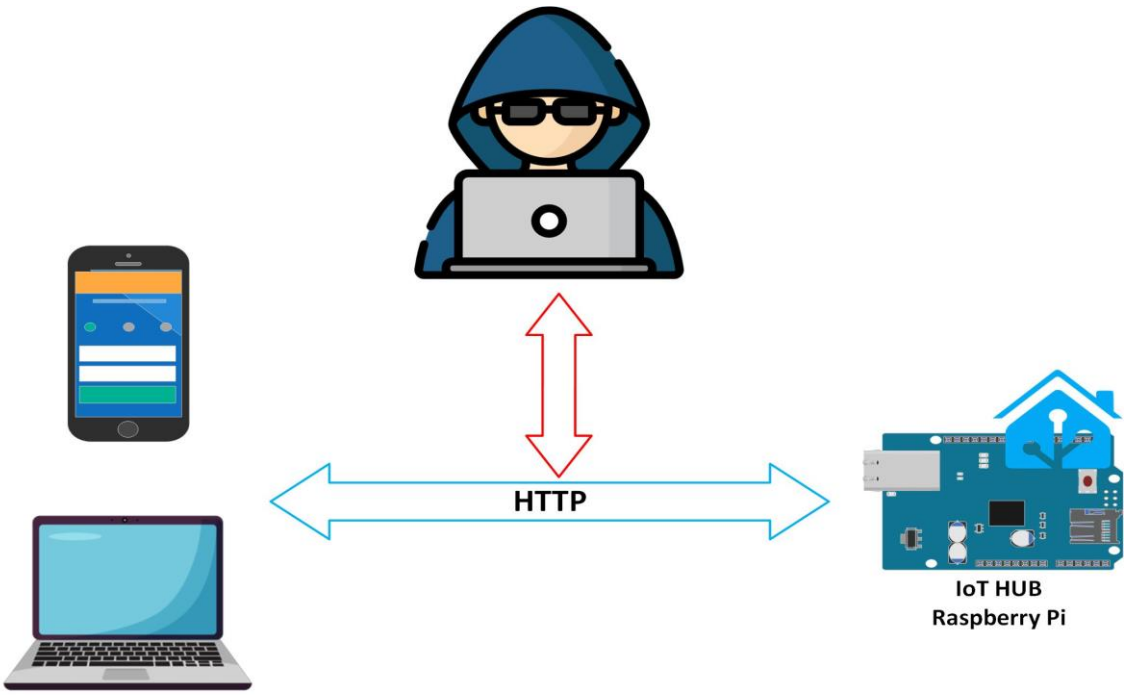


Рис. 3.2. Схема атаки на HTTP

Для визначення вимог роздягнемо схему атаки на HTTP в аспектах захоплення та аналізу трафіку в аспектах пошуку критичної інформації.

3.2. Аналіз атак на розумній будинок

На рис. 3.3 наведено приклад успішного перехоплення інформаційного пакету.

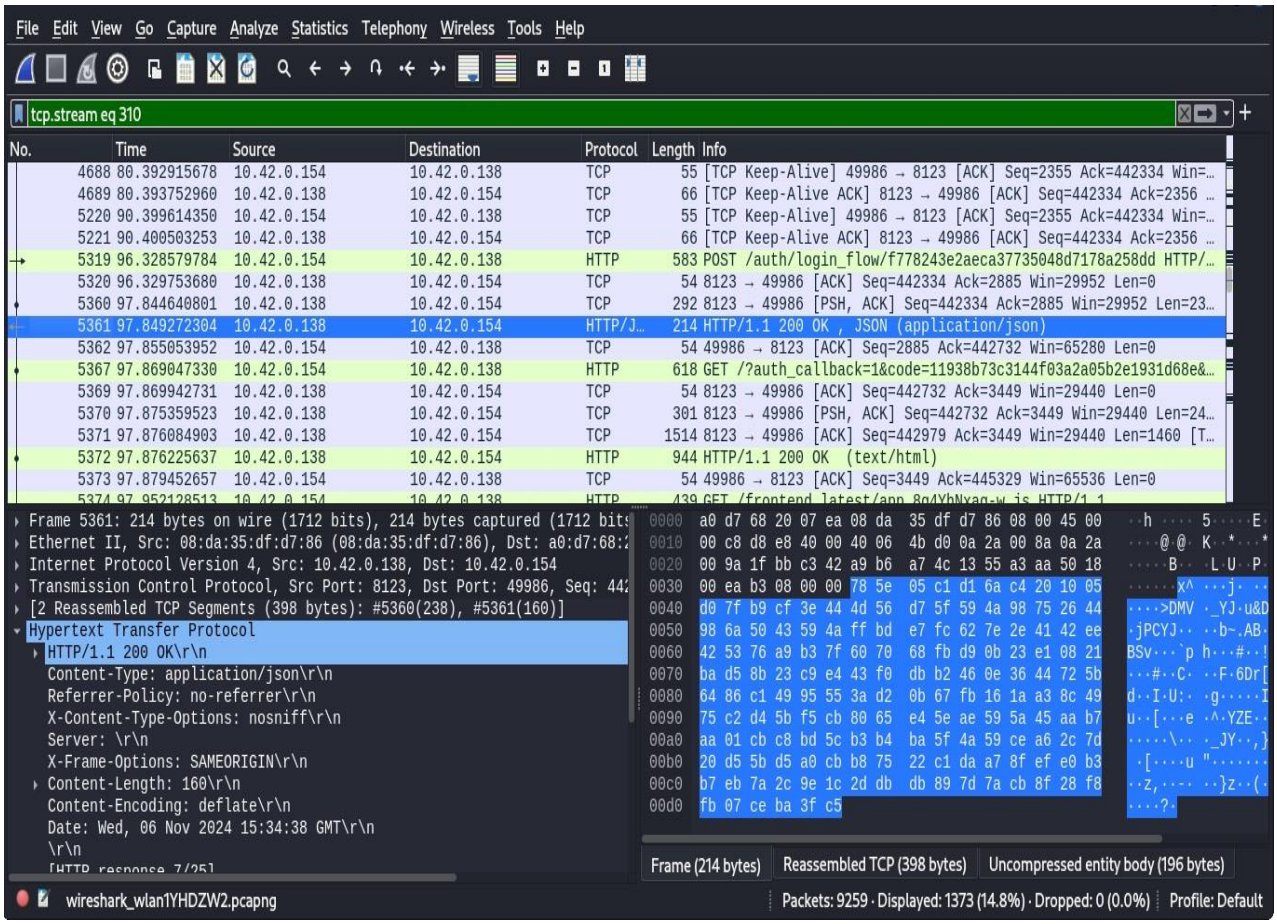


Рис. 3.3 Захоплення та аналіз трафіку

Внизу зліва показано тип перехопленої інформації справа шістнадцятирічній дамп. При удачі хакера, дамп може містити критичну інформацію. На рис 3.4 показана дешифровану дампу, яка містить парольну інформацію. Червоною рамкою відокремлено критичну інформацію.

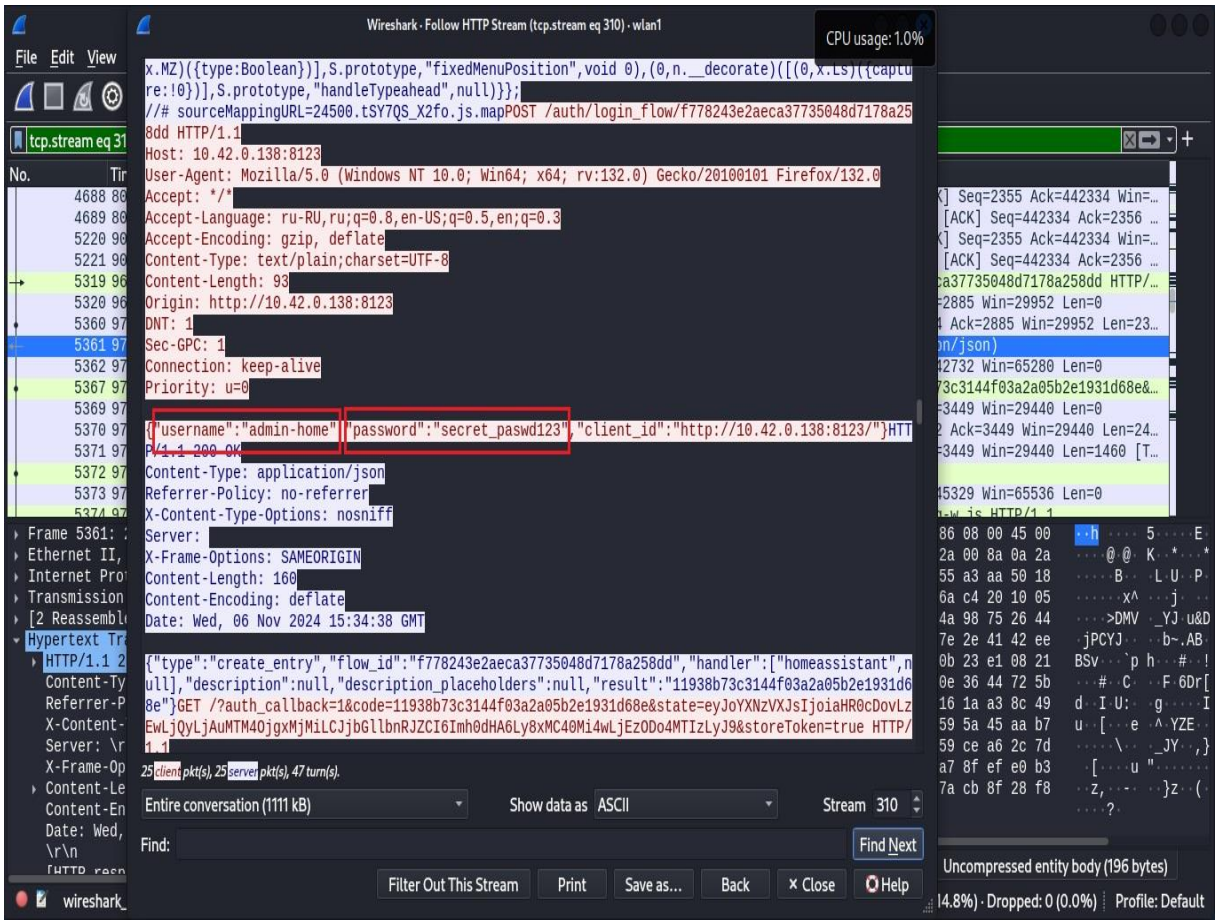


Рис. 3.4 Пошук паролю

Також неприємні наслідки може мати успішне сканування портів. На рис. 3.5 наведено результат успішного сканування портів а саме наявність чотирьох відкритих портів.

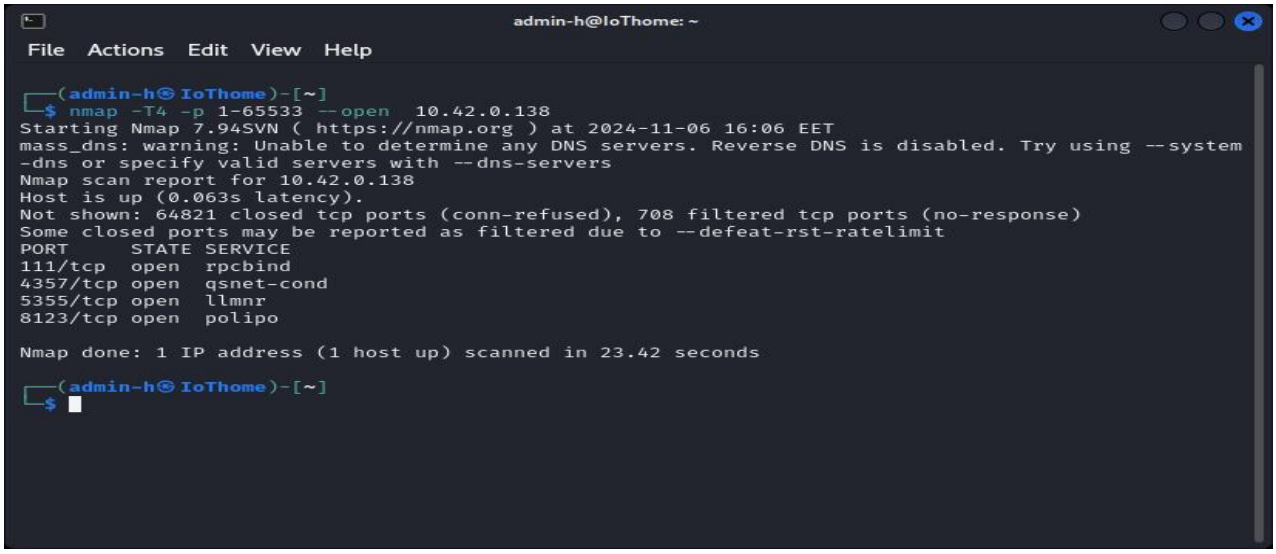


Рис. 3.5 Аналіз портів

Наведеній аналіз довів актуальність розробки модуля кіберзахисту з можливістю моніторингу та аудиту, як трафіку так і портів. Основою логіка роботи такого пристрою може бути функція аудиту безпеки сформуємо її.

