

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО»
НАВЧАЛЬНО-НАУКОВИЙ ФІЗИКО-ТЕХНІЧНИЙ ІНСТИТУТ
Кафедра інформаційної безпеки

До захисту допущено
Завідувач кафедри

_____ **Дмитро ЛАНДЕ**
(підпис)

« _____ » _____ 2025 р.

Дипломна робота
на здобуття ступеня бакалавра
за освітньо-професійною програмою «Системи, технології та математичні
методи кібербезпеки»
спеціальності 125 «Кібербезпека»

на тему: Реалізація концепції Zero Trust в архітектурі VoIP-телекомунікацій

Виконав (-ла): здобувач вищої освіти **IV** курсу, групи **ФБ-14**
(шифр групи)

Болгов Микола Валерійович
(прізвище, ім'я, по батькові)


(підпис)

Керівник **старший викладач каф. ІБ Козленко О. В.**
(посада, науковий ступінь, вчене звання, прізвище, ім'я, по батькові)

(підпис)

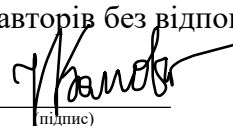
Рецензент **К.ф.-м.н., доцент кафедри ММАД Терещенко І. М.**

(посада, науковий ступінь, вчене звання, науковий ступінь, прізвище, ім'я, по батькові)

(підпис)

Засвідчую, що у цій дипломній роботі немає
запозичень з праць інших авторів без відповідних
посилань.

Здобувач вищої освіти


(підпис)

Київ – 2025 року

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО»
НАВЧАЛЬНО-НАУКОВИЙ ФІЗИКО-ТЕХНІЧНИЙ ІНСТИТУТ
Кафедра інформаційної безпеки

Рівень вищої освіти – перший (бакалаврський)

Спеціальність – 125 «Кібербезпека»

Освітньо-професійна програма «Системи, технології та математичні методи кібербезпеки»

ЗАТВЕРДЖУЮ

Завідувач кафедри

_____ Дмитро ЛАНДЕ
(підпис)

« ____ » _____ 2025 р.

ЗАВДАННЯ
на дипломну роботу здобувачу вищої освіти

Болгов Микола Валерійович

(прізвище, ім'я, по батькові)

1. Тема роботи «Реалізація концепції Zero Trust в архітектурі VoIP-телекомунікацій»,

керівник роботи Козленко Олег Віталійович, старший викладач каф. ІБ,
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

затверджені наказом по університету від « 26 » Травня 2025 р. № 1761-с

2. Термін подання здобувачем вищої освіти роботи « 10 » Червень 2025 р.

3. Вихідні дані до роботи: базова література про протоколи SIP, RTP, стандарти та принципи моделі Zero Trust, типові сценарії атак на VoIP-системи, особливості архітектури VoIP-систем на базі Asterisk та Kamailio

4. Зміст роботи: огляд основних принципів, на яких базується робота IP-телефонії, огляд принципів та переваг моделі Zero Trust, опис архітектурно-програмного рішення та порівняльний аналіз переваг VoIP-системи із імплементованою моделлю нульової довіри перед незахищеною системою

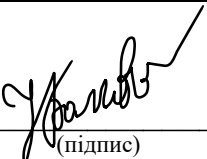
5. Перелік ілюстративного матеріалу: Презентація до захисту дипломної роботи.

6. Дата видачі завдання: 12.09.2024

Календарний план

№ з/п	Назва етапів виконання дипломної роботи	Термін виконання етапів дипломної роботи	Примітка
1	Ознайомлення із базовими принципами роботи VoIP технологій; аналіз основних вразливостей	12.09.2024 – 01.02.2025	Виконано
2	Теоретичний аналіз принципів моделі нульової довіри та її доцільності при впровадженні корпоративної IP-телефонії	01.02.2025 – 28.02.2025	Виконано
3	Практичне розгортання тестового VoIP середовища	01.03.2025 – 31.03.2025	Виконано
4	Впровадження принципів Zero Trust у конфігурацію SIP-проксі сервера	01.04.2025 – 14.04.2025	Виконано
5	Розробка веб-застосунку та системи моніторингу	14.04.2025 – 30.04.2025	Виконано
6	Формалізація рішення	01.05.2025 – 16.05.2025	Виконано
7	Порівняльний аналіз традиційної VoIP систем із впровадженою моделлю нульової довіри	17.05.2025 – 31.05.2025	Виконано
8	Аналіз отриманих результатів	01.06.2025 – 10.06.2025	Виконано

Здобувач вищої освіти


(підпис)

Микола БОЛГОВ
(Власне ім'я, ПРІЗВИЩЕ)

Керівник роботи

(підпис)

Олег КОЗЛЕНКО
(Власне ім'я, ПРІЗВИЩЕ)

РЕФЕРАТ

Обсяг дипломної роботи 80 сторінок, 23 ілюстрацій, 4 таблиці, 2 додатки і джерела літератури.

Об'єкт дослідження: рішення корпоративної IP-телефонії організації

Предмет дослідження: методи та засоби забезпечення кібербезпеки VoIP-системи з використанням концепції Zero Trust.

Мета дослідження: Дослідити, розробити та обґрунтувати практичний підхід до впровадження моделі Zero Trust у VoIP-системах з метою підвищення рівня захищеності телекомунікаційної системи.

Методи дослідження: Ознайомлення із базовою архітектурою VoIP систем (Asterisk, Kamailio) та основними протоколами взаємодії (SIP, RTP, SDP); аналіз типових загроз та вразливостей IP-телефонії; розгортання та практичне налаштування середовища Asterisk + Kamailio із впровадженням основних принципів Zero Trust у розгорнуту архітектуру, розробка та реалізація програмного забезпечення для моніторингу та логування роботи системи.

Отримані результати: Проаналізовано вразливі компоненти типової VoIP системи, запропоноване архітектурне рішення з реалізацією принципів Zero Trust, реалізовано систему логування та моніторингу системи, впроваджено захист медіа-сервера шляхом розгортання SIP-проксі сервера, розроблено веб-застосунок для забезпечення мінімального доступу користувачів системи до критичної інформації.

Ключові слова: VoIP, IP-телефонія, Zero Trust, модель нульової довіри, SIP, кібербезпека, Asterisk, Kamailio.

ABSTRACT

The thesis comprises 80 pages, 23 illustrations, 4 tables, 2 appendices, and a list of references.

Object of research: corporate IP telephony solutions of an organization.

Subject of research: methods and tools for ensuring cybersecurity of VoIP system using the Zero Trust concept.

Purpose of research: to investigate, design, and justify a practical approach to implementing the Zero Trust model in VoIP systems to enhance the security level of a telecommunications infrastructure.

Research methods: familiarization with the basic architecture of VoIP systems (Asterisk, Kamailio) and key communication protocols (SIP, RTP, SDP); analysis of typical threats and vulnerabilities of IP telephony; deployment and configuration of an Asterisk + Kamailio environment with the integration of Zero Trust principles into the architecture; development and implementation of software for system monitoring and logging.

Results obtained: vulnerable components of a typical VoIP system were analyzed; an architectural solution based on Zero Trust principles was proposed; a logging and monitoring system was implemented; the media server was secured by deploying a SIP proxy; a web application was developed to ensure minimum access for users to critical system data.

Keywords: VoIP, IP telephony, Zero Trust, SIP, cybersecurity, Asterisk, Kamailio.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ, СКОРОЧЕНЬ І ТЕРМІНІВ	8
ВСТУП.....	9
1 ПРИНЦИПИ РОБОТИ VOIP ТЕЛЕКОМУНІКАЦІЙ ТА ПРОБЛЕМИ ЇХ ЗАХИСТУ	11
1.1 Основи роботи VoIP-систем	12
1.2 Основні протоколи VoIP-систем.....	14
1.3 Протокол SIP	16
1.4 Роль Asterisk та Kamailio у VoIP-мережах	23
1.5 Основні вразливості у VoIP-телекомунікаціях.....	25
Висновки до розділу 1	30
2 МОДЕЛЬ ZERO TRUST ЯК ОСНОВА БЕЗПЕЧНОЇ VOIP- АРХІТЕКТУРИ. ФОРМАЛЬНЕ ПРЕДСТАВЛЕННЯ РІШЕННЯ	32
2.1 Концепція Zero Trust: принципи, вимоги та релевантність імплементатції моделі для VoIP-телекомунікацій	33
2.2 Проблеми безпеки VoIP та їх подолання за допомогою принципів Zero Trust	34
2.3 Формальна архітектура	39
2.4 Очікувані переваги імплементатції моделі нульової довіри при побудові VoIP-системи.....	41
Висновки до розділу 2	42
3 ПРАКТИЧНА РЕАЛІЗАЦІЯ ТА ПОРІВНЯЛЬНИЙ АНАЛІЗ	44
3.1 Розгортання базової VoIP-архітектури та впровадження у неї моделі Zero Trust.....	44

3.2 Порівняльний аналіз системи із моделлю нульової довіри та незахищеної системи	53
Висновки до розділу 3	58
ВИСНОВКИ	60
ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАНЬ	62
ДОДАТОК А.....	64
ДОДАТОК Б	72

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

VoIP (Voice over Internet Protocol) — технологія передавання голосу та мультимедійного трафіку через IP-мережі.

SIP (Session Initiation Protocol) — протокол сигналізації, що використовується для встановлення, управління та завершення VoIP-сеансів.

SDP (Session Description Protocol) — протокол опису мультимедійних сесій, що працює спільно з SIP.

RTP (Real-Time Transport Protocol) — протокол передавання аудіо- та відеоданих у режимі реального часу.

SRTP (Secure Real-Time Transport Protocol) — розширення RTP для забезпечення шифрування та автентичності медіатрафіку.

ATC (Автоматична телефонна станція) — пристрій або програмний комплекс для управління телефонними викликами в організації.

UDP (User Datagram Protocol) — транспортний протокол без встановлення з'єднання, який часто використовується у VoIP.

TCP (Transmission Control Protocol) — транспортний протокол з підтвердженням доставки та керуванням потоком.

TLS (Transport Layer Security) — криптографічний протокол для захищеної передачі даних.

DoS (Denial of Service) — атака на відмову в обслуговуванні, спрямована на виведення з ладу мережевого сервісу.

CRM (Customer Relationship Management) — система управління взаємовідносинами з клієнтами

ВСТУП

Останнім часом дедалі більше компаній від невеликого бізнесу, до корпоративних гігантів та кол-центрів на тисячі операторів обирають впровадження IP-телефонії як засобу корпоративного зв'язку та інструменту для автоматизації своїх бізнес-процесів. Технологія VoIP (Voice over IP) дозволяє передавати голосовий трафік через IP-мережі, тобто здійснювати дзвінки через Інтернет, що, своєю чергою, відкриває широкий спектр можливостей: зниження витрат компанії на зв'язок, автоматизація та гнучке масштабування бізнес-процесів, інтеграції з CRM системами. Таке рішення дозволяє розширити можливості бізнесу та покращити продажі та якість обслуговування клієнтів, об'єднати та централізовано адмініструвати всю систему зв'язку незалежно від географічного розташування співробітників організації.

Але, своєю чергою, впровадження рішення IP-телефонії викликає питання стосовно порушення основних принципів безпеки компанії, а саме конфіденційності розмов під час онлайн-викликів, цілісності системи від несанкціонованих дій (починаючи від порушення базових привілеїв та доступів до системи, закінчуючи крадіжкою та подальшим несанкціонованим використанням баз номерів клієнтів) та її постійної доступності, що є критичним для бізнесу, оскільки кожна хвилина простою бізнесу – це тисячі втрачених потенційних клієнтів, а отже фінансовий ризик для компанії.

VoIP системи можуть легко стати об'єктом атак злоумисників, зокрема, через brute-force атаки на сервер автентифікації хакери можуть отримати несанкціонований доступ до SIP-акаунтів, а DoS-атаки взагалі можуть вивести систему з ладу. Окрім того не варто забувати про можливість прослуховування голосового трафіку несанкціонованими сторонами та загрозами інформаційної безпеки зсередини системи.

Через це постає питання у побудові захищеної системи, що можна легко реалізувати шляхом впровадження моделі «нульової довіри» Zero Trust. Ця

модель передбачає постійне підтвердження автентичності користувачів, пристроїв та запитів до системи, незалежно від того, звідки вони походять. Ретельний контроль, постійний моніторинг активності та дотримання інших принципів моделі Zero Trust є ключем до побудови захищеної системи.

Метою роботи є визначення та практичної реалізації впровадження моделі Zero Trust у VoIP-системах з метою підвищення рівня захищеності телекомунікаційної інфраструктури організації.

Об'єктом дослідження є інформаційно-телекомунікаційна інфраструктура організації, яка включає VoIP-сервер, елементи маршрутизації трафіку, сервіси автентифікації та адміністрування.

Предметом дослідження є методи та засоби реалізації забезпечення безпеки VoIP-системи з використанням концепції Zero Trust, зокрема контроль доступу, перевірка ідентичності, моніторинг SIP-трафіку та обробка інцидентів.

У процесі роботи використано такі **методи дослідження**: аналіз технічних джерел з тематики VoIP та Zero Trust; вивчення основ роботи SIP-протоколу та механізмів автентифікації в Asterisk і Kamailio; практична реалізація захищеної VoIP-архітектури з використанням принципів Zero Trust;

Практичне застосування: запропоноване рішення можна легко запровадити у роботу організації, яка використовує рішення IP-телефонії для забезпечення комплексного захисту VoIP-системи.

1 ПРИНЦИПИ РОБОТИ VOIP ТЕЛЕКОМУНІКАЦІЙ ТА ПРОБЛЕМИ ЇХ ЗАХИСТУ

VoIP (Voice over Internet Protocol) – технологія передачі медіа-даних (наприклад, аудіо та відео) у режимі реального часу через мережу Інтернет, за допомогою стеку протоколів TCP/IP. Завдяки принципам роботи VoIP технологій можлива побудова системи зв'язку, яка зветься IP-телефонією. В такій системі голос абонента кодується у цифрову форму та в режимі реального часу передається через IP-мережу до кінцевого абонента.

Така система має низку численних переваг, які зробили спілкування через IP мережі потужним інструментом корпоративного зв'язку та популярною альтернативою традиційним телекомунікаційним системам та звичайним дзвінкам через месенджери. Однією із головних переваг IP телефонії є суттєве зниження витрат на зв'язок, яке дозволяє бізнесу вести ефективну комунікацію із клієнтами по всьому світу за оптимальну вартість дзвінків. Гнучкість та масштабованість VoIP-рішень є не менш важливою перевагою. Впровадження корпоративної телефонії забезпечує високу мобільність персоналу та дозволяє організувати ефективну дистанційну роботу (навіть, якщо працівник компанії працює у іншій країні – він матиме можливість здійснювати дзвінки, головне мати доступ до мережі Інтернет).

Окрім цього, VoIP-системи забезпечують широкий набір можливостей з маршрутизації та обробки телефонного виклику, які відсутні в традиційній аналоговій телефонії. До таких можливостей можна віднести запис розмов, автоматичні інформуючі дзвінки, можливість маршрутизувати дзвінок за різними правилами, починаючи від простого з'єднання із оператором, закінчуючи многорівневими голосовими меню (IVR), у яких кінцева точка, із якою буде з'єднано абонента залежатиме від його вибору. Також, розглядаючи ключові можливості VoIP систем не варто забувати про можливість інтеграцій із CRM системами взаємовідносин із клієнтами (наприклад, ZOHO, KeyCRM,

RemOnline, тощо), що дозволяє автоматизувати бізнес-процеси та підвищити ефективність роботи компанії та якості обслуговування клієнтів.

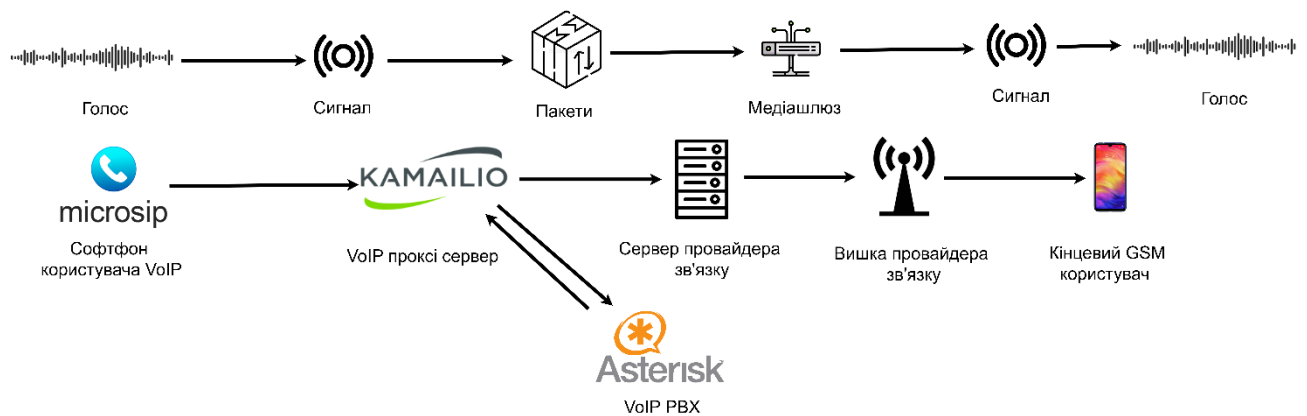
1.1 Основи роботи VoIP-систем

Технології VoIP забезпечують передачу голосових даних через IP-мережі, а отже для того, аби обидві сторони у діалозі чули одна одну необхідне перетворення аналогового голосового сигналу у цифровий, його передача пакетами IP, а потім зворотне перетворення на стороні отримувача.

Приблизна схема дзвінка у такій системі відбувається із взаємодією із мережею PSTN (телефонна мережа загального користування) або GSM (мережа мобільного зв'язку).

Все розпочинається на етапі ініціалізації виклику користувачем А, який користується софтбоном (програмне забезпечення для VoIP, наприклад MicroSIP, Zoiper), встановленим на ПК або мобільний телефон, або хардфоном (стаціонарний IP-телефон, наприклад Grandstream, Yealink). Користувач А ініціює виклик, вводячи SIP-адресу або номер, надсилаючи SIP-повідомлення **INVITE** на сервер (наприклад, Kamailio або Asterisk), який виконує маршрутизацію виклику, застосовуючи при цьому політики безпеки, перевіривши облікові дані користувача А.

Якщо отримувач також використовує софтфон або IP-телефон, SIP-повідомлення INVITE перенаправляється до нього. Якщо ж отримувач - користувач мобільної або фіксованої мережі (PSTN/GSM), SIP-сервер передає виклик серверу провайдера зв'язку. Далі VoIP-виклик передається через медіашлюз (Media Gateway) та маршрутизується оператором мобільного зв'язку на вишку базової станції, яка забезпечується зв'язок із мобільним пристроєм користувача В. У висновку, дзвінок потрапляє до кінцевого GSM абонента (на звичайну SIM-картку).



Рисунунок 1.1 - Приблизна схема дзвінка із IP мережі у GSM мережу

Сесія встановлюється та сигналізується за допомогою протоколу **SIP (Session Initiation Protocol)**, який є основою кожного дзвінка через IP-телефонію. Саме SIP забезпечує ініціацію, підтримку та завершення кожного сеансу.

На етапі встановлення сесії між учасниками виклику узгоджуються параметри медіа-трафіку, такі як кодеки (забезпечують перетворення аналогового сигналу у цифровий та навпаки декодування його назад у звук, наприклад, G.711, G.729, Opus), IP-адреси, порти, тощо за допомогою протоколу **SDP (Session Description Protocol)**, який передається у тілі SIP-повідомлень.

Медіа передається у вигляді RTP-потoku (**Real-Time Transport Protocol**), який передається окремо від сигнального трафіку. RTP-пакети, які містять закодований голос, передаються від софтфону користувача А безпосередньо до пристрою користувача В або через медіасервер. У випадку, якщо кінцевий користувач В знаходиться у GSM-мережі, медіашлюз виконує перетворення RTP-потoku у формат, сумісний із TDM (Time Division Multiplexing).

На стороні користувача В голосові пакети приймаються, декодуються у зворотному порядку (використовуючи той самий кодек), і перетворюються у **аналоговий голосовий сигнал**, який відтворюється через динамік телефону.

1.2 Основні протоколи VoIP-систем

Описаний вище процес передачі голосу в мережах VoIP ґрунтується на взаємодії кількох мережевих протоколів, кожен з яких виконує специфічну функцію: від ініціалізації сеансу до безпосередньої передачі та захисту медіаданих.

VoIP-протоколи визначають послідовність, правила та формат обміну даними для встановлення та підтримки сеансу VoIP-зв'язку [1]. Кожен виклик через IP-телефонію одночасно використовує кілька таких протоколів для забезпечення транспорту, встановлення сеансу та сигналізації передачі голосових даних у режимі реального часу та забезпечення безпеки з'єднання (автентифікації користувачів, контролю доступу та шифрування трафіку).

У VoIP-системах, у якості протоколів транспортного рівня зазвичай використовуються UDP (User Datagram Protocol), TCP (Transmission Control Protocol) та TLS (Transport Layer Security), саме вони забезпечують доставку даних між двома кінцевими пристроями.

Найчастіше за все VoIP-трафік передається саме із використанням **UDP**, оскільки саме цей протокол забезпечує мінімальну затримку передачі пакетів, що є доволі критичним для комунікації в режимі реального часу, особливо якщо інтернет-підключення операторів є нестабільним. Проте передача даних через UDP має і свої недоліки: протокол не гарантує доставки пакетів та під час його використання при поганій якості мережі можливі втрати голосових фрагментів, які можуть призводити до спотворення при передачі голосу або взагалі переривань під час розмови. **TCP**, зі свого боку, хоч і не є оптимальним для передачі мультимедійних потоків, часто застосовується саме для сигнального обміну, зокрема у протоколах SIP або HTTP-запитах до сервісів VoIP.

Однак, обидва протоколи у своїй базовій реалізації не забезпечують жодного шифрування, тобто вся передана інформація - включаючи SIP-повідомлення, IP-адреси, номери телефонів та навіть вміст голосових пакетів (якщо не використовується SRTP) – передається у відкритому, незашифрованому вигляді, що відкриває уразливість такого трафіку для перехоплення, спуфінгу

або прослуховування та створює значні ризики конфіденційності переданих даних. Саме для таких випадків, для усунення цих загроз використовується **TLS (Transport Layer Security)**, який створює захищене з'єднання між двома точками, шифруючи сигнальний трафік. TLS дозволяє захистити обмін SIP-повідомленнями та унеможливити перехоплення інформації про учасників дзвінка, маршрутизацію або зміст повідомлень. У поєднанні з SRTP для медіа TLS утворює основу повноцінної VoIP-безпеки для безпечної та конфіденційної комунікації [2].

Ключовим протоколом у VoIP-телекомунікаціях є **SIP (Session Initiation Protocol)**, саме він відповідає за створення, модифікацію та завершення сеансів зв'язку. За концепцією, він дуже схожий на HTTP (Hypertext Transfer Protocol), проте має свої особливості, які розглянемо у наступному розділі. Саме завдяки SIP відбувається реєстрація користувача на VoIP-сервері, надсилання викликів, встановлення з'єднання та завершення сеансів. Тобто, саме завдяки цьому протоколу кінцеві пристрої взаємодіють один між одним, ініціалізують початок та кінець дзвінка.

Однією з необхідних умов встановлення сеансу за допомогою SIP є обмін між сторонами інформацією про їх функціональні можливості (параметри та технології прийому даних). Зазвичай для цього використовується окремий протокол - **SDP (Session Description Protocol)**. SDP-повідомлення вбудовується у тіло SIP-повідомлення INVITE та містить дані про кодек, типи медіа, IP-адресу, порти та параметри сесії. Оскільки параметри сесії можуть змінюватися впродовж зв'язку (наприклад, при додаванні нових учасників у конференцію), SIP дозволяє надсилати оновлені SDP-повідомлення для внесення змін до активного сеансу.