Будемо позначати Gaud функцію аудиту безпеки. Тоді можна визначати що

$$Gaud = (Za1, Za2, \dots, Zan), \quad (3.1)$$

де Zai i=1,n i-тая складову аудиту яка залежить від відповідний загрози.

В якості прикладу побудови виміру такої функції оберемо аудит якості паролю.

Вимоги до якості паролю оберемо наступні:

Довжина паролю надійний пароль має складатися, щонайменше, з 18 символів. Чим довшим є пароль, тим він надійніший.

Складність паролю. Надійний пароль повинен складатися з комбінації великих та малих літер, цифр та спеціальних символів. Завдяки цьому збільшується кількість можливих комбінацій, а це, в свою чергу, значно ускладнює підбір чи злам паролю.

Унікальність. Кожен пароль повинен бути унікальним і використовуватись лише для одного облікового запису. Повторне використання паролю для доступу до кількох облікових записів збільшує ризик крадіжки даних, оскільки хакер зможе скористатися отриманими реєстраційними даними для доступу до кількох облікових записів.

Висока ентропія. Паролі з високою ентропією набагато важче підібрати або зламати, оскільки вони містять велику кількість можливих комбінацій.

Відсутність послідовних символів. Уникайте використання в паролях послідовних літер або цифр, таких як «abcd» або «1234», оскільки таку комбінацію легко підібрати. Хакери використовують високотехнологічні програми для визначення та зламу паролів, і ряд послідовних символів буде визначено за лічені секунди.

Цей перелік вад паролю не є вичерпним але дозволяє побудувати формулу оцінки складової аудиту пов’язаної з якістю паролю.

$$Zai = Pd * Pu * Pe * Pp, \quad (3.2)$$

де, відповідно Pd вірогідність якісної довжини; Pu вірогідність якісної унікальності; Pe вірогідність якісної ентропії; Pp вірогідність відсутності повторів.

Алгоритм використання модулю кіберзахисту з урахуванням цього буде мати вигляд:

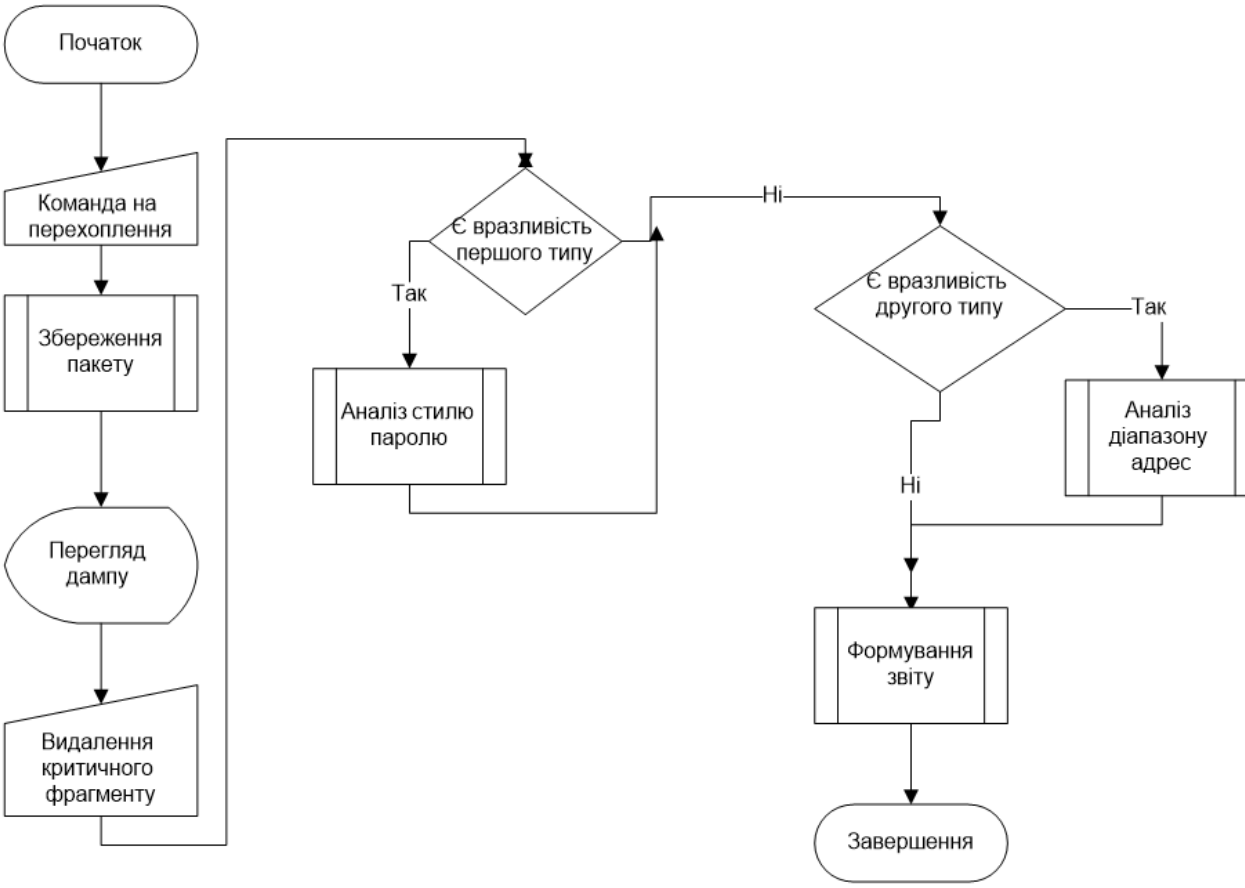


Рис. 3.6 Алгоритм використання модулю кіберзахисту

3.3. Реалізація апаратної частині модулю кіберзахисту

Оберемо апаратне рішення для центру, який буде все оброблювати, таким мозком буде Raspberry Pi — одноплатний комп'ютер.

Мікрокомп'ютер Raspberry Pi 5 Board 4GB – це п'яте покоління одноплатного комп'ютера, який розроблено для різноманітних застосувань у сфері комп'ютерної техніки та інтернету речей (IoT). Цей пристрій відзначається потужною апаратною платформою та різноманітними можливостями для розробки проектів.

Raspberry Pi 5 базується на оновленому процесорі і включає в себе різні порти та інтерфейси для підключення різноманітних пристроїв, таких як монітори, камери, сенсори і багато інших. Завдяки 64-розрядному чотирьох ядерному процесору Arm Cortex-A76, що працює на частоті 2,4 ГГц, Raspberry Pi 5 забезпечує збільшення продуктивності ЦП у 2-3 рази порівняно з Raspberry Pi 4.

Завдяки новому графічному процесору 800 МГц VideoCore VII GPU, значно підвищено графічну продуктивність (подвійний вихід дисплея 4Kp60 через HDMI), та підтримку камери від оновленого процесора сигналів Raspberry Pi Image Signal Processor.

Чіп контролера RP1 надає основну частину можливостей вводу-виводу для Raspberry Pi 5:

- загальна пропускна здатність USB збільшена більш ніж вдвічі, що забезпечує вищу швидкість передачі даних на зовнішні накопичувачі UAS та інші високошвидкісні периферійні пристрої;
- спеціальну двосмугову камеру та інтерфейси дисплея MIPI зі швидкістю 1 Гбіт/с, наявні в попередніх моделях, було замінено парою трансиверів MIPI з чотирма смугами 1,5 Гбіт/с, що втричі збільшує загальну пропускну здатність і підтримує будь-яку комбінацію до двох камер або дисплеїв;
- максимальна продуктивність SD-карти подвоюється завдяки підтримці високошвидкісного режиму SDR104;

• вперше платформа демонструє одноканальний інтерфейс PCI Express 2.0, що забезпечує підтримку периферійних пристроїв з високою пропускною здатністю.

Ця плата має вбудований бездротовий зв'язок Wi-Fi та Bluetooth, що полегшує підключення до мережі та інших пристроїв.

Raspberry Pi 5 підтримує різноманітні операційні системи та мови програмування, що робить його універсальним інструментом для розробників і творців. Він ідеально підходить для створення IoT-проектів, серверів, медіацентрів, емуляторів і багатьох інших застосувань, де потрібна надійна та потужна обчислювальна платформа. Raspberry Pi 5 – це зручний та доступний інструмент для реалізації технічних і креативних ідей.

Основні характеристики:

• Quad Arm Cortex-A76 на 2,4 ГГц

• Підтримка Cryptographic Extension (AES на апаратному забезпеченні)

• 512 Кб на ядро кешу L2

• 2 Мб кеша L3

• 4 ГБ LPDDR4X-4267 SDRAM

• Подвійний вихід дисплея HDMI 4kp60 із підтримкою HDR

• Декодер 4kp60 HEVC

• Графіка VideoCore VII з OpenGL-ES 3.1, Vulkan 1.2

• Процесор датчика зображення Raspberry Pi (ISP)

• Роз’єм Raspberry Pi для PCIe (1 порт 2.0, потрібна додаткова плата розширення)

• 802.11ac дводіапазонний Wi-Fi

• Bluetooth 5.0 (з підтримкою BLE)

• Гігабітний Ethernet

• Підтримка PoE (потрібна додаткова плата розширення)

• 2x USB 3.0 (з можливістю одночасної повної пропускної здатності)

• 2x USB 2.0

- Подвійні 4-канальні трансивери MIPI CSI/DSI, підтримка
- 2х дисплеїв; або
- 2х камер; або
- 1х дисплей + 1х камера
- 40-контактний роз’єм GPIO Raspberry Pi
- Роз'єм вентилятора
- Годинник реального часу (RTC)
- Роз'єм акумулятора RTC
- Кнопка живлення