Якщо протокол SIP керує сесіями дзвінків, а SDP допомагає кінцевим пристроям «домовитися» про те, як вони будуть проводити сеанс, то протокол **RTP (Real-time Transport Protocol)** є протоколом, який дозволяє в режимі реального часу передавати голосові дані. Він забезпечує механізми відстеження порядку пакетів, синхронізації та часової мітки, що дозволяє отримувачу

правильно відтворити аудіопотік під час дзвінка, проте навіть ці механізми не забезпечують надійність доставки даних (особливо якщо дзвінок відбувається через UDP). Окрім того, RTP не забезпечує шифрування, тому при передачі голосу через незахищенні канали можна легко перехопити вміст розмови або підмінити пакети. Саме тому для захисту VoIP-трафіку був створений **SRTP (Secure Real-time Transport Protocol)** - розширення до RTP, що забезпечує шифрування та аутентифікацію пакетів.

Супутнім до роботи RTP/SRTP є протокол **RTCP (RTP Control Protocol)**, який забезпечує зворотний зв'язок про якість обслуговування (**QoS**) під час мультимедійної сесії. Він працює паралельно з RTP, періодично надсилаючи службові повідомлення, які дають змогу відстежити кількість переданих та втрачених RTP-пакетів, середню затримку та наявність джитеру.

Для контролю за медіашлюзами між IP-мережею (Інтернетом) та телефонною мережею використовується **MGCP (Media Gateway Control Protocol)**.

1.3 Протокол SIP

SIP (Session Initiation Protocol) – це ключовий протокол у IP-телефонії, який відіграє керуючу роль у сеансах дзвінків та є механізмом встановлення, маршрутизації, модифікації та завершення сеансів зв'язку. Саме завдяки SIP встановлюється з'єднання між софтбоном абонента А та звичайним мобільним телефоном абонента В із SIM картою. Робота протоколу описана у стандарті RFC 3261 [3].

Знаходиться SIP на прикладному рівні моделі OSI та працює поверх протоколу UDP (рідше, TCP або TLS). Проте, у базовій реалізації та при використанні перших двох типів транспорту (які використовуються найчастіше) SIP не передбачає шифрування, що означає, що SIP-повідомлення можуть передаватись у відкритому вигляді, що створює серйозні ризики безпеки, які будуть розглянуті більш детально далі.

Для забезпечення сумісності з існуючими веб-технологіями та досягнення мобільності користувачів, SIP використовує **адресацію у вигляді SIP URI**

(Universal Resource Identifier), яка схожа на адресу електронної пошти. Формат SIP-адреси виглядає як: sip:користувач@домен. Для визначення IP-адреси за доменом клієнт звертається до **DNS-серверів** (служба доменних імен). Якщо ж вказано безпосередньо IP, з'єднання встановлюється напряду без резолюції.

SIP – це текстовий протокол, структура якого базується на запитах та відповідях, аналогічно до HTTP. Він використовує набір стандартних методів (команд), які реалізують різні функції управління сеансами зв'язку.

Основними запитамі під час SIP-сесії є:

- **INVITE** - ініціює новий виклик та містить SDP-опис медіапараметрів (кодеки, тощо). Саме з INVITE починається створення SIP-діалогу.
- **ACK** - підтверджує прийняття відповіді 200 OK на INVITE. Завершує процес встановлення виклику.
- **BYE** - завершує активний сеанс між двома сторонами.
- **CANCEL** - скасовує INVITE, якщо з'єднання ще не встановлено.
- **REGISTER** - реєструє SIP-ідентифікатор користувача на SIP-сервері, використовується для автентифікації користувачів.
- **OPTIONS** - запитує інформацію про можливості іншої сторони без встановлення виклику (наприклад, підтримувані методи, кодеки тощо).

Кожен запит супроводжується відповіддю, яка містить трицифровий код стану та короткий текстовий опис. Відповіді у SIP класифікуються за першою цифрою коду, яка вказує на категорію:

Таблиця 1.1 - Класифікація кодів SIP-відповідей

Клас	Тип	Опис
1xx	Інформаційні	Проміжні відповіді -запит обробляється, але ще не завершено
2xx	Успішні	Запит успішно виконано
3xx	Перенаправлення	Необхідна додаткова дія для завершення запиту

Кінець таблиці 1.1

4xx	Клієнтські помилки	Помилка на стороні відправника
5xx	Серверні помилки	Помилка на стороні SIP-сервера
6xx	Глобальні відмови	Відмова, що не залежить від маршрутизації

Основними відповідями у SIP є:

1xx - Інформаційні відповіді

- **100 Trying** - запит прийнято для обробки, але ще не передано адресатові. Відповідь генерує найближчий сервер.
- **180 Ringing** - термінал викликається, сигнал “дзвінок” вже прямує до абонента.
- **183 Session Progress** - повідомлення про те, що виклик обробляється, і можна розпочати передачу медіа (наприклад, програвання мелодії очікування).

2xx - Успішні відповіді

- **200 OK** - успішне завершення запиту. Найважливіша відповідь у SIP, яка підтверджує:
- встановлення виклику (INVITE → 200 OK), у ній зазвичай міститься SDP-відповідь з прийнятими параметрами медіасесії. Після неї клієнт надсилає ACK.;
- завершення сесії (BYE → 200 OK);
- реєстрацію (REGISTER → 200 OK);
- або відповідь на OPTIONS, CANCEL тощо.

3xx - Перенаправлення

- **302 Moved Temporarily** - SIP URI тимчасово змінився, але клієнт має слідувати за новим маршрутом

- **305 Use Proxy** - запит має бути перенаправлений через зазначений SIP-проксі.

4xx - Помилки на стороні клієнта.

Саме ці відповіді використовуються для діагностики складнощів із викликами (чому не змогли додзвонитись до абоненту В).

- **400 Bad Request** - запит має синтаксичну помилку або є некоректним.
- **401 Unauthorized** - потрібна автентифікація (наприклад, при реєстрації через REGISTER).
- **403 Forbidden** - запит відхилено, автентифікація не допоможе.
- **404 Not Found** - користувача або ресурсу не знайдено на сервері (наприклад, набрано неіснуючий номер).
- **407 Proxy Authentication Required** - автентифікація потрібна, але для проксі.
- **480 Temporarily Unavailable** - користувач тимчасово недоступний (наприклад, телефон абонента В вимкнено або знаходиться поза зоною досяжності).
- **484 Address Incomplete** - неповна адреса (наприклад, введено неповний номер абонента В).
- **486 Busy Here** - користувач зайнятий, не може або не хоче (відхиляє) виклик, користується білим списком або програмою для
- **487 Request Terminated** – виклик завершено до встановлення з'єднання (Наприклад, абонент А завершив виклик вручну)
- **488 Not Acceptable Here** - невідповідність медіапараметрів (непідтримуваний кодек, порт тощо).

5xx - Помилки сервера

- **500 Server Internal Error** - внутрішня помилка SIP-сервера, можливо, через перевантаження.
- **503 Service Unavailable** - сервіс недоступний

6xx - Глобальні відмови

- **600 Busy Everywhere** - усі можливі пристрої користувача зайняті.
Наприклад, якщо користувач зареєстрований одночасно на телефоні та комп'ютері.
- **603 Decline** - користувач відмовився приймати виклик.
- **606 Not Acceptable** - неможливо погодити параметри з'єднання (медіа/SDP/кодеки).

У кожного дзвінка є Call-ID, унікальний ідентифікатор SIP-сеансу, який дозволяє серверам і пристроям розпізнати, що всі SIP-повідомлення належать до одного логічного виклику (дзвінка чи сесії). Виглядає він як випадковий ідентифікатор + домен пристрою ініціатора, наприклад:

73hf92hdsf7234h8f7sd89f7sdf@sip.deniro.ua

Для того, аби софтбокс абонент А міг здійснити дзвінок потрібно спочатку авторизувати його на сервері телефонії. Під час реєстрації відправляється від клієнта на сервер повідомлення REGISTER, де клієнт вказує SIP URI та свою IP-адресу. Завдяки чому сервер може динамічно зв'язати SIP-ідентифікатор користувача (наприклад, sip:user@example.com) з поточним мережевим розташуванням. Сервер, своєю чергою, у відповідь надсилає відповідь 401 Unauthorized для того, аби аутентифікувати користувача, після чого користувач софтбокса заново надсилає запит REGISTER із іменем та паролем. Якщо авторизація успішна – сервер надсилає 200 OK та запам'ятовує зв'язку URI – IP для майбутньої маршрутизації. Якщо ж авторизація неуспішна – діалог буде завершено із відповіддю 401 Unauthorized. Для збереження актуальної інформації процес реєстрації може оновлюватись із певною частотою (частота залежить від налаштувань софтбоксу).

Після успішної реєстрації на сервері абонент отримує можливість ініціювати та приймати дзвінки. Встановлення виклику здійснюється за допомогою обміну SIP-повідомленнями, що створюють SIP-діалог — логічне з'єднання між

учасниками, під час якого відбувається узгодження параметрів мультимедійної сесії.

Спочатку від абонента А на сервер надсилається запит INVITE, який містить опис можливих параметрів (SDP). Сервер, що отримав INVITE – надсилає проміжні відповіді: 100 Trying та 180 Ringing, які, відповідно, означають, що запит отримано та що абонент В викликається (зазвичай супроводжується звуковим сигналом гудків).

Якщо абонент В приймає дзвінок – надсилається відповідь 200 OK, після чого сеанс вважається встановленим та відбувається обмін RTP. Якщо ж абонент В не може прийняти запит (наприклад, відхиляє виклик), абонентові А надійде відповідь про помилку на стороні клієнта (умовно, 486 Busy Here). При цьому, код відповіді буде такий, який нам відповість сервер постачальника зв'язку (наприклад, деякі постачальники зв'язку можуть віддавати 480 Temporarily Unavailable як 486 Busy Here).

Коли будь-яка із сторін вирішує завершити розмову надсилається повідомлення BYE, на яку інша сторона відповідає повідомленням 200 OK – сесія успішно завершена.

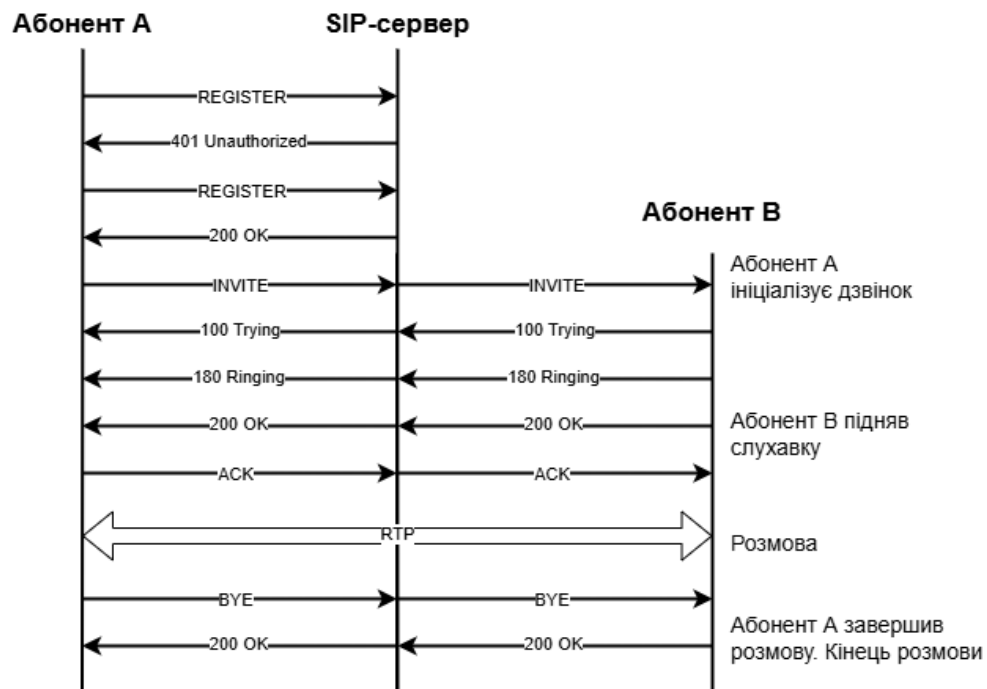


Рисунок 1.2 - Приклад базової сигналізації під час успішного дзвінка

Кожне SIP-повідомлення, відповідно стандарту RFC-3261 містить обов'язковий набір наступних заголовків:

Таблиця 1.2 - Обов'язкові заголовки SIP-повідомлень

Заголовок	Призначення	Приклад значення
Via	Вказує шлях доставки повідомлення, допомагає маршрутизувати відповіді	Via: SIP/2.0/UDP 192.168.1.100:5060;branch=z9hG4bKx9i38
From	Ідентифікує ініціатора виклику; містить тег для унікальності	From: "Alice" <sip:alice@client.com>;tag=1928301774
To	Ідентифікує адресата (того, кого викликають)	To: <sip:bob@example.com>
Call-ID	Унікальний ідентифікатор сесії виклику	Call-ID: a84b4c76e66710@client.com
CSeq	Номер послідовності запиту та його тип	CSeq: 314159 INVITE
Max-Forwards	Ліміт кількості проксі, через які може пройти запит	Max-Forwards: 70
Contact	Адреса, на яку треба надсилати відповіді або медіа-з'єднання	Contact: <sip:alice@192.168.1.100:5060>
Content-Type	Тип вмісту в тілі повідомлення (якщо є, наприклад, SDP)	Content-Type: application/sdp
Content-Length	Довжина тіла повідомлення у байтах	Content-Length: 142

1.4 Роль Asterisk та Kamailio у VoIP-мережах

Для побудови IP-телефонії важливо правильно розподілити функції між елементами архітектури, що забезпечують сигналізацію, маршрутизацію викликів та обробку мультимедійного трафіку. Найбільш популярним та практичним рішенням для реалізації VoIP-системи є архітектура, побудована на зв'язці **Asterisk + Kamailio**. Asterisk є універсальною автоматичною телефонною станцією (далі - АТС) з підтримкою SIP, службової логіки та медіаобробки, а Kamailio – високопродуктивний SIP-проксі сервер. Така інтеграція дозволяє реалізувати масштабовану, надійну та безпечну VoIP-інфраструктуру, що відповідає сучасним вимогам конфіденційності, доступності та керованості. Розглянемо обидва інструменти більш детально.

Asterisk – програмне рішення АТС (PBX, Private Branch Exchange), яке забезпечує повноцінний набір функціональних можливостей для побудови IP-телефонії та є основою VoIP-систем корпоративного рівня. Саме завдяки Asterisk реалізується маршрутизація дзвінків, автентифікація абонентів, підключення транків, обробку викликів та можливість реалізувати многорівневі IVR-меню, голосові черги дзвінків, функції колбеку, автоматичних продзвонів та аудіоанонсів.

Asterisk не має графічного інтерфейсу за замовчуванням і керується через конфігураційні файли або зовнішні API (наприклад, AMI, AGI або ARI), що дозволяє інтегрувати його в різноманітні бізнес-процеси та застосунки.

Asterisk Manager Interface (AMI) — це TCP-базований текстовий протокол, який дозволяє зовнішнім програмам віддалено керувати Asterisk у режимі реального часу. AMI надає доступ до внутрішніх подій та дозволяє виконувати керівні дії через спеціальні команди (Наприклад, створювати записи дзвінків, отримувати інформацію про поточні дзвінки, тощо). Саме AMI використовується для моніторингу навантаження операторів, або, наприклад, при CRM-інтеграціях.

Asterisk Gateway Interface (AGI) — це інтерфейс між Asterisk та зовнішніми скриптами. AGI дозволяє виконувати логіку викликів з-поза меж

діалплану. Завдяки йому є можливість виконувати IVR-системи з динамічним контентом або реалізовувати авторизаційні голосові сервіси.

Asterisk REST Interface (ARI) — API, який надає HTTP-доступ до внутрішньої логіки Asterisk, призначений для програмного управління медіа та логікою викликів. ARI дозволяє розробникам повністю контролювати виклики (наприклад, створювати WebRTC застосунки чи реалізовувати продвинуті голосові помічники).

Архітектура Asterisk побудована за модульним принципом. Головним елементом є ядро, яке координує взаємодію між окремими компонентами. Усі функції системи реалізуються через модулі, які можуть бути завантажені або відключені динамічно під час роботи.

Ядро зчитує конфігураційні файли, відповідає за побудову та обробку діалплану (сценарію логіки маршрутизації викликів), керує каналами та сесіями зв'язку та реалізує управління подіями та взаємодію із API.

Таблиця 1.3 - Основні типи модулів Asterisk

Категорія	Префікс	Призначення
Драйвери каналів	chan_	Взаємодія з зовнішніми пристроями та протоколами (SIP, IAX2, WebRTC)
Застосунки (аплікації)	app_	Реалізація прикладних функцій: голосова пошта, IVR, логіка викликів
Ресурси	res_	Бібліотеки функцій, що підтримують роботу протоколів, API, шифрування тощо
Інтерфейси діалплану	pbx_	Інтерпретація діалплану у різних форматах
Кодеки	codec_	Кодування та декодування аудіо
Формати медіа	format_	Обробка медіафайлів, зокрема збереження дзвінків у різних форматах

Діалплан (**Dialplan**) - серце логіки маршрутизації викликів, сценарій, у якому описано, як саме Asterisk повинен реагувати на вхідні або вихідні виклики, які дії необхідно виконати, які додатки або сервіси залучити та як управляти взаємодією між абонентами або зовнішніми системами. Фактично, діалплан виконує роль програмованого маршрутизатора викликів та реалізує всю бізнес-логіку обробки дзвінків. До прикладу, саме діалплан визначає, яким чином для абонента Б, який подзвонив на номер, заведений у АТС, буде програватися дзвінок (програвання аудіоанонсу, переадресація на операторів або IVR).

Kamailio – SIP-проксі сервер, який обробляє сигналізаційний трафік та виконує роль маршрутизатора SIP-повідомлень. На відміну від Asterisk, Kamailio обробляє лише SIP-повідомлення, не втручаючись у передавання медіа-трафіку (RTP). Саме він відповідає за встановлення, маршрутизацію та завершення SIP-сесій, забезпечує масштабованість і захищеність інфраструктури IP-телефонії [4].

SIP-проксі ідеально підходить для централізованого керування реєстрацією SIP-користувачів, реалізації політик автоентифікації, контролю доступу та антифлуд-захисту. Окрім того, завдяки модулю dispatcher є можливість динамічного розподілу запитів між кількома медіа-серверами(умовно, між кількома Asterisk`ами), що забезпечує високу доступність та відмовостійкість системи IP-телефонії.

Для побудови максимально відмовостійкої та масштабованої архітектури VoIP-системи, реалізується розподілення обов'язків між Kamailio та Asterisk. Kamailio виконує роль SIP-проксі-сервера, тобто обробляє сигналізацію та реалізує безпекові заходи, тоді як Asterisk виступає як медіасервер та обробляє аудіо.

1.5 Основні вразливості у VoIP-телекомунікаціях

В зв'язку із відкритістю стандартів, обширним використанням текстових протоколів та недостатнім впровадженням механізмів шифрування, VoIP-телекомунікації є доволі вразливими до немалої кількості загроз, які можуть

порушити конфіденційність, доступність та цілісність сервісів, що, своєю чергою, є критичним для бізнес-процесів організацій, які ставлять на меті впровадити систему IP-телефонії. При цьому, кожна з цих загроз є суттєвим ризиком для компанії, оскільки навіть кілька хвилин недоступності або хибної роботи сервісу можуть призвести не лише до втрати клієнтської довіри та погіршення репутації компанії, а й до прямих фінансових збитків. Особливо це стосується великих організацій із високим навантаженням на контакт-центри, де кожна хвилина простою операторів – це втрачені ліди, зірвані угоди, а отже, відповідно, втрачений прибуток. Окрім того, витік навіть мінімального обсягу конфіденційної інформації (зокрема, контактних даних або змісту розмов) може не лише до підриву довіри до компанії, а й використання таких даних для шахрайства або конкурентної розвідки. Що, своєю чергою, підкреслює критичну важливість побудови надійної безпеки VoIP-системи: від технічних налаштувань до внутрішнього контролю доступу.

Перш за все, однією із ключових уразливостей у IP-телефонії є протокол RTP. Як основний протокол, що використовується для передачі голосових аудіоданих у режимі реального часу, при відсутності належного захисту це робить його надзвичайно чутливим до атак, а отже великою проблемою для збереження конфіденційності та цілісності розмов [4].

Відсутність шифрування RTP-трафіку в стандартній реалізації відкриває можливості для пасивного перехоплення (eavesdropping). Будь-хто, хто має доступ до мережевого трафіку між абонентами, може без особливих зусиль (наприклад, за допомогою таких інструментів, як Wireshark чи tcpdump) відновити повний зміст розмови у відкритому вигляді. Це є прямим порушенням принципу конфіденційності й створює значний ризик витоку чутливої інформації.

Окрім того, варто враховувати, що протокол RTP за своєю суттю не передбачає жодних вбудованих механізмів автентифікації або авторизації учасників сесії. Унаслідок цього виникає серйозна проблема ідентифікації легітимного джерела аудіо-трафіку. Якщо злоумисник отримує доступ до

мережевої інфраструктури та зуміє спрямувати власні RTP-пакети до одного з кінцевих пунктів VoIP-з'єднання, система, яка не має захисних механізмів, може некоректно розпізнати це джерело як справжнього учасника сесії. У таких випадках можливі не лише втручання у розмову (man-in-the-middle), але й повне перенаправлення медіапотoku до зломисника, з подальшим його записом або модифікацією.

Загрозу посилює те, що у багатьох випадках ініціалізація RTP-сесій базується на інформації, переданій через протокол SIP у тілі SDP-повідомлення, яке також часто передається у відкритому вигляді, що, своєю чергою, дозволяє зломиснику, який здійснив перехоплення SIP-трафіку (навіть тимчасове), отримати всі необхідні параметри RTP-сесії, зокрема IP-адреси та порти.

Другою, не менш важливою загрозою у погано захищеній IP-телефонії є протокол SIP. Він є ключовим компонентом, який відповідальний за встановлення сесії, а отже загрози, пов'язані із цим протоколом напряду впливають на доступність та цілісність зв'язку у VoIP-системах.

За замовчуванням SIP працює поверх протоколів UDP або TCP, де повідомлення передаються у відкритому вигляді без належного шифрування. Що, своєю чергою, дає змогу зломиснику, який має доступ до мережі, легко перехоплювати SIP-запити та відповіді, які містять чутливу інформацію (імена користувачів, IP-адреси, незахищені порти та опис SDP-потоків. Як висновок – отримуємо витік автентифікаційних даних та доступ до облікових записів.

Окрім того, існує загроза атаки типу user enumeration, за якого зломисник шляхом сканування системи може виявити дійсні SIP-ідентифікатори транків та користувачів на VoIP сервері, а отже відкрити можливість до більш складних атак, наприклад brute-force, після якої, зломисник як і легітимний користувач отримає доступ до облікового запису, а отже зможе легко не тільки здійснювати дзвінки від чужого імені, а й приймати вхідні дзвінки, що може призвести не тільки витоку конфіденційної інформації, а й серйозних репутаційних та фінансових втрат, або, навіть, кримінального переслідування через фрод та спам виклики (дзвінки із номеру компанії із повідомленням про замінування шкіл або

критичної інфраструктури). Така атака зазвичай проводиться шляхом масового надсилання SIP-запитів із підставними SIP URI. Зловмисник аналізує відповіді сервера (401 Unauthorized, 403 Forbidden, 404 Not Found) для виявлення існуючих облікових записів [5].