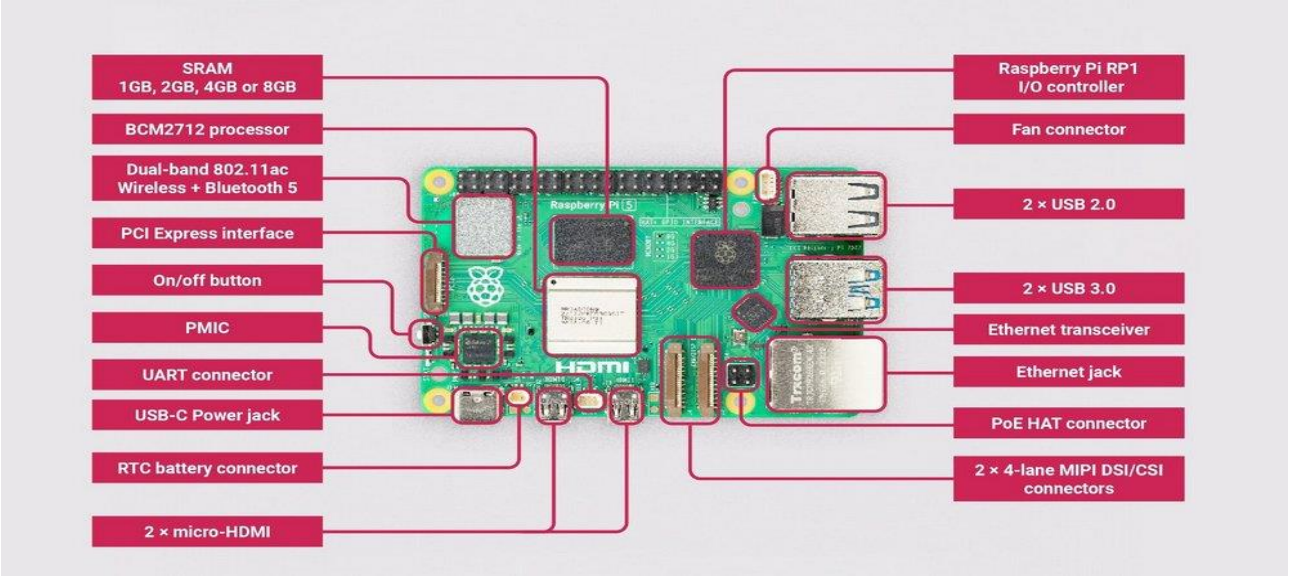


Рис 3.7 Схема розміщення елементів та конекторів

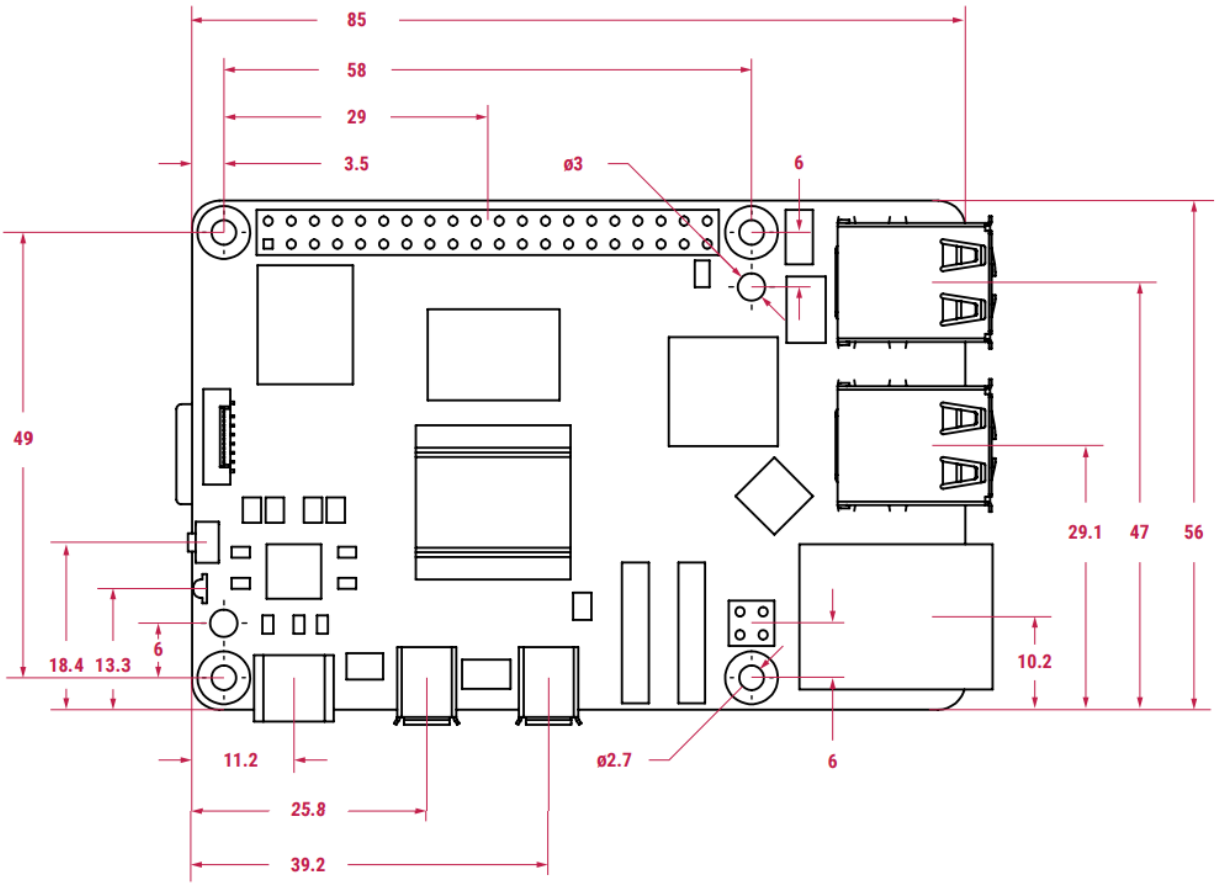


Рис 3.8 Схема розмірів Raspberry PI

Використання Raspberry Pi 5 як підсистеми керування розумного будинку має багато переваг:

1. Низька вартість: Raspberry Pi 5 доступний за вигідною ціною, що робить його привабливим вибором для домашнього автоматизаційного проєкту.

2. Гнучкість і розширюваність: Raspberry Pi 5 має велику кількість розширюваних портів, таких як GPIO, USB, HDMI та інші. Це дозволяє підключати до нього різні датчики, реле, камери та інші пристрої для контролю різних аспектів розумного будинку.

3. Низька споживана потужність: Raspberry Pi 5 витрачає невелику кількість електроенергії, що робить його економічним в експлуатації і дозволяє заощадити кошти на рахунках за електроенергію.

4. Зручне програмне забезпечення: Raspberry Pi 5 підтримує широкий спектр операційних систем, включаючи Raspbian, Ubuntu, а також спеціалізовані ОС для розумного будинку, які полегшують розробку та використання.

5. Спільнота та підтримка: Raspberry Pi має велику спільноту користувачів та розробників, які надають підтримку, поради та різноманітні готові рішення для автоматизації розумного будинку.

6. Надійність та стабільність: Хоча Raspberry Pi 5 є компактним і недорогим пристроєм, він відзначається стабільною роботою та надійністю, що робить його відмінним вибором для системи керування розумним будинком.

7. Можливості взаємодії з іншими пристроями: Raspberry Pi 5 може легко інтегруватися з іншими пристроями та платформами для створення комплексної системи розумного будинку.

8. Зручність у налаштуванні та управлінні: Завдяки своїй простоті у використанні та доступності різноманітних програмних інтерфейсів, Raspberry Pi 5 дозволяє легко налаштувати та керувати різними аспектами розумного будинку з будь-якого пристрою з підтримкою мережі.

Висновок до розділу 3

Проведено аналіз ієрархічної структурі телекомунікаційної мережі "інтелектуального будинку".

Визначено типові загрозі та розроблено функцію аудиту, на основі якої запропоновано алгоритм використання до модулю кіберзахисту.

Запропоновано апаратне рішення модулю.

РОЗДІЛ 4
ПРОГРАМНА РЕАЛІЗАЦІЯ ТА ТЕСТУВАННЯ МОДУЛЮ КІБЕРЗАХИСТУ

4.1 Вибір середовища реалізації

Для розробки соціалізованих блоків програмного забезпечення модуля кіберзахисту було обрано середовище реалізації:

PyCharm 2024.2.1 (Community Edition)

Build #PC-242.21829.153, built on August 29, 2024

Runtime version: 21.0.3+13-b509.11 amd64 (JCEF 122.1.9)

VM: OpenJDK 64-Bit Server VM by JetBrains s.r.o.

Toolkit: sun.awt.windows.WToolkit

Windows 11.0

GC: G1 Young Generation, G1 Concurrent GC, G1 Old Generation

Memory: 1500M

Cores: 12

Registry:

ide.experimental.ui=true

i18n.locale=

Azure App Service Tools v3.0.0 17.11.231.19466

Azure App Service Tools v3.0.0

Azure Functions and Web Jobs Tools 17.11.231.19466

Azure Functions and Web Jobs Tools

C# Tools 4.11.0-3.24428.4+1ea9c390a5bb6815fdff2137ee155e23e78d6ff3

Для C# компонент, що використовуються в IDE, може використовуватися різна версія компілятора. Це залежить від типу проєкту та налаштувань.

Common Azure Tools 1.10 - Надає загальні послуги для використання Azure Mobile Services and Microsoft Azure Tools.

GitHub Copilot 0.2.1648.49400

GitHub Copilot це сервіс AI , який допомагає писати код швидше та з меншими витратами зусиль.

Microsoft JVM Debugger 1.0 - Забезпечує підтримку підключення налагоджувача Visual Studio до віртуальних машин Java, сумісних з JDWP

NuGet Package Manager 6.11.0

Менеджер пакетів NuGet у Visual Studio. Для отримання додаткової інформації про NuGet відвідайте https://docs.nuget.org/

Razor (ASP.NET Core) 17.11.3.2442001+68650a7d94261bc56a1f4bc522c2ee35314b1abb

Надає мовні послуги для ASP.NET Core Razor.

SQL Server Data Tools 17.11.38.0

Microsoft SQL Server Data Tools

Test Adapter for Boost.Test 1.0 -

Включає інструменти тестування Visual Studio з модульними тестами, написаними для Boost.Test.

Test Adapter for Google Test 1.0

Включає інструменти тестування Visual Studio з модульними тестами, написаними для Google Test.

TypeScript Tools 17.0.30715.2002

TypeScript Tools for Microsoft Visual Studio

Visual Basic Tools 4.11.0-3.24428.4+1ea9c390a5bb6815fdff2137ee155e23e78d6ff3

Visual F# Tools 17.11.0-beta.24421.7+af2f522de602281d4ef5a7b71507c428e814c5c1

Microsoft Visual F# Tools

Visual Studio IntelliCode 2.2

AI- помічник для розробки в Visual Studio.

В якості основи реалізації обрана платформа home-assistant.io. Розглянемо її більш детально.

4.2 Програмна реалізація модуля

Базовою системою для модуля кіберзахисту буде платформа home-assistant.io

Home Assistant – це платформа з відкритим кодом для домашньої автоматизації, яка може використовуватися для керування різними пристроями та системами у вашому домі. Її можна встановити на Raspberry Pi 5, що робить її доступним і потужним рішенням для домашньої автоматизації.

Деякі з основних функцій та можливостей Home Assistant на Raspberry Pi 5:

- підтримує широкий спектр пристроїв, включаючи розумні лампочки, термостати, датчики, замки, камери та багато іншого.

- дозволяє створювати автоматизації, які автоматично керують вашими пристроями. Наприклад, ви можете створити автоматизацію, яка вимикає світло, коли ви залишаєте кімнату, або автоматизацію, яка вмикає кондиціонер, коли температура піднімається вище певного рівня.

- дозволяє створити власну панель керування для керування вашими пристроями. Ця панель керування може бути доступна з будь-якого веб-браузера або мобільного пристрою.

- підтримує голосове керування за допомогою Google Assistant, Amazon Alexa та Siri.

- може працювати локально, без підключення до хмари. Це забезпечує більшу конфіденційність та безпеку.

Попереднє налаштування модуля AppArmor для забезпечення взаємодії з модулем кіберзахисту

По замовчанню підключені тільки базові елементи AppArmor (варіант Ubuntu Server) набір профілів застосунків необхідне додавати окремо.

```
[admin@server ~]$ sudo aptitude install apparmor-utils apparmor-profiles
```

Далі перевіряємо статусі

```
[admin@server ~]$ sudo apparmor_status
apparmor module is loaded.
31 profiles are loaded.
31 profiles are in enforce mode.
 /snap/snapd/10492/usr/lib/snapd/snap-confine
 /snap/snapd/10492/usr/lib/snapd/snap-confine//mount-namespace-capture-helper
 /usr/bin/man
 /usr/lib/NetworkManager/nm-dhcp-client.action
 /usr/lib/NetworkManager/nm-dhcp-helper
 /usr/lib/connman/scripts/dhclient-script
 /usr/lib/snapd/snap-confine
 /usr/lib/snapd/snap-confine//mount-namespace-capture-helper
 /usr/sbin/tcpdump
   ...
0 profiles are in complain mode.
0 processes have profiles defined.
0 processes are in enforce mode.
0 processes are in complain mode.
0 processes are unconfined but have a profile defined.
```

Як ми бачимо з лістингу модуль завантажено. 31 профіль завантажено та знаходяться в enforce моді. Enforce — забезпечення тобто профілі готові до активної взаємодії. Це підтверджується прикінцевим повідомленням про їх вільний стан та відсутність процесів які очікують вільні профілі.