Постійна доступність серверу – запорука стабільної роботи організації та її фінансового прибутку. Проте, VoIP-інфраструктура є доволі чутливою до атак типу відмова в обслуговуванні (Denial of Service, DoS). За допомогою автоматичних інструментів, по типу sipvicious або власноруч написаних скриптів зловмисник може створити одну із найпоширеніших атак на проксі- та медіасервер, а саме SIP Flooding, найбільш поширену реалізацію DoS у VoIP. Масовим надсиленням запитів типу INVITE та REGISTER зловмисник може створити падіння процесу Asterisk або банально перевантажити сигналізаційну підсистему SIP-проксі, що призведе до втрати можливості реєстрації легітимних користувачів, порушення маршрутизації викликів та виведення з ладу сервісів передачі голосу в реальному часі. Успішна реалізація DoS атак ставить під загрозу не лише технічну доступність VoIP-сервісів, але й завдає прямих фінансових збитків: втрачені дзвінки клієнтів, непродзвонені бази лідів та простій операторів контакт-центру.

Також, однією із популярних атак у VoIP-системах є атака типу spoofing (англ. підміна). Цей тип атаки полягає у підробці або фальсифікації ідентифікаційної інформації у SIP-повідомленнях з метою введення в оману мережу або абонентів. Найчастіше спуфінг реалізується через підміну заголовків From, To, Contact, Via та інших полів SIP-протоколу, що відповідають за адресацію, ідентифікацію та маршрутизацію дзвінків. Це, своєю чергою, може призвести до несанкціонованого доступу до системи, зловживання платними сервісами (завдаючи прямих фінансових збитків) та підміни автентичного трафіку [6].

Окрім вищеописаних зовнішніх ризиків, компанії, які впроваджують IP-телефонію, можуть зіштовхнутись також із високим рівнем уразливості до внутрішніх загроз, пов'язаних із недобросовісними співробітниками, недостатнім

розмежуванням прав доступу, відсутністю контролю дій користувачів і збереженням конфіденційних даних у незахищеному вигляді. Зокрема, мова йде про несанкціонований доступ до даних дзвінків, крадіжку баз контактів клієнтів та зловживанням адміністративними правами.

Для контролю якості роботи компанії та покращення бізнес-процесів, зокрема інтеграції із CRM-системами, компанії часто передбачають зберігання записів дзвінків. Проте, при відсутності належного обмеження доступу до файлів записів існує ризик несанкціонованого прослуховування чи копіювання записів розмов, що порушує вимоги до конфіденційності та створює юридичні ризики для компанії та її клієнтів.

В сучасних цифрових продажах база клієнтів компанії, так званих «теплих лідів» (англ. Lead), тобто контактів, які вже виявили зацікавленість у послугах чи товарах компанії – це високовартісний стратегічний актив. Її формування зазвичай супроводжується значним фінансовим ресурсом та займає немало часу. Витік або несанкціоноване копіювання такої бази у компанії, яка використовує рішення IP-телефонії становить серйозну бізнес-загрозу. Якщо конкуренти отримують доступ до такої інформації, це дає їм змогу заволодіти зацікавленими контактами, підірвати довіру клієнтів, або, взагалі, провести агресивну контропопозицію.

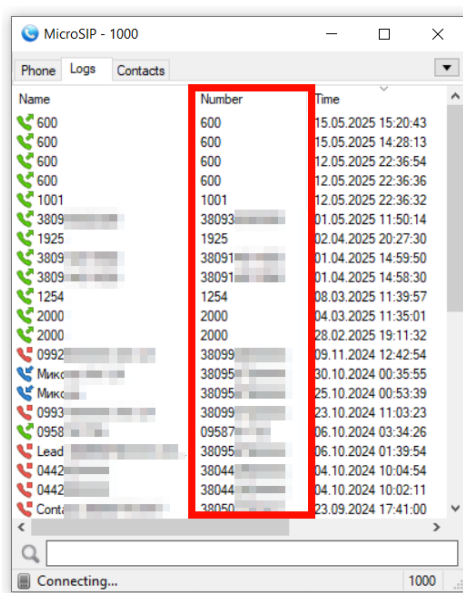


Рисунок 1.3 - Софтфон MicroSIP, який зберігає інформацію про історію викликів у відкритому вигляді (номери приховано у цілях конфіденційності)

Отже, загрози інформаційної безпеки висвітлюють критичну важливість недовіри навіть до внутрішніх користувачів у межах корпоративної VoIP-інфраструктури, оскільки саме внутрішній персонал із доступом до системи телефонії чи контактої бази клієнтів є основним вектором загроз витоку даних, тобто порушення конфіденційності.

Висновки до розділу 1

У першому розділі було систематизовано та розглянуто основні компоненти побудови VoIP-систем, ролі та функціональні можливості ключових складових архітектури - SIP-проксі-сервера Kamailio та медіасервера Asterisk. Окрім того, було розглянуто основні протоколи, на яких побудована робота VoIP-телекомунікацій, а саме – SIP, RTP, SRTP, SDP, їх ролі при здійсненні дзвінків та розглянуто їх основні уразливості.

Встановлено, що відкритість специфікацій, слабка автентифікація, відсутність шифрування та належної сегментації мережі сприяють виникненню широкого спектру загроз – від перехоплення трафіку до зловмисної реєстрації та DoS-атак. Особливо небезпечними є атаки типу Man-in-the-Middle, spoofing, enumeration, а також внутрішні загрози інформаційній безпеці, пов'язані з доступом до критичних бізнес-даних, таких як бази номерів та зміст розмов.

Отже, на основі проведеного аналізу зроблено висновок, що звичайного впровадження IP-телефонії, як засобу захищеної автоматизованої бізнес-комунікації недостатньо. Традиційні підходи впровадження IP-телефонії, хоча і мають переваги в швидкості та зручності комунікації, але не враховують сучасних викликів у сфері інформаційної та кібербезпеки. Це обґрунтовує важливість створення та впровадження захищеної системи, побудованої на принципах збереження конфіденційності, цілісності та доступності, у якій доступ до інфраструктури та критично важливих для бізнесу даних буде

обмежено не лише для зовнішніх, але й внутрішніх загроз. Об'єктивна необхідність впровадження такої концепції становить виклик, який доцільно може вирішити один із найперспективніших та найефективніших підходів сучасної кібербезпеки – реалізація моделі нульової довіри. На відміну від традиційних моделей безпеки, Zero Trust передбачає відсутність довіри навіть до внутрішніх користувачів та компонентів системи, необхідність постійного моніторингу та логування системи, багаторівневої автентифікації та жорсткої політики контролю доступу на всіх рівнях.

Впровадження Zero Trust у VoIP-телекомунікації є логічною відповіддю на виявлені уразливості й виклики, та є основою для побудови захищеної телекомунікаційної архітектури, що здатна забезпечити стійкість до атак і безпечне функціонування критично важливих комунікаційних сервісів у межах організації.

2 МОДЕЛЬ ZERO TRUST ЯК ОСНОВА БЕЗПЕЧНОЇ VOIP-АРХІТЕКТУРИ. ФОРМАЛЬНЕ ПРЕДСТАВЛЕННЯ РІШЕННЯ

Тендеції сьогодення встановлюють один із головних векторів для розвитку бізнесу - автоматизація бізнес-процесів. Це, своє чергою, ставить на мету впровадження захищеної копоративної інфраструктури для комунікації, як основного засобу взаємодії із клієнтами та розвитку компанії. Проте, як було розглянуто детально у першому розділі, базова реалізація такої системи при неналежному захисті є вкрай вразливою до дій зловимсників, як ззовні так і всередині інформаційної VoIP-системи.

Впровадження таких систем за класичною префиферійною (perimeter-based) моделлю безпеки, що базується на довірі до всіх суб'єктів усередині мережі, не відповідає сучасним викликам [7]. Таку модель можна уявити як підхід «замку з ровом», де міжмережеві екрани та VPN виконують роль захисних стін, охороняючи внутрішню корпоративну мережу від зовнішніх загроз. Така модель припускає, що усі пристрої та користувачі системи є легітимними, а отже, безпечними. Однак, після проникнення загрози всередину, така модель дозволяє зловмиснику безперешкодно переміщуватись по мережі. Також, така модель не враховує те, що загрозу для системи можуть становити внутрішні користувачі, що є особливо критичним для бізнес-інтересів компанії та може викликати великі втрати клієнтської довіри та подальших фінансових ризиків.

Натомість модель нульової довіри (англ. Zero Trust) базується на припущенні, що загроза може надійти як ззовні, так і зсередини. Вона заперечує саму ідею довіреної внутрішньої мережі: кожен користувач, пристрій чи мережевий потік повинні бути автентифіковані та авторизовані перед тим, як отримають доступ. Такий підхід зменшує ризик від зовнішніх атак шляхом обов'язкової аутентифікації та шифрування, забезпечує повну видимість взаємодії користувачів із системою за допомогою моніторингу та журналювання та контролює доступ користувачів до ресурсів та інформації у самій системі.

2.1 Концепція Zero Trust: принципи, вимоги та релевантність імплементації моделі для VoIP-телекомунікацій

Модель нульової довіри (англ. Zero Trust) наразі є однією із провідних парадигм сучасних інформаційних систем, яка не тільки влучно реагує на сучасні виклики кібербезпеки, а й пропонує системний підхід до побудови надійного захисту. Така модель передбачає постійний контроль та перевірку усіх користувачів та пристроїв, які намагаються отримати доступ до ресурсів системи, як ззовні, так і зсередини, що є доволі важливою частиною організації такої системи, як корпоративна IP-телефонія. Івацу

Zero Trust — це підхід до інформаційної та кібербезпеки, згідно якого при побудові системи за замовчуванням не варто довіряти нікому: ні внутрішнім користувачам, ні зовнішнім. Кожен з користувачів, який хоче отримати доступ спершу має довести свою надійність. Згідно такого підходу, доступ повинен суворо контролюватись незалежно від місця розташування користувач чи пристрою та піддаватись постійним перевіркам.

Існуючі безпекові правила у традиційних периферійних моделях безпеки, такі як фаєрволи або антивіруси, хоча ефективно блокують зовнішні атаки, але недостатньо захищають систему від внутрішніх загроз всередині корпоративної інфраструктури. Саме тому, метою моделі Zero Trust є створення середовища, де жодна сутність, незалежно від її розташування у мережі, не отримує повної довіри, а доступ до будь-якого ресурсу надається виключно на основі перевірки ідентичності і мінімально необхідних прав.

Основними принципами моделі нульової довіри є правило «Ніколи не довіряй, завжди перевіряй», принцип найменших привілеїв, мікросегментація та постійний збір інформації про стан системи задля виявлення загроз та побудови гнучкої моделі захисту.

Таким чином, система, побудована на принципах моделі нульової довіри буде зашифрованою, увесь трафік, як зовнішній, так і внутрішній буде передаватись через TLS, аби виключити можливість пасивного прослуховування та перехоплення вмісту пакетів у мережі. У такій системі довіра є динамічною

змінною, що повинна постійно перевірятись та переглядатись: кожен користувач або пристрій у системі є потенційно небезпечним та підлягає перевірці ідентичності, авторизації та контексту дій. Навіть отримавши доступ до ресурсів, користувачі отримують доступ лише до тих ресурсів, які є критично необхідними для виконання їх функцій, що, своєю чергою, запобігає зловживанню повноваженнями та обмежує можливості атак. Але, навіть отримавши мінімальні привілеї жоден запит та жодна дія не проходить непоміченою. У системі, яка побудована на принципах Zero Trust відбувається логування та централізований аудит усіх дій у системі. Окрім того, рекомендований поділ мережі на ізольовані сегменти із різними політиками доступу.