В останних рядках згадуються– Enforce і Complain.

У режимі Enforce ядро забезпечує дотримання правил, записаних у файлі профілю. Порушення не допускаються і відповідний запис вноситься в журнали.

У режимі **Complain** AppArmor реєструє лише порушення, не блокуючи самі дії.

Повний зміст пакету apparmor-profiles розміщено за адресою /usr/share/apparmor/extra-profiles/

Там є більше сотні готових профілів.

```
[admin@server ~]$ ll /usr/share/apparmor/extra-profiles/ |head
total 484
-rw-r--r-- 1 root system 1724 May 19 2020 README
drwxr-xr-x 3 root system 4096 Dec 8 10:14 abstractions/
-rw-r--r-- 1 root system 1319 May 19 2020 bin.netstat
-rw-r--r-- 1 root system 1815 May 19 2020 etc.cron.daily.logrotate
-rw-r--r-- 1 root system  948 May 19 2020 etc.cron.daily.slocate.cron
-rw-r--r-- 1 root system  722 May 19 2020 etc.cron.daily.tmpwatch
-rw-r--r-- 1 root system 2623 May 19 2020 sbin.dhclient
[admin@server ~]$ ll /usr/share/apparmor/extra-profiles/ |wc -l
118
```

Перш ніж профіль стане активним, вам потрібно перемістити його з папки /usr/share/apparmor/extra-profiles/ до /etc/apparmor.d/. Тепер ми можемо вивчити його і змінити за необхідністю. Візьмемо щось простіше, наприклад /etc/apparmor.d/bin.ping.

```
#include <tunables/global>
profile ping /{usr/,}bin/{,iputils-}ping flags=(complain) {
  #include <abstractions/base>
  #include <abstractions/consoles>
  #include <abstractions/nameservice>
  capability net_raw,
  capability setuid,
```

```
  network inet raw,
  network inet6 raw,
  /{,usr/}bin/{,iputils-}ping mixr,
  /etc/modules.conf r,
  # Site-specific additions and overrides. See local/README for details.
  #include <local/bin.ping>
}
```

Все досить зрозуміло, якщо не брати до уваги прапорців міксера. Опис значень прапорів наведено нижче:

• p — зчитування;

• W - запис

• a — інкрементний запис в кінці файлу, з англійського append;

• k - Блокування файлів

• l — створення символьних посилань на виконувані файли;

• m — завантаження виконуваних файлів у пам'ять;

• CX — перехід на профіль нижчого рівня при виконанні

• Cx - Перехід до профілю нижчого рівня при виконанні з очищенням змінних середовища.

• ix — успадкування продуктивності;

• px — потрібен дискретний профіль безпеки для ресурсу.

• Px — вимагає дискретний профіль безпеки для ресурсу, очищає змінні середовища.

• UX — не перевіряйте запуск нових процесів;

• Ux: не перевіряти запуск нових процесів і не очищати змінні оточення;

Можно вказати Capabilities ядра Linux, які дозволено використовувати процесу.

Щоб переключитися з режиму тренування в форсований, потрібно виконати команду aa-enforce, <prog_name>return - aa-complain <prog_name>.

Якщо після включення примусового режиму ping спробує щось зробити, AppArmor заблокує це.

```
[admin@server ~]$ sudo aa-enforce ping
Setting /usr/bin/ping to enforce mode.
[admin@server ~]$ sudo cp /usr/bin/man /usr/bin/ping
[admin@server ~]$ /usr/bin/ping ping
/usr/bin/ping: can't open the manpath configuration file /etc/manpath.config
```

Якщо нам потрібно створити новий профіль, то це не складе труднощів. Спочатку потрібно створити шаблон за допомогою команди aa-autodep, а потім заповнити його за допомогою aa-genprof.

Наступній приклад взято з сайту /etc/apparmor.d/usr.sbin.tcpdump on Ubuntu 9.04:

```
#include <tunables/global>

/usr/sbin/tcpdump {
  #include <abstractions/base>
  #include <abstractions/nameservice>
  #include <abstractions/user-tmp>

  capability net_raw,
    capability setuid,
  capability setgid,
  capability dac_override,
  network raw,
  network packet,

  # for -D
```

```
  capability sys_module,
  @{PROC}/bus/usb/ r,
  @{PROC}/bus/usb/** r,

  # for -F and -w
  audit deny @{HOME}/.* mrwkl,
  audit deny @{HOME}/.*/ rw,
  audit deny @{HOME}/.*/** mrwkl,
  audit deny @{HOME}/bin/ rw,
  audit deny @{HOME}/bin/** mrwkl,
  @{HOME}/ r,
  @{HOME}/** rw,

  /usr/sbin/tcpdump r,
}
```

Цей фрагмент потрібен для tcpdump та демонструє кілька властивостей AppArmor:

Профілі є простими текстовими файлами

Коментарі підтримуються в профілі

Абсолютні шляхи, а також замінник файлів можуть використовуватися при визначенні доступу до файлів.

Присутні різні елементи керування доступом до файлів. У профілі ми бачимо 'r' (читання), 'w' (запис), 'm' (карта пам'яті як виконуваний), 'k' (блокування файлів) і 'l' (створення жорстких посилань). Є й інші, які не продемонстровані в цьому профілі, включаючи (але не обмежуючись цим) 'ix' (виконувати та успадковувати цей профіль), 'Px' (виконувати під іншим профілем, після очищення середовища) та 'Ux' (виконувати без обмежень, після очищення середовища)

Контроль доступу для можливостей присутній

Присутній елемент керування доступом для роботи в мережі специфічність у відповідності правил, тобто найконкретніші збіги правил (наприклад, доступ до @{HOME}/bin/bad.sh відмовляється при аудиті через 'audit deny @{HOME}/bin/** mrwkl', навіть якщо загальний доступ до @{HOME} дозволено з '@{HOME}/** rw,')

В першу чергу нам потрібні функції аудиту які будуть відтворені в нашому модулі кіберзахисту.

Деякі з обмежень використання Home Assistant на Raspberry Pi 5:

- Налаштування Home Assistant може бути складним завданням для користувачів без досвіду роботи з технологіями.

- Оскільки Home Assistant – це платформа з відкритим кодом, офіційна підтримка обмежена.

- Продуктивність Home Assistant на Raspberry Pi 5 може бути обмеженою кількістю пристроїв та автоматизацій, які ви використовуєте.

В якості прикладу розглянемо керування простим датчіком присутности с апаратної індикацією (світлодіод).

Створимо новий файл для програми на Пітон і відкриємо його в редакторі nano:

Скопіюємо наведений нижче код у файл:

```
#!/usr/bin/env python
# Ілюстрація роботи в середовищі с апаратним сенсором sensor MQ135 + ADC-DAC PCF8591P

import os
import time
from smbus import SMBus

DEV_ADDR = 0x48
adc_channel = 0b1000010 # 0x42 (input AIN2 for ADC + use DAC)
```

```
dac_channel = 0b1000000 # 0x40

bus = SMBus(1)          #1 – адреса шини I2C для RPi rev.2

while(1):
    os.system('clear')
    print("Натисні Ctrl C для виходу...\n")
    # читати значення датчика з АЦП
    bus.write_byte(DEV_ADDR, adc_channel)
    bus.read_byte(DEV_ADDR)
    bus.read_byte(DEV_ADDR)
    value = bus.read_byte(DEV_ADDR)
    print 'AIN value = ' + str(value)
    # порівняти значення з АЦП і встановлене значення в ЦАП
    if value > 120:
        bus.write_byte_data(DEV_ADDR, dac_channel, 220)
    else:
        bus.write_byte_data(DEV_ADDR, dac_channel, 0)
    # пауза 100 мілісекунд
    time.sleep(0.1)
```

На цьому кроці на простому прикладі ми впевнилися в коректному підключенні заліза та вірному налаштуванні програмної платформі.

Наступним кроком побудуєм тестовий приклад для тестування запитів типу GET та їх подальшої інтерпретації. В приклади використана стандартна тестова адреса http://example.com/ після доступу до якої перевіряється коректність виконання запиту.

```
"""Tests for our Mod_Subsec"""

import pytest
```

```
from .aiohttp import AiohttpClientMocker

async def test_matching_url() -> None:
    """Тестування коренисті читання  urls."""
    mocker = AiohttpClientMocker()
    mocker.get("http://example.com")
    await mocker.match_request("get", "http://example.com/")

    mocker.clear_requests()

    with pytest.raises(AssertionError):
        await mocker.match_request("get", "http://example.com/")

    mocker.clear_requests()

    mocker.get("http://example.com?a=1")
    await mocker.match_request("get", "http://example.com/", params={"a": 1, "b": 2})
```

Впевнившись в працездатності апаратної частини перейдемо к тестуванню програної частині.

Тестування версії релізу

```
    # Всі релізи s
    def test_11_heap_protector(self):
        ''' heap protection'''

        self.assertShellExitEquals(0, ["./heap", 'safe'])
        self.assertShellExitEquals(-6, ["./heap", 'unsafe'])
```

Повертає від’ємне число в відпадку неактуальної версії

Перевірка процесору, що використовується

```
expected = 0
jb_pc = '-1'
jb_unenc = '-1' # unencrypted isn't stable
if self.dpkg_arch == 'i386' or self.dpkg_arch == 'lpia':
   jb_pc = '5'
   #jb_unenc = '1'
elif self.dpkg_arch == 'amd64':
   jb_pc = '7'
   #jb_unenc = '0'
elif self.dpkg_arch == 'armel':
   self._skipped("not on ARM")
   expected = 2
```

Ці фрагменті ілюструють простоту перевірки але побудові повної перевірки таких треба де кілька тисяч.

4.3. Тест функції перехоплення трафіку

Для тестування функції перехоплення трафіку зберемо тестовий стенд.

Під час експериментів на Raspberry Pi 5 додатково було установлено дисплей для індикації дій контролеру. Фото наведено на рис. 4.2.