2.2 Проблеми безпеки VoIP та їх подолання за допомогою принципів Zero Trust

Коропативний зв'язок є критично важливим інструменту розвитку бізнесу та його взаємодії із клієнтами, а отже і фінансового збагачення. У разі найменшої компрометації системи, організація може понести величезні репутаційні та фінансові втрати, а отже, виникає потреба у мінімізації ризиків як ззовні, так і всередині корпоративної IP-телефонії. Модель нульової довіри є чудовим рішенням для побудови такої системи, оскільки враховує не тільки загрози від зовнішніх користувачів, а й обмежує вплив внутрішніх користувачів. Це є важливою перевагою Zero Trust орієнтованої системи перед системою із імplementованою периферійною моделлю безпеки, оскільки як було встановлено вище, внутрішні загрози від недобросовісних співробітників є однією із найбільших загроз інформаційній та кібербезпеці корпоративного VoIP рішення.

Однією з основних проблем у VoIP-системах є ризик перехоплення трафіку. У стандартних реалізаціях SIP-протоколу сигналізаційні повідомлення передаються у відкритому текстовому вигляді через транспортний протокол UDP або TCP. Це створює загрозу для компрометації облікових даних, захоплення SIP-ідентифікаторів, прослуховування викликів або підміни

викликів через атаки типу Man-in-the-Middle. Модель Zero Trust наполягає на повному шифруванні всього трафіку, а отже стає необхідним шифрування сигналізації (SIP over TLS) для забезпечення конфіденційності та цілісності SIP-запитів, унеможливаючи зчитування або зміну даних третіми сторонами. Окрім того, шифрується увесь медіатрафік, замість стандартного RTP потоку, голос передається через його захищений аналог SRTP, що унеможливує пасивне перехоплення та подальше відтворення медіасесії. Таке рішення закриває основну загрозу конфіденційності для корпоративних дзвінків та захищає зміст та автентичність розмов у режимі реального часу [8].

Модель нульової довіри наполягає не тільки на захисті внутрішньої мережі, а й розвиває ідею захисту системи від зовнішнього впливу. Отже, перш за все обмежується можливість автентифікації. IP-адреси класифікуються як довірені та недовірені. Звичайно ж, навіть довірені адреси все одно підлягають моніторингу та перевірці автентичності, але для untrusted IP-адрес діє жорстка політика: за замовчуванням усі запити, окрім явно дозволених блокуються, кожен User-Agent перевіряється на відповідність whitelist, а кількість спроб автентифікації є обмеженою. Таке обмеження захищає систему від одного із найбільш поширених векторів атак на VoIP-системи, а саме brute-force атак на SIP-облікові записи через багаторазове підбирання паролів або використання автоматизованих сканерів. Встановлюється жорстке обмеження на рівні SIP-проксі при досягненні N невдалих спроб, за його досягнення система автоматично вносить IP-адресу до тимчасового або постійного чорного списку та логує такий інцидент у центральній системі моніторингу.

Всі користувачі системи, які мають доступ для здійснення дзвінків повинні проходити обов'язкову багатофакторну автентифікацію (MFA), для точного підтвердження ідентичності користувача, який хоче здійснити ту, чи іншу дію. Кожна сесія оцінюється за політиками активності, які визначають допустиму кількість неактивних секунд, допустимі типи повідомлень і дозволені напрямки викликів.

Для запобігання зловживанням тривалих SIP-сесій впроваджується механізм автоматичного виведення з системи після визначеного інтервалу неактивності, що є критично важливим для запобігання утриманню привілеїв у компроментованих клієнтів та повторному використанню старих реєстрацій. Кожна сесія, як і її завершення, має бути зафіксована у централізованій системі логування з прив'язкою до користувача, User-agent, часу та причини завершення.

Для запобігання атак класу user enumeration та подальшого використання отриманої інформації для brute-force (зловмисник надсилає численні SIP REGISTER або INVITE запити з різними To: / Request-URI полями, а потім аналізує відповіді сервера: наприклад, 401 Unauthorized або 404 Not Found), рекомендовано уніфікувати відповіді на успішні та неуспішні запити. Наприклад, на рівні Kamailio налаштовується відповідь на запит 401 Unauthorized, навіть якщо акаунту не існує. Окрім того, рекомендовано фільтрацію за User-Agent і заголовками. Атаки такого типу зазвичай здійснюються інструментами з характерними заголовками (sipvicious, friendly-scanner, тощо). Такі запити або одразу блокуються та логуються для подальшого аналізу.

Внутрішні інформаційні загрози, які у традиційній периферійній моделі безпеки залишаються непомітними, при реалізації моделі нульової довіри повинні бути нейтралізованими та залогованими [9]. Користувач системи є потенційно небезпечним, а доступ до ресурсів здійснюється виключно за перевірки повноважень та контексту. Систему, а особливо критично чутливу інформацію, як номери телефонів, потрібно приховувати від звичайних операторів. Можливість дзвонити користувач отримує не просто завівши робочу SIP-лінію у софтфон, а отримує можливість дзвонити лише по кліку із програми, де першочергово було перевірено його автентичність. Окрім того, у такій програмі за замовчуванням приховано номер абонента, тобто оператор не здійснює вихідний дзвінок на повний номер телефону, а на його лінію надходить вхідний дзвінок із прихованим номером у форматі +38095879XXXX, що унеможлиблює крадіжку номеру та передачу такої інформації у рамках

конкретної розвідки. Номери телефону клієнтів доступні лише користувачам із відповідним явним дозволом. Вхідні виклики для абонентів гарячої лінії так само приховані і відкриваються лише для перевірених користувачів (які не тільки пройшли ідентифікацію, але й мають на це явні дозволи на рівні як системи, так і на рівні репутації у компанії). Також, рекомендовано дозволити користувачам дзвонити лише із перевірених софтонів, тобто створити строгу фільтрацію із дозволеними User-agent.

У реалізації архітектури VoIP відповідно до принципів Zero Trust критично важливу роль відіграє модульний SIP-проксі-сервер Kamailio. Саме він слугує інструментом обробки SIP-сигналізації і фактично виступає першою лінією оборони від зовнішніх атак, основною точкою для контролю й маршрутизації SIP-повідомлень. Kamailio дозволяє реалізувати концепцію «мінімально необхідного доступу», де всі запити проходять багаторівневу перевірку перед тим, як отримати дозвіл на взаємодію з медіасервісами або базами даних користувачів.

Одним з ключових механізмів контролю у Kamailio є списки контролю доступу (Access Control Lists, ACL). Вони дозволяють обмежити вхідні з'єднання до строго визначеного переліку IP-адрес або мережевих підмереж, з яких дозволяється ініціація SIP-повідомлень.

У фізичній реалізації архітектури Kamailio зазвичай розміщується в демілітаризованій зоні (DMZ), ізольованому сегменті мережі, який виконує роль буфера між зовнішнім середовищем (Інтернет) та внутрішніми сервісами (наприклад, медіасервери Asterisk, бази даних). Мікросегментація дозволяє створити чіткі мережеві межі: Kamailio взаємодіє з Asterisk виключно через заздалегідь визначені порти, при цьому такі порти потрібно обирати відмінними від стандартних 5060 та 5061. Це забезпечує не лише логічну, а й фізичну ізоляцію між компонентами, унеможливлюючи прямий доступ зломисника до медіасерверу. Окрім того, на самому Asterisk впроваджується механізм IP-білого списку, який відкидає всі запити, що не надходять із Kamailio.

Кожен запит, який надходить на SIP-проксі не може вважатися безпечним за замовчуванням. Натомість він підлягає суворій ідентифікації, автентифікації та авторизації згідно з попередньо визначеними політиками доступу. Зокрема, на самому Kamailio можна впровадити ефективний захист від DOS атак. За допомогою модуля `pike` можна аналізувати частоту запитів з IP-адреси та якщо кількість перевищує певний поріг тимчасово блокувати IP-адресу. Окрім того, на Kamailio є можливість автоматично блокувати IP-адреси, якщо було введено неіснуючу SIP-лінію (захист від enumeration), превентивно захищатись від атак brute-force шляхом блокування зв'язки IP + SIP після певної кількості невдалих спроб автентифікації. Такі налаштування унеможливають як масовий підбір реєстраційних даних, так і проведення enumeration-атак для визначення наявних у системі SIP-акаунтів. Можливість ідентичних відповідей для захисту від SIP-enumeration легко реалізуються внесенням змін через логіку обробки у конфігураційних файлах [10].

Проте, лише простої фільтрації на рівні SIP-проксі та перевірки доступів користувачів шляхом постійного підтвердження автентичності та ідентичності недостатньо. Згідно принципів моделі нульової довіри, кожна дія, здійснена у системі повинна бути відстежена та занесена у журнал аудиту у режимі реального часу. Така умова є принциповою: моніторинг системи повинен проводитись постійно та безумовно.

Для логування та аналізу VoIP-дзвінків доцільно використовувати інструмент VoIPmonitor. Цей інструмент дозволяє проводити глибокий аналіз RTP-трафіку, SIP-сесій, затримок, втрат пакетів і якості голосу. Система дає змогу не лише вести журнал SIP-реєстрацій і дзвінків, а й оперативно виявляти аномальні шаблони поведінки. Окрім того, завдяки цьому інструменту можна легко виявити складнощі на стороні постачальників номерів для їх подальшого детального аналізу та можливості продовжувати безупинну роботу системи.

Паралельно до аналізу VoIP-трафіку, доцільно впроваджувати централізовану систему логування на базі стеку ELK (Elasticsearch, Logstash, Kibana). За допомогою `rsyslog` Kamailio може надсилати структуровані

повідомлення про всі запити, реєстрації, автентифікаційні помилки, блокування IP та інші події до Logstash, де вони уніфікуються, нормалізуються та індексуються в Elasticsearch. Надалі цю інформацію можна зручно знайти та візуалізувати в Kibana. Перевагами такого підходу, який є однією із основ моделі нульової довіри є відстеження усіх дій у системі: починаючи від авторизації користувачів на бекенді, закінчуючи алертами на спроби DOS атак, невдалих авторизацій та можливості подальшої поведінкової аналітики для виявлення відхилень від нормального трафіку. Окрім того, можлива інтеграція системи логування із сторонніми сервісами для швидкого оповіщення відповідальних осіб стосовно інцидентів у системі. Логування є критично важливим у IP-телефонії організації, як з точки зору кібербезпеки, так і з точки зору інформаційної безпеки, оскільки саме постійний моніторинг системи реалізує оперативну видимість усіх подій та дозволяє зрозуміти причини інцидентів та можливі вектори загроз. Такий підхід дозволяє інформувати та аналізувати не тільки дії зловмисників ззовні, а й розуміти ситуацію зсередини системи, перехоплювати дії працівників та давати змогу відтворити повний хід дій при інциденті.

2.3 Формальна архітектура

Для впровадження моделі Zero Trust у корпоративну VoIP-інфраструктуру, надзвичайно важливим є не лише реалізація політик безпеки, але й правильний вибір архітектурної моделі, яка дозволяє забезпечити суворий контроль доступу, сегментацію та моніторинг усіх взаємодій між компонентами системи. Ключовими технологічними елементами у побудові такої архітектури виступають SIP-проксі-сервер Kamailio та медіасервер Asterisk, кожен із яких виконує чітко визначені функції в межах захищеної та розмежованої комунікаційної інфраструктури.

Архітектура побудована за принципом розділення зон відповідальності, що реалізується через DMZ-сегментацію. Kamailio розгортається у ізольованій мережевій зоні, що слугує буфером між Інтернетом та критичними внутрішніми

сервісами (Asterisk, база даних, система логування). Це дозволяє ізолювати зовнішній трафік від медіа-серверів, а також централізувати всі контрольні точки перевірки у Kamailio. Таке архітектурне рішення є величезним плюсом: доступ до медіасервісів отримують лише перевірені та авторизовані клієнти, при цьому усі сторонні user агенти, як наприклад сканери мережі чи злоумисники не пройдуть перевірку на SIP-проксі та будуть заблоковані. Це імплементує перевірку автентичності запитів та захищає медіасервер Asterisk від можливих атак: brute-force, flood, spoofing, тощо. Іде жорстка перевірка User-Agent, а Asterisk не доступен напряму ззовні. Усі взаємодії з ним здійснюються виключно через Kamailio, який виконує функції шлюзу, контролера доступу та системи перевірки автентичності. Лише після успішної авторизації Kamailio дозволяє доступ до медіасерверу, при успішній побудові сеансу передача медіатрафіку відбувається лише через зашифровані пакети.