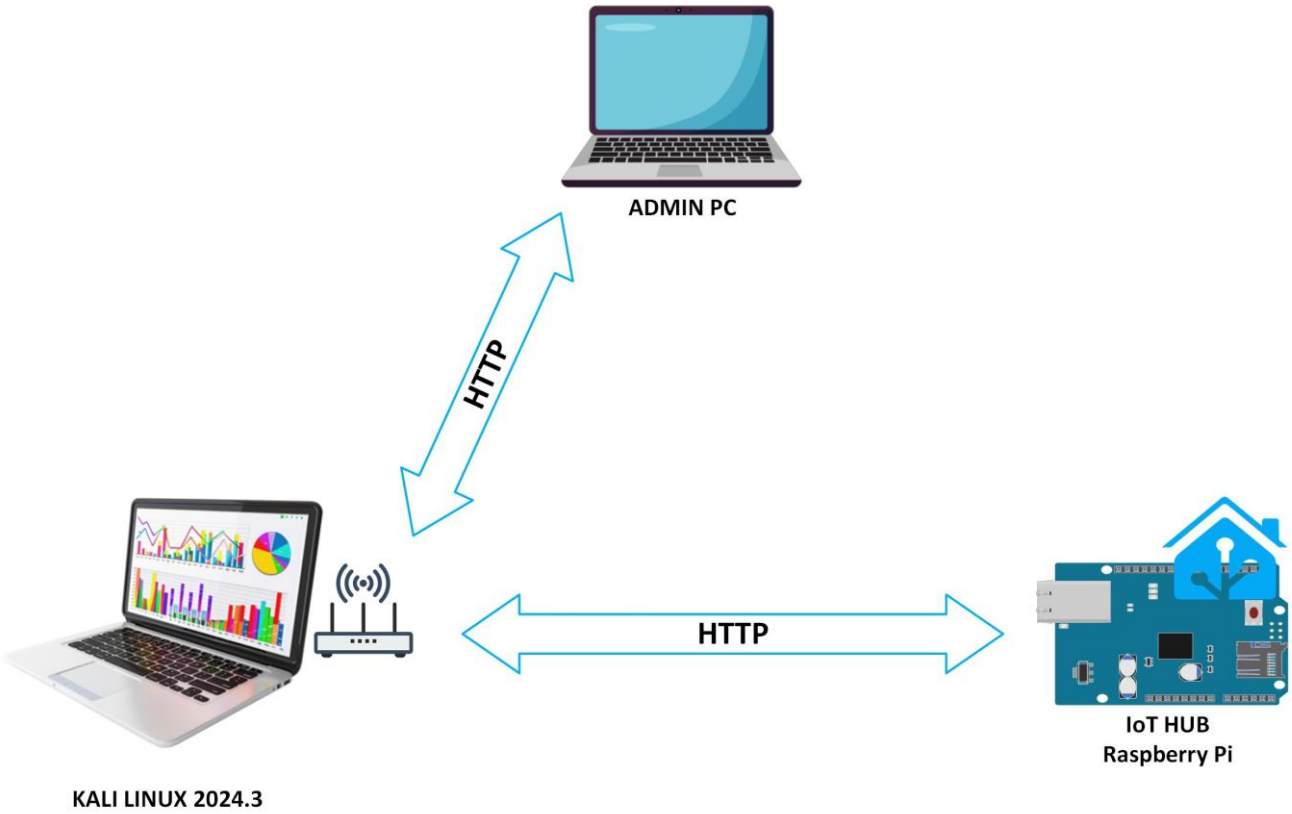


Рис 4.1. Стенд для перехоплення та аналізу мережевого трафіку

Рис 4.2. Фото Raspberry Pi 5 с модулем зовнішній індикації якій було використано в якості IoT хабу

Для створення точки доступу на ПК або ноутбуці мережева карта повинна підтримувати режим роботи AP. Перевіряємо за допомогою команди iw list.

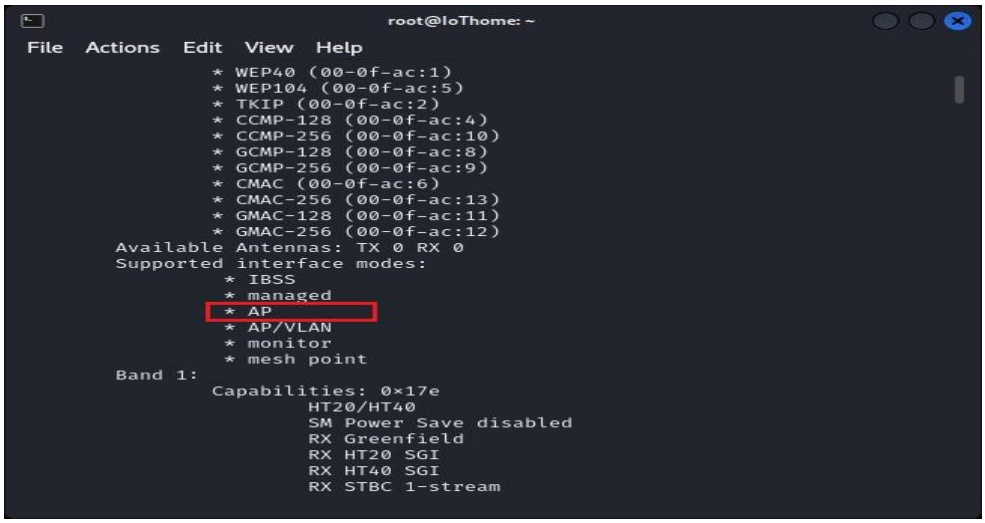


Рис 4.3. Перевірка режиму роботи мережевої карти

Далі необхідно налаштувати точку доступу, це зручніше всього зробити за допомогою графічного інтерфейсу Network Manager. Хоча можна і за допомогою команд.

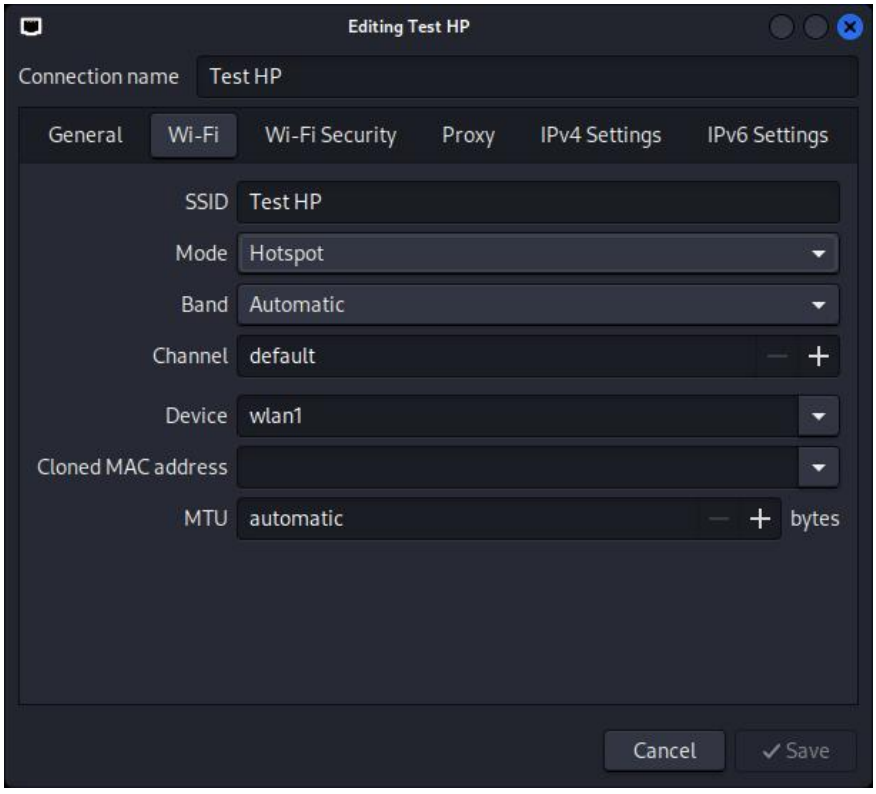


Рис 4.4 Налаштування точки доступу

Здійснено далі захоплення та аналіз трафіку. Підзапуску якої необхідно вибрати мережевий інтерфейс, який налаштований як точка доступу.

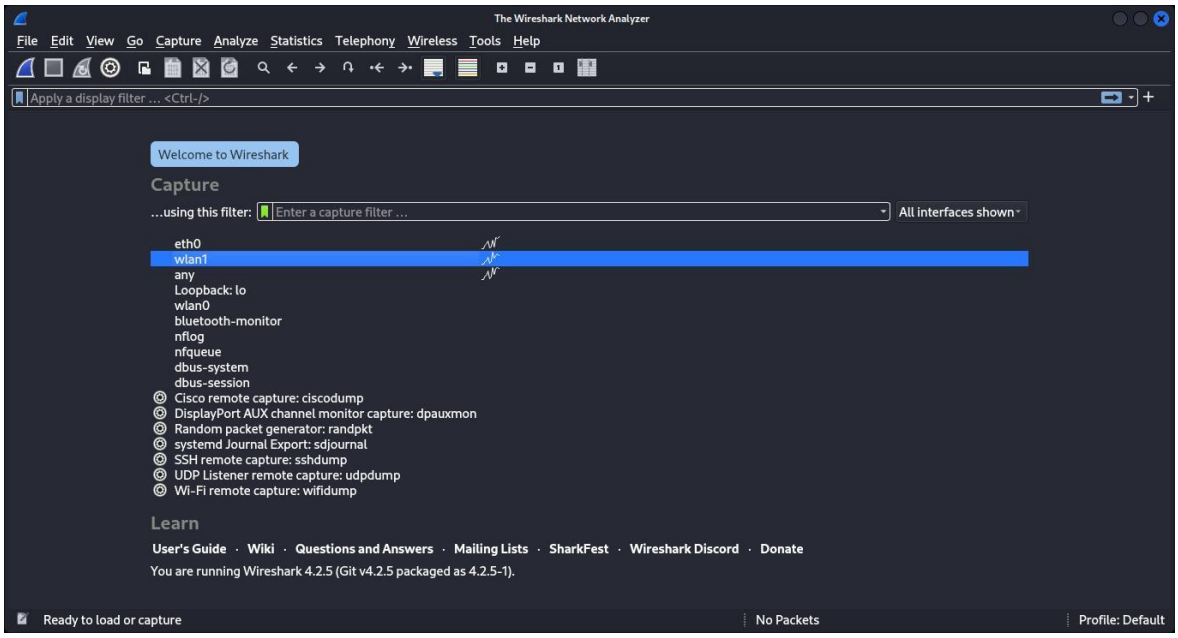


Рис 4.5 Вибір мережевого інтерфейсу.

Захоплення трафіку почнеться в автоматичному режимі. Якщо це не сталося потрібно в Головному меню вибрати Capture->Start або Ctrl+E.

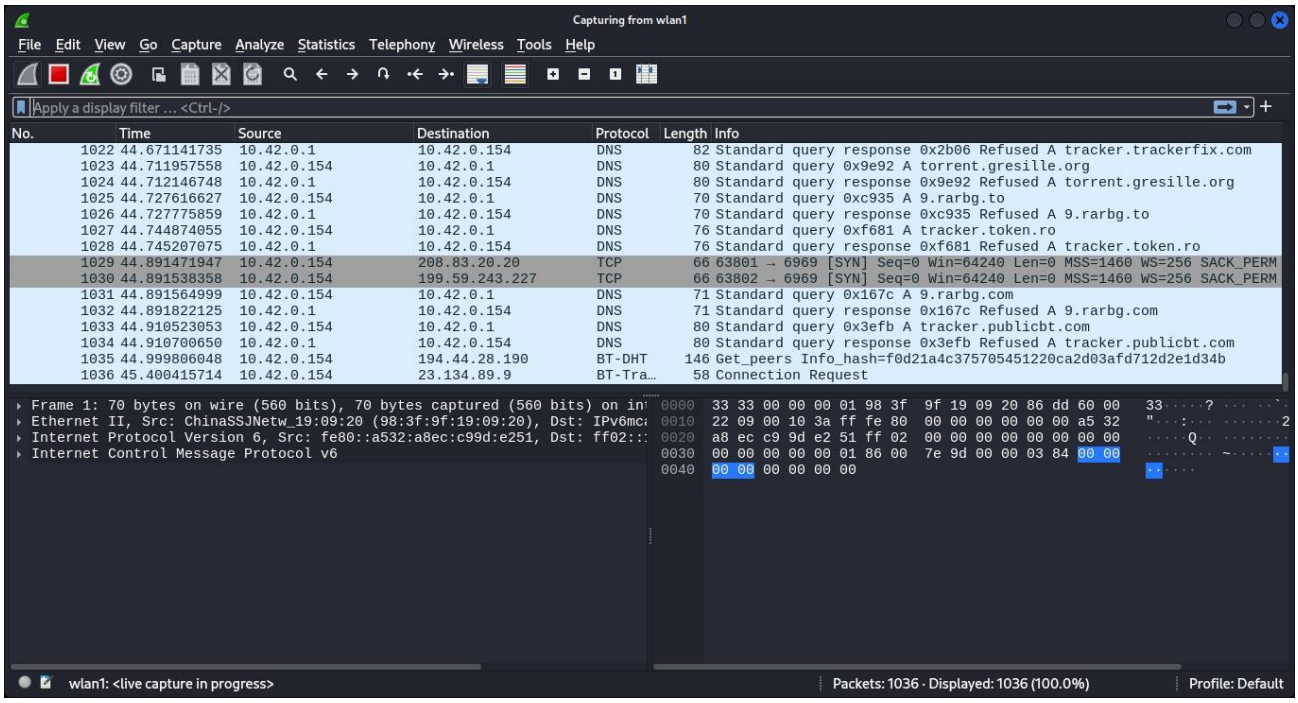


Рис 4.6 Приклад захоплення трафіку

Для припинення захоплення в потрібно в Головному меню вибрати Capture->Stop або Ctrl+E. Після чого можна зберегти захоплений трафік в файл .pcapng.