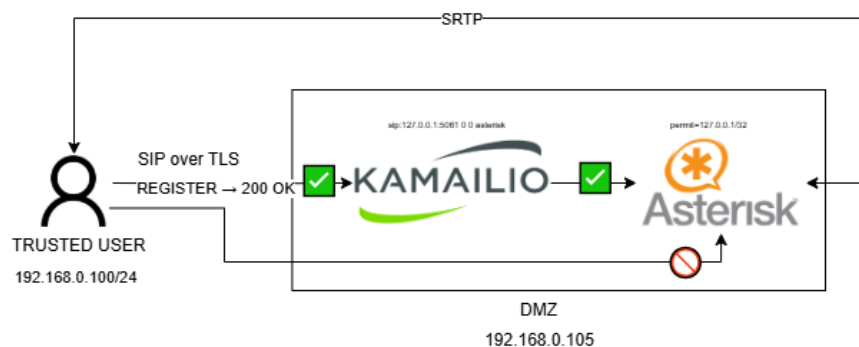


Рисунок 2.1 - Трафік від довіреного користувача проходить лише через SIP проху сервер

При цьому, згідно логіки Zero Trust, простої мікросегментації медіасерверу недостатньо. Доступ до Kamailio не повинен отримати неперевірений користувач, а той користувач, який отримав доступ не повинен бути довіреним, його привілеї повинні бути зведеними до мінімуму. Тому, усі дзвінки здійснюються не просто через софтфон, а лише після попередньої авторизації користувача на сайті, де, згідно рольового розмежування та прав доступу користувачу надається можливість здійснювати дзвінки. Такі дзвінки оригінуються лише по кліку та надходять на перевірений софтфон із одразу прихованим номером абонента, аби забезпечити конфіденційність номерів

абонентів та недопустити їх подальшого витоку. При цьому, користувачі, які не авторизувались напряму із браузера не можуть здійснити вихідний виклик, навіть маючи доступ до заведеного транку. На рівні проксі-сервера сторонні спроби здійснити виклик обробляються із однаковою відповіддю – 403 Forbidden, що не просто не допускає можливості здійснювати дзвінки без попередньої ідентифікації прав доступу, а й виключає можливість enumeration атак злоумисниками.

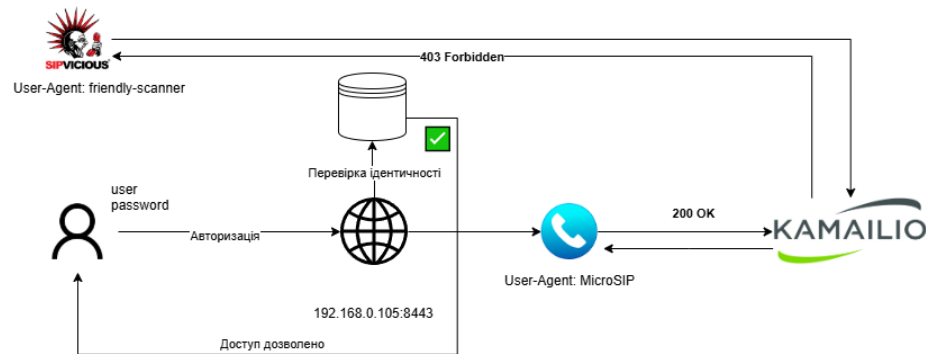


Рисунок 2.2 - Приблизна схема автентифікації довіреного користувача

2.4 Очікувані переваги імплементації моделі нульової довіри при побудові VoIP-системи

Імплементація моделі нульової довіри у сфері IP-телефонії реалізує принципово важливий підхід до захисту системи, який не просто нейтралізує зовнішні загрози, а й дозволяє обмежити вплив всередині, що дозволяє реалізувати ефективну нейтралізацію загроз та постійний моніторинг системи. Жоден користувач не є довіреним за замовчуванням, кожен запит на доступ до ресурсу має бути строго перевірений та авторизований, кожна дія, яка була зроблена у системі має бути занесена у журнал подій та у випадку підозрілої ефективності превентивно заблокована.

Всебічне шифрування сигналізаційного та медіа-трафіку усуває можливе прослуховування та компрометації інформації внаслідок атак типу Man-in-the-Middle. Використання таких технологій, як SIP over TLS для сигналізації та SRTP для передачі голосу, забезпечує конфіденційність, цілісність та автентичність комунікації у режимі реального часу.

Реалізація суворих механізмів автентифікації та авторизації, включаючи багатофакторну автентифікацію (MFA) та обмеження прав доступу до системи надає можливість захистити систему від зловживання доступом та навіть при фізичному доступі до клієнтського пристрою зловмисник не отримає можливості зловживати ресурсами без проходження усіх рівнів перевірки. Якщо користувач не підтвердив свою особу через веб-інтерфейс із наданими правами доступу йому не будуть доступні вихідні виклики. Усі користувачі системи розглядаються як потенційно недовірені: їм дозволяється виконувати лише ті дії, які прямо випливають із наданих їм ролей і прав. Система автоматичного завершення сесій при перевищенні тайм-ауту неактивності виключає можливість несанкціонованого збереження активних привілеїв на компрометованих пристроях.

Реалізується жорстке розмежування зовнішнього та внутрішнього трафіку, що виключає прямий доступ до медіасерверу Asterisk лише через SIP-проксі сервер. Крім того, підтримується уніфікація відповідей на некоректні SIP-запити, що нейтралізує атаки типу enumeration.

Окрім того, впроваджується постійний аудит та централізоване логування, реалізоване за допомогою систем VoIPmonitor та стеку ELK (Elasticsearch, Logstash, Kibana). Це дозволяє не лише реєструвати всі сесії, виклики та спроби автентифікації, але й оперативно виявляти аномалії, аналізувати якість сервісу та швидко реагувати на інциденти безпеки, які були викликані як щовнішніми, так і внутрішніми загрозами.

Висновки до розділу 2

Вимоги до інформаційної та кібербезпеки зростають, а отже постає питання у впровадженні дієвого рішення, яке дозволить впровадити гнучкий захист системи. Особливо гостро це питання постає при впровадженні корпоративної IP-телефонії, оскільки при запровадженні незахищених VoIP-системи зростають ризики збереження доступності та цілісності системи та конфіденційності голосового та сигналізаційного трафіку. При чому, загрози

зазвичай є не тільки зовнішніми, а й виникають всередині системи, зокрема від недобросовісних користувачів.

Концепція Zero Trust знімає припущення, що трафік та запити всередині системи є безпечними, що мінімізує ризики зловживань та витоків даних, як із зовнішнього, так і з внутрішнього середовища.

Завдяки впровадженню Kamailio як SIP-проксі у демілітаризованій зоні, забезпечується повноцінне відокремлення зовнішнього трафіку від критичних внутрішніх компонентів, наприклад від медіасерверу Asterisk. Використання мікросегментації дозволяє реалізувати контрольований доступ до кожного елементу системи, а обов'язкове шифрування трафіку (SIP over TLS та SRTP) гарантує захист від атак типу Man-in-the-Middle та несанкціонованого перехоплення даних.

Уніфікація відповідей на некоректні SIP-запити дозволяє нейтралізувати атаки типу user enumeration, які в незахищених системах дозволяють зловмиснику виявити існуючі акаунти через відмінності в кодах помилок (наприклад, 401 vs 404). У Kamailio впроваджується логіка однакових відповідей незалежно від того, існує обліковий запис чи ні, що унеможлиблює інформаційне зондування системи.

Особлива увага у моделі приділяється не лише автентифікації, але й авторизації та управлінню привілеями, що в сукупності унеможлиблює використання навіть скомпрометованих облікових даних без проходження додаткових рівнів перевірки. Захист від brute-force, спам-дзвінків, enumeration-атак, а також постійний аудит активностей реалізовано через логіку Kamailio, VoIPmonitor та стек ELK, що створює цілісну систему виявлення, реагування та протидії інцидентам безпеки.

Отже, імплементація Zero Trust у VoIP-інфраструктурі дозволяє досягти високого рівня захищеності, який не лише відповідає сучасним вимогам, але й надає компанії операційну стійкість, гнучкість у масштабуванні та здатність швидко адаптуватися до нових викликів у сфері кіберзагроз.

3 ПРАКТИЧНА РЕАЛІЗАЦІЯ ТА ПОРІВНЯЛЬНИЙ АНАЛІЗ

Доцільність застосування моделі Zero Trust у VoIP-системах було теоретично обґрунтовано у минулому розділі, а отже наступним кроком стає практичне впровадження відповідної архітектури, що враховує всі виявлені загрози, вразливості та вимоги до інформаційної безпеки.

У цьому розділі описано поетапне розгортання тестового середовища: налаштування медіасервера Asterisk, SIP-проксі сервера Kamailio, інструментів логування та моніторингу. Особливу увагу приділено налаштуванню конфігураційних файлів Kamailio та Asterisk, де внесені зміни для забезпечення шифрування трафіку, автентифікації та маршрутизації сигналізаційного трафіку та захисту системи від атак типу Denial of Service, brute-force та enumeration. Також, було розроблено застосунок із вбудованим розмежуванням прав доступу для впровадження принципу найменших привілеїв для користувачів телефонії.

Завершальним етапом розділу є проведення порівняльного аналізу: впроваджена модель Zero Trust розглядається у порівнянні з типовою VoIP-системою без засобів контролю доступу, шифрування та логування, що дозволяє наочно продемонструвати переваги реалізованого підходу.

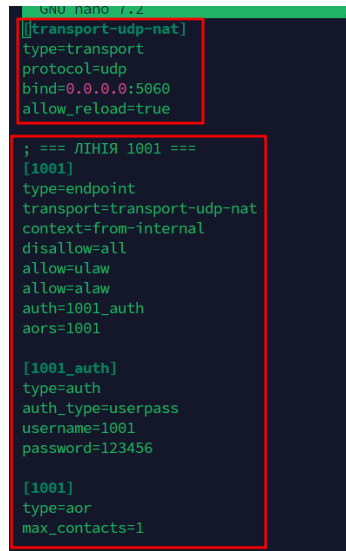
3.1 Розгортання базової VoIP-архітектури та впровадження у неї моделі Zero Trust

Перед впровадженням моделі нульової безпеки необхідно розгорнути програмні рішення корпоративної телефонії. Першим етапом при розгортанні такої системи є налаштування медіасервера Asterisk. Спочатку встановлюється сама утиліта, далі налаштовуються конфігураційні файли, а саме `pjsip.conf`, `extensions.conf`.

Фактично, налаштовується діалплан та створюються користувачі, які матимуть доступ для здійснення дзвінків. Окрім того, важливим етапом є заведення транкового номеру, який розширить можливість здійснювати дзвінки

не тільки між внутрішніми користувачами (умовно, 1001 та 1002), а й здійснювати дзвінки на клієнтські номери.