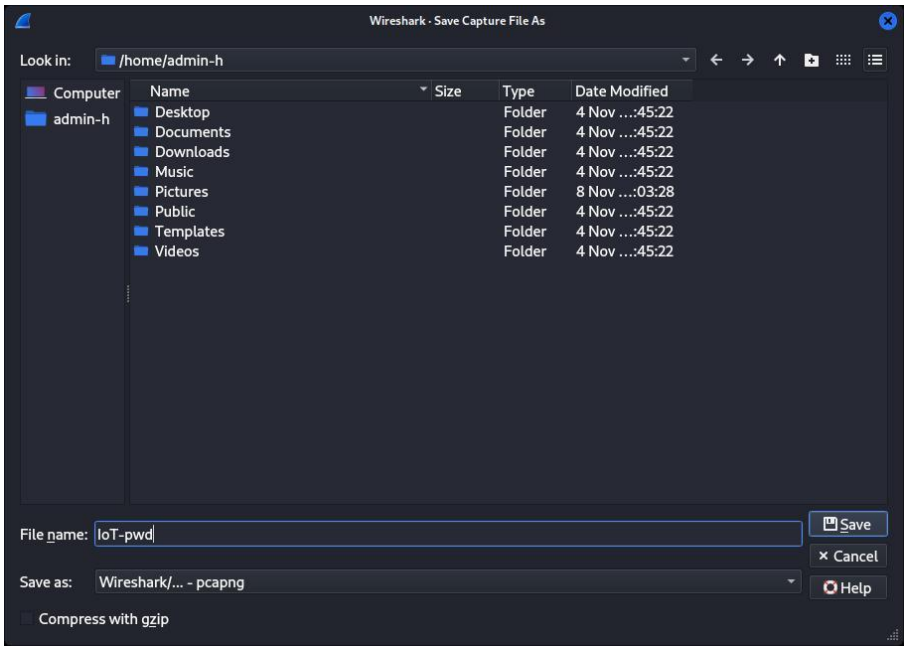


Рис 4.7 Збереження трафіку

Далі відкриваємо аналізатор протоколів і бачимо велику кількість пакетів. Нас цікавлять конкретні пакети, які містять POST-дані, що формуються на нашій локальній машині при заповненні форми на екрані та надсилаються на віддалений сервер після натискання кнопки «Вхід» або «Авторизація» в браузері.

Аналіз паролю

Для цього необхідно використати спеціальний фільтр для відображення захоплених пакетів

http:request.method==”POST”

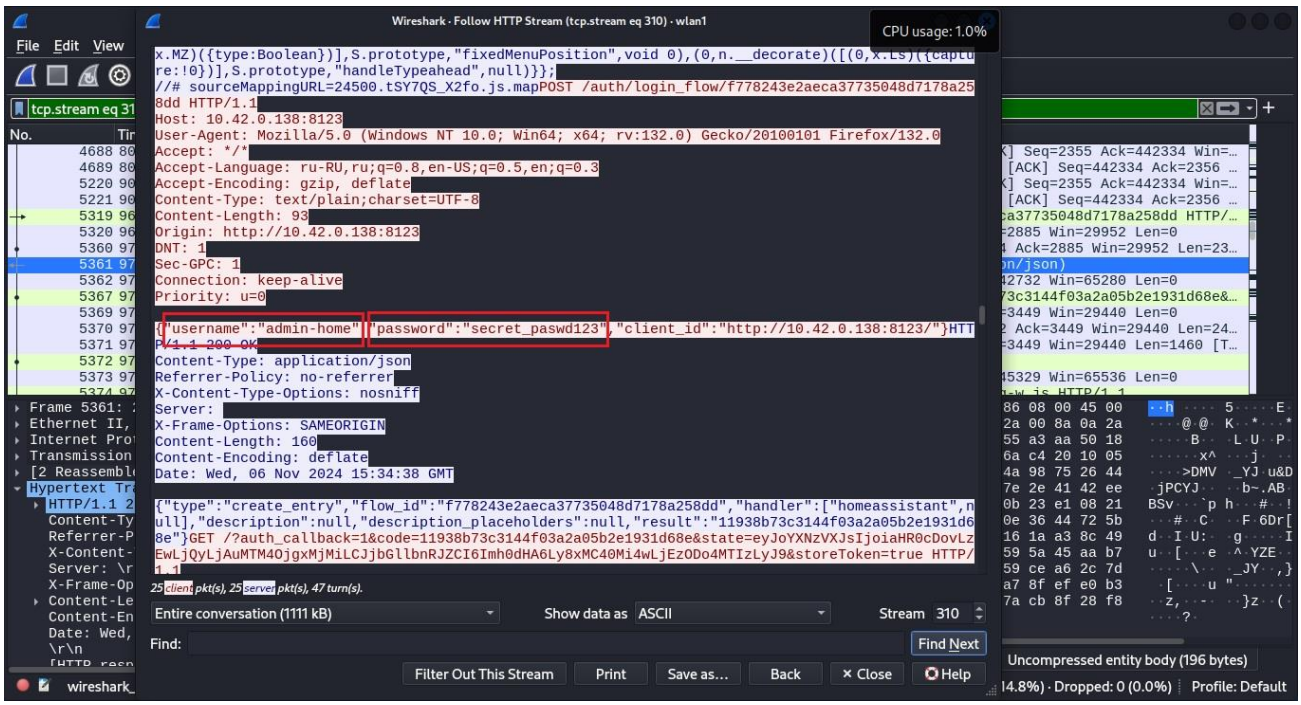


Рис 4.8 Пошук паролю в перехопленому трафіку.

Аналіз адресі

Оскільки Home Assistant за замовчуванням працює на 8123 порту, то за допомогою фільтру можна tcp.port == 8123 and http, можна відфільтрувати потрібний нам трафік. Клієнт: це пристрій, який надсилає GET-запит. GET-запит буде ініційовано з клієнтського пристрою (наприклад, вашого браузера або мобільного пристрою).

На малюнку IP клієнта 10.42.0.154. виділено червоним, а IP сервера(Home Assistant) 10.42.0.138 зеленим кольором.

Після того, як клієнт надішле GET-запит, сервер (наприклад, Home Assistant) повинен надіслати відповідь (наприклад, HTML-сторінку або JSON-дані). Відповідь виглядатиме як пакет з методом HTTP/1.1 200 OK і міститиме дані, які сервер надсилає назад клієнту.

У нашому випадку сервер 10.42.0.138 надсилає HTTP 200 OK відповідь на запит клієнта 10.42.0.154.

Порівняльна характеристика засобів оцінювання показано в Табл. 4.1.

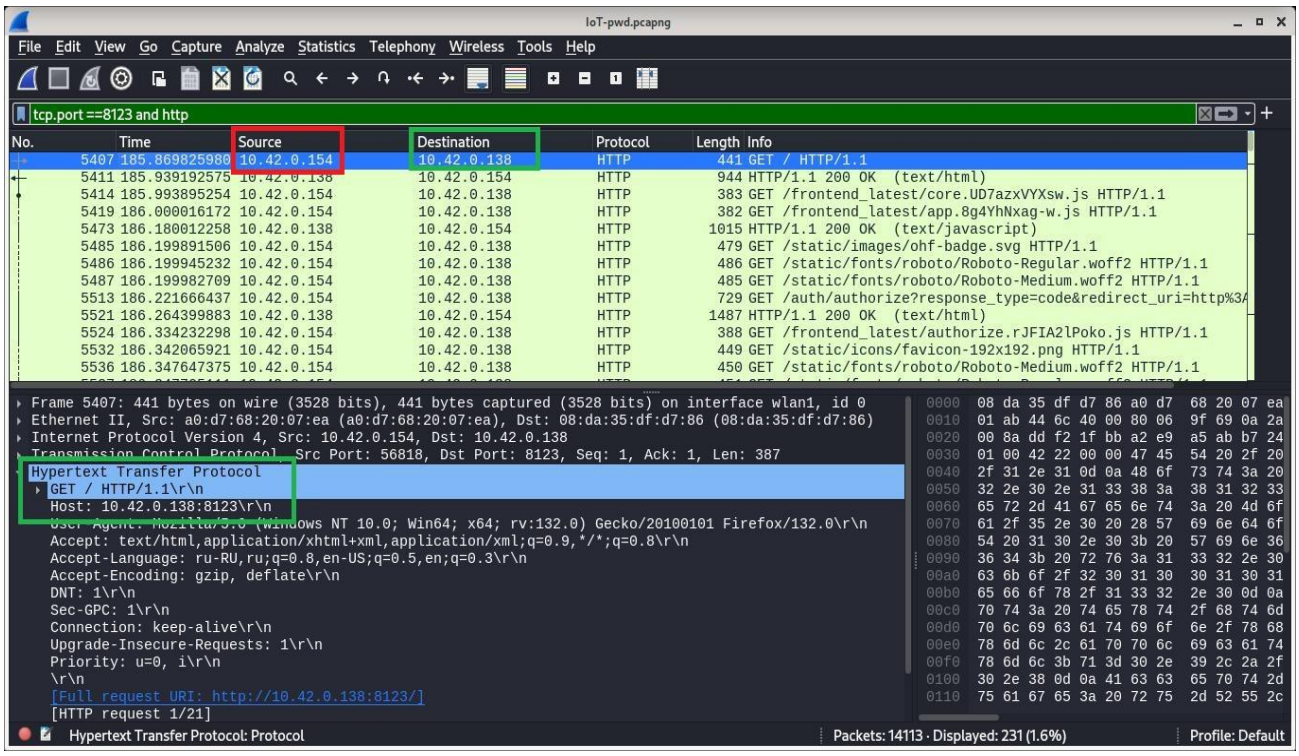


Рис. 4.9 Відзначення адреси

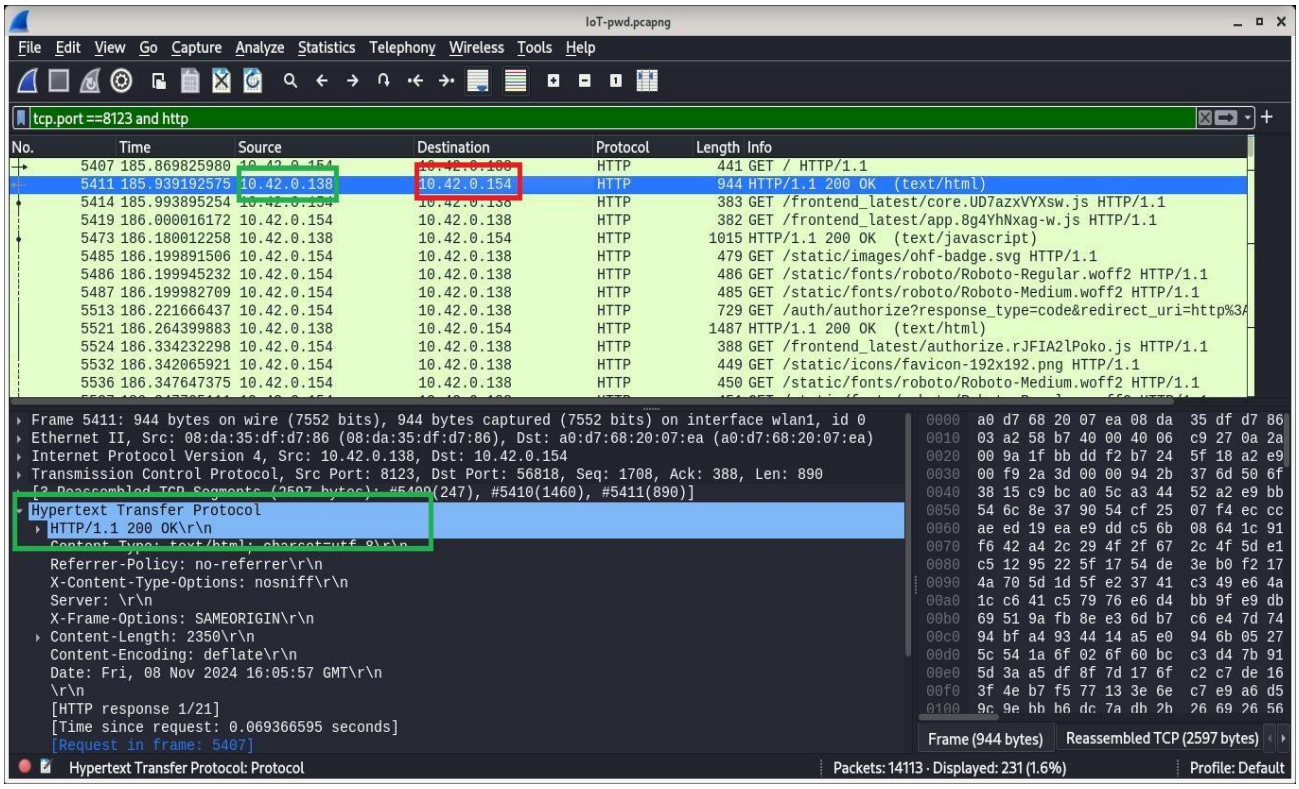


Рис. 4.10. Назва ілюстрації

Таблиця 4.1

Порівняльна характеристика засобів оцінювання, що вже існують

Характеристика	Ajax Systems	BroadLink	Fibaro	Orvibo	Xiaomi	МК
Доступність	+	+	+	+	+	-
Інтерфейс	+	-	-	-	+	+
Масштабов-ність	+	+	+	+	+	+
Автономність	+	-	+	-	-	+
Гнучкість	+	-	-	-	-	+
Сумісність	-	+	-	-	+	+
Аудит	-	-	-	-	-	+

Висновок до розділу 4

Обґрунтовано вибір програмного середовища та платформі.

Розроблено програмну складову модуля кібербезпеки.

Проведено тестування розробленого модуля кібербезпеки розумного будинку, що дало змогу довести коректність його роботи.

ВИСНОВКИ

Інформаційні технології навколо нас. Це твердження має багато аспектів. В першу чергу прогрес та радість нових можливостей. Це холодильник, який знає, що і де придбати тобі на вечерю. Але, це і зовнішнє спостережене за нашими діями, це і добре, як медик спостерігає за хворою людиною і не зовсім, коли хтось бачить наші маленькі таємниці. Це можливість миттєвого переказу грошей за нашими рахунками і миттєва втрата накопичень, як внаслідок атаки хакера. Інформаційні технології - це тільки інструмент, корисне використанння якого залежить тільки від навичок користувача. Якщо кібергігієні та кіберзахисту приділяється достатньо уваги, то вірогідність шкоди від інформаційних технологій мінімальна. Виконання функцій кіберзахисту теж можна доручити комп'ютеру. Але важливо знати, які функції і як він має виконувати саме в том середовищі, де він функціонує. Тому, тема кваліфікаційної роботи «Модуль кіберзахисту системи керування розумного будинку» є актуальною.

В кваліфікаційній роботі ставилось за мету розробити модуль кібербезпеки розумного будинку, який, за рахунок забезпечення аудиту, підвищує загальний рівень безпеки використання інформаційних технологій розумного будинку.

В результаті виконання кваліфікаційної роботи отримано наступні результати:

1. Проаналізовані існуючі інформаційні технології в інтелектуальному будинку, та на основі результату проведеного аналізу, визначено функціональні вимоги щодо модулю кіберзахисту системи керування розумного будинку.

2. Розроблено архітектуру модулю кіберзахисту в аспектах топології, формату та політик підключення, а саме обрана хмарна топологія підключення користувача, візначено формат підключення модуля кіберзахисту шляхом розширення та поєднання з WiFi роутером, політика підключення реалізується модулем AppArmor в середовище Home Assistans.

3. Розроблено апаратну та програмну складові модуля кібербезпеки. На основі мікрокомп'ютеру Raspberry Pi 5 Board 4GB та середовища Home Assistans

побудоване комплексне рішення, що забезпечує функціонування застосунку AppArmor та модулю кіберзахисту, який виконує фукції аудиту шляхом переверки слабкості налаштувань застосунку.

4. Проведено тестування розробленого модуля кібербезпеки розумного будинку, що дало змогу довести коректність його роботи.

В першому розділі проведено аналіз інформаційних технологій, які застосуються в розумному будинку. А саме, визначені базові поняття інформаційних технології в інтелектуальному будинку. Виконано аналіз ключових виробників систем розумних будинків. Розглянуто основні напрями автоматизації розумного будинку. Визначено функціональні вимоги щодо модулю кіберзахисту системи керування розумного будинку.

В другому розділі проведено аналіз і побудована архітектура топології, формату та політик підключення модулю кіберзахисту. А саме, здійснено вибір технології та топології підключення модулю кіберзахисту. Виконано вибір варіанту підключення модулю кіберзахисту. Зроблено аналіз архітектури безпеки при керування розумним будинком - в другому розділі.

В третьому розділі, визначено вимоги та апаратна реалізація модулю кіберзахисту, аналіз вимог щодо модулю кіберзахисту. Зроблено аналіз атак на розумний будинок. Здійснена реалізація апаратної частині модулю кіберзахисту.

В четвертому розділі, здійснена програмна реалізація та тестування модулю кіберзахисту. А саме, обрано середовище реалізації. Виконана програмна реалізація модуля. Протестовано функції перехоплення трафіку.

Наукова новизна - запропонована модель аудиту, яка за рахунок контролю імовірності реалізації критичних вразливостей дозволяє підвищити рівень безпеки використання інформаційних технологій в розумному будинку.

Практичне значення отриманих результатів. Розроблено модуль кібербезпеки розумного будинку, який рекомендується використовувати під час аудиту кібербезпеки розумного будинку, що за рахунок виявлень слабкості налаштувань шляхом перевірок, дозволить покращити загальний рівень безпеки.

Особистий внесок здобувача вищої освіти. Результати кваліфікаційної роботи отримані автором особисто.

Апробація отриманих результатів. Основні положення роботи доповідалися та обговорювалися на міжнародної науково-практичної конференції: Innovations and New Directions in Scientific Research: Proceedings of the International Scientific Conference (2024, October 14). Manchester, UK: Bookmundo.

СПИСОК БІБЛІОГРАФІЧНИХ ПОСИЛАНЬ ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Ajax Home Security — Alarms & Smart Systems | Official website https://ajax.systems/

2. Broadlink Ukraine - ведущий поставщик решений для умного дома и домашней автоматизации https://broadlink.com.ua/

3. Wireless Smart Home and Home Automation | FIBARO https://www.fibaro.com/en/

4. ORVIBO- Your Smart Home https://www.orvibo.com/en/

5. Xiaomi Україна | Офіційний сайт Xiaomi https://www.mi.com/ua/

6. Аязбай А.Є. Система голосового управління на одноплатному ARM-мінікомп'ютері / Аязбай А.Є., Конуркульжин Д.А., Орінбай А.А. - Вища школа Казахстану. - 2014. - 189 с.

7. Васьковська В.П. Механізм забезпечення права людини на безпеку/Васьковська В.П. - Київ, 2021 - 28 с.

8. Васьковська В.П. Поняття та характерні риси безпеки людини /Васьковська В.П., 2018 – 184 с.

9. Васьковська В.П. Проблеми визначення категорії «безпеки» та її роль у правознавстві /Васьковська В.П., 2021 – 62 с.

10. Галицький А. В. Захист інформації у мережі. Аналіз технологій та синтез рішень / Галицький О. В., Рябко С. Д., Шаньгін В. Ф., 2018. - 456 с.

11. Горбачов Г.М. Промислова електроніка: Підручник для вузів / Горбачов Г.М., Чаплігін Є.Є., 2020. - 320 с.

12. Долгий А. А. Проблеми правового регулювання охоронної діяльності податкових органів / Регіональні проблеми боротьби з економічною злочинністю. - Харків, 2021. - 209 с.

13. Зайцев І. Врятуй та збережи / Російський діловий тижневик «Контракти».- № 12.- 21 березня, 2010. - 18 с.

14. Казначеєв В. Мікросхеми для керування електродвигунами. - Київ: Додека, 2013. - 288 с.

15. Калінін Г. Недержавна служба безпеки: поточний момент та перспективи / Служба безпеки. – 2021. – 19 с.

16. Дужак І.О. Розумний будинок. Автоматизація технологічних і бізнеспроцесів. Одеська національна академія харчових технологій. 2013. №13. С. 31.

17. Бондарев О. Хто в домі господар. Розумні будинки через кілька років набудуть широкої популярності. Кореспондент. 2012. №30. С. 42–46.

18. Saha, S., Ishraque, H., Islam, M.T., & Rahman, M.A. IoT based smart home automation and energy management. In 2019 Thesis & Report, BSc (Electrical and Electronic Engineering) (Department of Electrical and Electronic Engineering, Brac University). (2019). P. 85.

19. Jiang L., Liu D.Y., Yang B. Smart home research. Proceedings of the 2004 International Conference on Machine Learning and Cybernetics, Shanghai, China, August. 2004. Vol. 2. P.165-169.

20. Паньків В. Г. Український ринок систем автоматизації та диспетчеризації. Мережі та бізнес системи, 2011. №3. С. 58–62.

21. Chan M., Esteve D., Escriba C., Campo E. A review of smart homes – Present state and future challenges. Computer Methods and Programs in Biomediscine. 2008. Vol. 91. P. 55-81.

22. Теслюк В.М., Борейко О.Ю., Сидор А.Р., Береговська Х.В. Модель телекомунікаційної мережі інтелектуального будинку. Науковий вісник НЛТУ України. 2016. №26.1. С.351-357.

23. Teslyuk V., Beregovskyi V., Pukach A. Automation of the smart house system-level design. Informatyka Automatyka Pomiary w Gospodarce i Ochronie Środowiska. Polish magazin. 2013. Zeszyt 4. P.81 – 84.

24. Granzer W.P. Security in Building Automation Systems. Munich: Appress, 2018. 578 c.

25. Teslyuk V.M., Beregovskyi V.V., Pukach A.I. Development of smart house system model based on colored Petri nets, Proc. of the XVIII-th International Seminar.

Workshop On Direct And Inverse Problems Of Electromagnetic And Acoustic Wave Theory (DIPED – 2013), Lviv, Ukraine, 2013. P. 205-208.

26. Котунова, Д. Г. Огляд DIY елементів для систем «Smart Home» / Д. Г. Котунова, О. М. Павловський // XIII Науково-практична конференція студентів, аспірантів та молодих вчених «Погляд у майбутнє приладобудування», 13-14 травня 2020 р., м. Київ, Україна : збірник праць конференції. – Київ : КПІ ім. Ігоря Сікорського, 2020. – С. 35–38.

27. Collotta M., Pau G. A Solution Based on Bluetooth Low Energy for Smart Home Energy Management. Energies. 2015. Т. 8. №. 10. С. 11916-11938.

28. Cheng J., Kunz T. A survey on smart home networking. Carleton University, Systems and Computer Engineering, Technical Report, SCE-09-10. 2009

29. Fouladi B., Ghanoun S. Security Evaluation of the Z-Wave Wireless Protocol. Black hat USA. 2013. Т. 24.

30. Liang L., Huang L., Jiang X., Yao Y. Design and implementation of wireless Smart-home sensor network based on ZigBee protocol. International Conference “Communications, Circuits and Systems” (ICCCAS’2008), October 14-17, 2008. P. 434-438

ДОДАТКИ

Додаток А

Фрагмент коду модулю кіберзахисту розумного будинку

робота з файлами формату json


```
{

 "dockerFile": "../Dockerfile.dev",
 "postCreateCommand": "git config --global --add safe.directory ${containerWorkspaceFolder} && script/setup",
 "postStartCommand": "script/bootstrap",
 "containerEnv": {
  "PYTHONASYNCIODEBUG": "1"
 },
 "features": {
  "ghcr.io/devcontainers/features/github-cli:1": {}
 },
 // Port 5683 udp is used by Shelly integration
 "appPort": ["8123:8123", "5683:5683/udp"],
 "runArgs": [
  "-e",
  "GIT_EDITOR=code --wait",
  "--security-opt",
  "label=disable"
 ],
 "customizations": {
  "vscode": {
   "extensions": [
    "charliermarsh.ruff",
    "ms-python.pylint",
```

```
        "ms-python.vscode-pylance",
        "visualstudioexptteam.vscodeintellicode",
        "redhat.vscode-yaml",
        "esbenp.prettier-vscode",
        "GitHub.vscode-pull-request-github",
        "GitHub.copilot"
      ],

      "settings": {
        "python.experiments.optOutFrom": ["pythonTestAdapter"],
        "python.defaultInterpreterPath": "/home/vscode/.local/ha-venv/bin/python",
        "python.pythonPath": "/home/vscode/.local/ha-venv/bin/python",
        "python.terminal.activateEnvInCurrentTerminal": true,
        "python.testing.pytestArgs": ["--no-cov"],
        "pylint.importStrategy": "fromEnvironment",
        "editor.formatOnPaste": false,
        "editor.formatOnSave": true,
        "editor.formatOnType": true,
        "files.trimTrailingWhitespace": true,
        "terminal.integrated.profiles.linux": {
          "zsh": {
            "path": "/usr/bin/zsh"
          }
        },
        "terminal.integrated.defaultProfile.linux": "zsh",
        "yaml.customTags": [
          "!input scalar",
          "!secret scalar",
          "!include_dir_named scalar",
```

```
        "!include_dir_list scalar",
        "!include_dir_merge_list scalar",
        "!include_dir_merge_named scalar"
      ],
      "[python]": {
        "editor.defaultFormatter": "charliermarsh.ruff"
      },
      "json.schemas": [
        {
          "fileMatch": ["homeassistant/components/*/manifest.json"],
          "url": "./script/json_schemas/manifest_schema.json"
        }
      ]
    }
  }
}
```

Фрагмент коду модуля кіберзахисту аудит паролів

```
import unittest
import os
import testlib

class SecurityTest(testlib.TestlibCase):
   '''Test  security features'''

   def setUp(self):
      '''Set up prior to each test_* function'''
```

```
    self.fs_dir = os.path.abspath('.')
    os.chdir('glibc-security')

def tearDown(self):
    '''Clean up after each test_* function'''
    os.chdir(self.fs_dir)

# ----h
def test_00_make(self):
    '''Build helper tools'''

    self.announce("gcc %s" % (self.gcc_version))
    self.assertShellExitEquals(0, ["make", "clean"])
    self.assertShellExitEquals(0, ["make"])

# Всі релізи releases
def test_11_heap_protector(self):
    '''glibc heap protection'''

    self.assertShellExitEquals(0, ["./heap", 'safe'])
    self.assertShellExitEquals(-6, ["./heap", 'unsafe'])

# All releases (got fixed in Intrepid)
def test_11_sprintf_unmangled(self):
    '''sprintf not pre-truncated with -D_FORTIFY_SOURCE=2'''
    expected = 0
    if self.lsb_release['Release'] == 8.04:
        self._skipped("Hardy known broken")
        expected = 1
    self.assertShellExitEquals(expected, ["./sprintf"])
```

```
# All releases
def test_12_glibc_pointer_obfuscation(self):
    '''glibc pointer obfuscation'''

    # glibc implementation of PTR_MANGLE/PTR_DEMANGLE
    #
    # locate values via:
    #  cd glibc-*
    #  fakeroot debian/rules patch
    #  cd build-tree/glibc-*
    #  grep -R JB_PC sysdeps/ | grep define

    expected = 0
    jb_pc = '-1'
    jb_unenc = '-1' # unencrypted isn't stable
    if self.dpkg_arch == 'i386' or self.dpkg_arch == 'lpia':
        jb_pc = '5'
        #jb_unenc = '1'
    elif self.dpkg_arch == 'amd64':
        jb_pc = '7'
        #jb_unenc = '0'
    elif self.dpkg_arch == 'armel':
        self._skipped("not on ARM")
        expected = 2

    if self.lsb_release['Release'] < 7.04:
        self._skipped("only Edgy and later")
        expected = 100
```

```
    self.assertShellExitEquals(2, ["./ptr-enc", '-1', '-1'])
    self.assertShellExitEquals(200, ["./ptr-enc", '-2', '-2'])
    self.assertShellExitEquals(expected, ["./ptr-enc", jb_pc, jb_unenc])

# Precise and later (glibc 2.15+)
def test_13_select_overflow(self):
    '''select macros detect overflow with -D_FORTIFY_SOURCE=2'''
    expected = -6
    if self.lsb_release['Release'] < 12.04:
        self._skipped("only Precise and later")
        expected = 4
    self.assertShellExitEquals(0, ["./select", "200"])
    self.assertShellExitEquals(expected, ["./select", "1500"])
    self.assertShellExitEquals(expected, ["./select", "-100"])

def _get_expected_hashes(self):
    '''Returns a tuple of (algortihm, expected pattern) for the hash
       algorithm in the current ubuntu release'''
    algorithm = 'yescrypt'
    expected_pattern = '$y$'
    if self.lsb_release['Release'] < 8.10:
        algorithm = 'md5'
        expected_pattern = '$1$'
    elif self.lsb_release['Release'] < 22.04:
        algorithm = 'sha512'
        expected_pattern = '$6$'
    self.announce(algorithm)

    return (algorithm, expected_pattern)
```

```
        # Is this really glibc?  I guess it is since it's crypt() kinda...
        def test_41_passwd_hashes(self):
            '''Password hashes'''

            (expected, pattern) = self._get_expected_hashes()

            # in some CI environments, there may not be a user with a
            # password set; create one temporarily
            test_user = testlib.AddUser()

            rc, output = self.shell_cmd(['cut', '-d:', '-f2', '/etc/shadow'])
            self.assertEqual(rc, 0, "Got %d (expected %d):\n%s" % (rc, 0, output))
            seen = False
            star = False
            for hash in output.splitlines():
                if hash.startswith(pattern):
                    seen = True
                if hash == '*':
                    star = True
            self.assertTrue(star, "'*' locked password not found in /etc/shadow:\n%s" %
(output))
            self.assertTrue(seen, "%s hash not found in /etc/shadow:\n%s" %
(expected, output))

        def test_42_passwd_hash_alg_pam(self):
            '''test the default hash algorithm configured in pam'''

            (expected, _) = self._get_expected_hashes()
            seen = None
            pam_file = None
```

```
            for pam_path in ['/etc/pam.d/common-password', '/etc/pam.d/system-auth']:
                if os.path.exists(pam_path):
                    pam_file = pam_path
                    break
            self.assertTrue(pam_file != None, "Cannot find pam password config file")
            with open(pam_file) as f:
                for line in f:
                    if line.startswith('password') and 'pam_unix.so' in line:
                        seen = expected in line
                        break
            self.assertTrue(seen!=None, "pam_unix.so line not found in %s" % (pam_file))
            self.assertTrue(seen, "%s argument not found in %s" % (expected, pam_file))

        def _run_env_cmd(self, cmd, env_var, value, expected=0):
            new_env = os.environ.copy()
            new_env[env_var] = value
            self.assertShellExitEquals(expected, ['su', '-c', '%s %s' % (cmd, env_var), os.environ['SUDO_USER']], env=new_env)

        env_vars = [
            ('LD_LIBRARY_PATH', os.path.join(os.environ['HOME'], 'lib')),
            ('LD_PRELOAD', os.path.join('./libtest-library.so')),
            ('LD_AUDIT', os.path.join('./libtest-library.so')),  # XXX find a real audit library
            ('LD_DEBUG', 'statistics'),
        ]
```

```
def test_70_env_okay_non_setuid(self):
    '''Ensure environment vars passed through to non-setuid progs'''
    for (env, value) in self.env_vars:
        with self.subTest(testcase=env):
            self._run_env_cmd("./env-is-defined", env, value)

def test_71_env_filtered_setuid(self):
    '''Ensure environment vars filtered for setuid progs'''
    for (env, value) in self.env_vars:
        with self.subTest(testcase=env):
            self._run_env_cmd("./env-is-defined-setuid", env, value, expected=1)

def test_72_envs_okay_for_setuis(self):
    '''Ensure we didn't break misc env vars for setuid progs'''

    self._run_env_cmd("./env-is-defined", "FOO", "BAR")
    self._run_env_cmd("./env-is-defined-setuid", "FOO", "BAR")



# Edgy and newer
def test_80_stack_guard_exists(self):
    '''Stack guard exists'''

    rc_expected = 0
    if self.lsb_release['Release'] < 6.10:
        self._skipped("only Edgy and later")
        rc_expected = 1

    rc, one  = testlib.cmd(["./guard"])
    self.assertEqual(rc, rc_expected, one)
```

```
# Edgy and newer
def test_81_stack_guard_leads_zero(self):
    '''Stack guard leads with zero byte'''

    rc_expected = 0
    expected = True
    if self.lsb_release['Release'] < 6.10:
        self._skipped("only Edgy and later")
        rc_expected = 1
    # This should not be: https://bugs.launchpad.net/bugs/413278
    #if self.lsb_release['Release'] > 9.04:
    #    expected = False
    #    self._skipped("stopped in Jaunty+")

    rc, one  = testlib.cmd(["./guard"])
    self.assertEqual(rc, rc_expected, one)
    if rc_expected == 0:
        # Try three times just to avoid randomized luck
        self.assertEqual(one.startswith('00 '), expected, one)
        rc, two  = testlib.cmd(["./guard"])
        self.assertEqual(rc, rc_expected, two)
        self.assertEqual(two.startswith('00 '), expected, two)
        rc, three  = testlib.cmd(["./guard"])
        self.assertEqual(rc, rc_expected, three)
        self.assertEqual(three.startswith('00 '), expected, three)

# Intrepid and newer
def test_82_stack_guard_randomized(self):
    '''Stack guard is randomized'''
```

```
            rc_expected = 0
            if self.lsb_release['Release'] < 6.10:
                # Only Edgy and later can run this test
                rc_expected = 1

            expected = True
            # Fixed for Hardy in 2.7-10ubuntu5 from -proposed
            if self.lsb_release['Release'] < 8.04:
                self._skipped("only Hardy and later")
                expected = False

            rc, one  = testlib.cmd(["./guard"])
            self.assertEqual(rc, rc_expected, one)

            if rc_expected == 0:
                rc, two  = testlib.cmd(["./guard"])
                self.assertEqual(rc, rc_expected, two)
                rc, three = testlib.cmd(["./guard"])
                self.assertEqual(one != two and one != three and two != three, expected,
one + two + three)

        # Karmic and newer
        def test_90_abort_msg(self):
            '''Retains assert()/*_chk() message'''

            self.announce(self.path_libc)

            # Not strictly security, but good to check for anyway.
            expected = True
```

```
        if self.lsb_release['Release'] < 9.10:
            # Only Karmic and later are expected to have this
            self._skipped("only Karmic and later")
            expected = False

        rc, out = testlib.cmd(["readelf", "--wide", "-s", self.path_libc])
        self.assertEqual(rc, 0, out)
        self.assertEqual(expected, ' __abort_msg@@' in out, out)

# other things to test...
#~~~~~~~~~~~~~~~~~~~~~~~~~~
```