Кожен внутрішній користувач має тип сутності `type=endpoint`, у той час як транк має `type=registration`, заведені користувачі та транк заповнюються у файлі `pjsip.conf`. Мінімально налаштований діалплан `extensions.conf` та `pjsip.conf` мають наступний вигляд:



```

[transport-udp-nat]
type=transport
protocol=udp
bind=0.0.0.0:5060
allow_reload=true

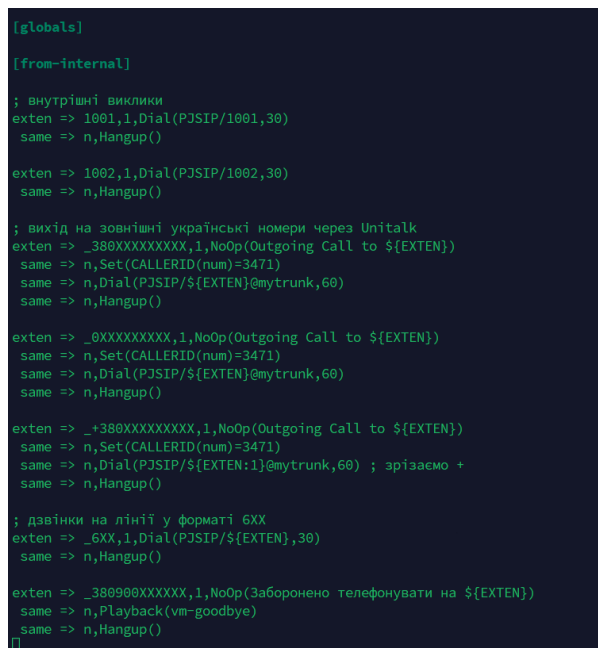
; === ЛІНІЯ 1001 ===
[1001]
type=endpoint
transport=transport-udp-nat
context=from-internal
disallow=all
allow=ulaw
allow=alaw
auth=1001_auth
aors=1001

[1001_auth]
type=auth
auth_type=userpass
username=1001
password=123456

[1001]
type=aor
max_contacts=1
  
```

Рисунок 3.1 - Конфігураційний файл `pjsip.conf`

У самому діалплані налаштуємо можливість дзвонити лише на українські номери та виключимо можливість дзвінків на платні номери 380900:



```

[globals]

[from-internal]

; внутрішні виклики
exten => 1001,1,Dial(PJSIP/1001,30)
same => n,Hangup()

exten => 1002,1,Dial(PJSIP/1002,30)
same => n,Hangup()

; вихід на зовнішні українські номери через Unitalk
exten => _380XXXXXXX,1,NoOp(Outgoing Call to ${EXTEN})
same => n,Set(CALLERID(num)=3471)
same => n,Dial(PJSIP/${EXTEN}@mytrunk,60)
same => n,Hangup()

exten => _0XXXXXXX,1,NoOp(Outgoing Call to ${EXTEN})
same => n,Set(CALLERID(num)=3471)
same => n,Dial(PJSIP/${EXTEN}@mytrunk,60)
same => n,Hangup()

exten => _+380XXXXXXX,1,NoOp(Outgoing Call to ${EXTEN})
same => n,Set(CALLERID(num)=3471)
same => n,Dial(PJSIP/${EXTEN:1}@mytrunk,60) ; зрізаємо +
same => n,Hangup()

; дзвінки на лінії у форматі 6XX
exten => _6XX,1,Dial(PJSIP/${EXTEN},30)
same => n,Hangup()

exten => _380900XXXXXX,1,NoOp(Заборонено телефонувати на ${EXTEN})
same => n,Playback(vm-goodbye)
same => n,Hangup()
  
```

Рисунок 3.2 - Діалплан

Проте, практика зберігання паролів у відкритому вигляді у конфігураційному файлі є дуже ненадійним та небезпечним рішенням, тому реалізуємо зберігання даних для авторизації у окремій базі даних. Також, одразу реалізуємо шифрування трафіку, оскільки при типі транспорту UDP можна легко перехопити сигналізаційний трафік, а при передачі медіапотоків через RTP, маючи доступ до сніфферу мережевого трафіку, наприклад Wireshark, будь-хто, хто має доступ до мережі може легко перехопити зміст розмов. Зокрема, прослуховуючи потрібний мережевий інтерфейс можна легко відтворити аудіопотоки через інструмент «Телефонія – Виклики VoIP – Play Streams».

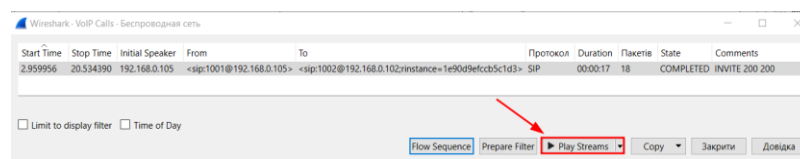


Рисунок 3.3 - Можливість прослухати перехоплений RTP потік

Перехоплена розмова виглядатиме наступним чином, її можна спокійно прослухати чи завантажити:

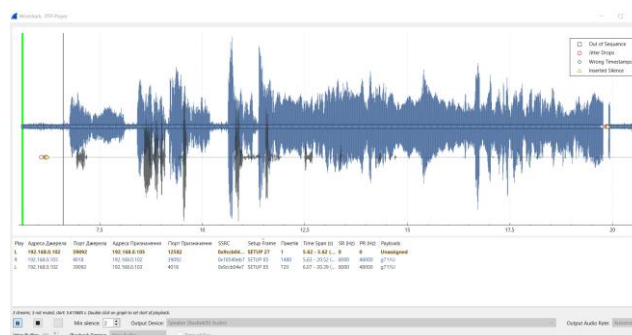


Рисунок 3.4 - Відтворення вмісту перехопленої розмови

Тому, виконуємо зміни у конфігураційних файлах, налаштовуємо передачу сигнального трафіку через TLS, а медіатрафік передаємо через SRTP.

```
[transport-tls]
type=transport
protocol=tls
bind=0.0.0.0:5061
cert_file=/etc/asterisk/keys/asterisk.pem
priv_key_file=/etc/asterisk/keys/asterisk.key
method=tlsv1_2
verify_client=no
require_client_cert=no
allow_reload=yes
```

Рисунок 3.5 - Увімкнене шифрування сигналізаційного трафіку

Окрім шифрування сигналізаційного трафіку важливо для кожної сутності endpoint прописати тип транспорту TLS та налаштувати шифрування медіатрафіку, перед цим попередньо згенерувавши сертифікат та ключ:

```
; === ЛІНІЯ 1001 ===
[1001]
type=endpoint
transport=transport-tls-nat
context=from-internal
disallow=all
allow=ulaw
allow=alaw
auth=1001_auth
aors=1001
media_encryption=dtls
media_encryption_optimistic=yes
dtls_verify=fingerprint
dtls_cert_file=/etc/asterisk/keys/asterisk.pem
dtls_private_key=/etc/asterisk/keys/asterisk.key
dtls_setup=actpass
[]
```

Рисунок 3.6 - Увімкнене шифрування медіатрафіку

У висновку, внесені зміни у `pjsip.conf` забезпечують шифрування як сигналізаційного трафіку (SIP), так і медіатрафіку (RTP), що реалізує один із основних принципів моделі Zero Trust. Передача сигналів виклику через протокол TLS унеможлиблює їх перехоплення або модифікацію на етапі встановлення дзвінка, а використання SRTP для медіапотоків дозволяє шифрувати сам голосовий вміст, виключаючи можливість несанкціонованого прослуховування зломисниками навіть у разі фізичного доступу до мережевого сегменту.

Після забезпечення шифрування та здійснення основних заходів безпеки конфігураційних файлів необхідно налаштувати SIP-проху сервер Kamailio, який стане основною захищеної системи. Усі основні налаштування проводяться у конфігураційному файлі `kamailio.cfg`.

Логіка роботи SIP проху аналогічна до тої, яку було формально описано у 2 розділі. Конфігурація Kamailio ґрунтується на парадигмі Zero Trust: кожен запит перевіряється незалежно від походження, а доступ надається виключно за підтвердженими політиками.

```

# --- Перевірка на IP-бан ---
if ($sht(ipban=>$si)!=null) {
    xlog("L_WARN", "Request from banned IP $si\n");
    exit;
}

# --- Антифлуд-захист ---
if (!pike_check_req()) {
    $sht(ipban=>$si) = 1;
    xlog("L_ALERT", "PIKE: blocking $rm from $si:$sp\n");
    exit;
}

# --- Детектори сканерів SIP ---
if ($ua =~ "friendly-scanner|sipvicious|sipcli|pplsip|iWar|VaxSIP") {
    xlog("L_WARN", "Scanner detected $ua from $si, blocking\n");
    exit;
}

# --- Захист від SQL-ін'єкцій ---
if (is_method("REGISTER|INVITE") && $au =~ "(=)|(--)|('|#|%27|%24)" && $au != $null) {
    xlog("L_WARN", "SQLi attack detected from $si\n");
    exit;
}

```

Рисунок 3.7 – Реалізація механізмів безпеки у Kamailio: перевірка IP, User-Agent, SQL-ін'єкцій та обмеження автентифікації

Перший з наведених блоків конфігурації Kamailio реалізує перевірку IP-адреси клієнта у хеш-таблиці \$sht(ipban), що містить адреси, які були попередньо заблоковані. Якщо IP клієнта наявний у таблиці, запит відхиляється з відповідним логом.

Другий блок забезпечує захист від флуда з використанням модуля pike. Механізм дає змогу запобігти атакам типу DoS, що реалізуються через надмірну кількість SIP-пакетів.

Третій блок виконує ідентифікацію сканерів протоколу SIP, таких як sipvicious, sipcli, friendly-scanner тощо. Визначення базується на регулярному виразі, який порівнюється з заголовком User-Agent (\$ua). У разі збігу запит миттєво блокується, оскільки він, найімовірніше, надійшов від автоматизованого інструмента для сканування або злому.

Останній блок призначений для виявлення SQL-ін'єкцій у полі \$au (автентифікаційне ім'я користувача). Регулярний вираз визначає характерні ознаки SQL-ін'єкцій, такі як символи ', --, =, #, %27, %24, які часто

використовуються в ін'єкційних запитах. Якщо відповідні шаблони виявлені, запит одразу відкидається і фіксується у логах.

Також, реалізовано блокування після 3-х невдалих спроб автентифікації:

```
# --- Обробка автентифікації з баном після 3-х спроб ---
route[AUTH] {
    if (!www_authorize("kamailio.org", "subscriber")) {
        $var(attempts) = $sht(authfail=>$au);
        if ($var(attempts) == $null) $var(attempts) = 0;
        $var(attempts) = $var(attempts) + 1;
        $sht(authfail=>$au) = $var(attempts);

        if ($var(attempts) >= 3) {
            # Банимо користувача, юзер-агент та IP на годину
            $sht(authban=>$au) = 1;
            $sht(ipban=>$si) = 1;
            xlog("L_ALERT", "User $au permanently banned from $si after 3 failed attempts\n");
        }

        www_challenge("kamailio.org", "0");
        exit;
    }

    # Скидуємо лічильник при успішній автентифікації
    $sht(authfail=>$au) = 0;
    consume_credentials();
    if (!lookup("location")) {
        sl_send_reply("401", "Unauthorized");
        exit;
    }
}
```

Рисунок 3.8 – Обробка 3-х невдалих спроб автентифікації

Окрім того, у Kamailio описується логіка маршрутизації успішних викликів на Asterisk. Детально ознайомитись із конфігураційним файлом kamailio.cfg можна у Додатку Б.

Сам Kamailio розміщується у DMZ-сегменті і є єдиною точкою доступу для SIP-клієнтів. Всі абоненти реєструються через Kamailio, який після перевірок та авторизації передає трафік до внутрішнього медіасервера (Asterisk), ізолюваного на окремій мережі. Між Kamailio та Asterisk діє політика мережевого фаєрволу, що дозволяє SIP-трафік лише з IP-адреси Kamailio. Клієнти не можуть напряму звертатися до Asterisk, що є ключовою вимогою Zero Trust. Для цього вносимо наступні зміни у конфігураційний файл pjsip.conf, ізолювавши медіасервер і надавши можливість доступу до нього лише із Kamailio:

```
[transport-tls]
type=transport
protocol=tls
bind=127.0.0.1:5061
```

Рисунок 3.9 – Доступ до Asterisk лише від Kamailio (яке на цьому ж хості)

Далі прописуємо фільтрацію мережевого трафіку за допомогою засобів керування доступом на рівні ядра системи (iptables). Доступ до SIP-портів Asterisk дозволено лише з локального інтерфейсу, яким користується Kamailio. Таким чином, жоден клієнт не має змоги напряму звернутися до Asterisk, навіть перебуваючи в тій самій мережі. Усі запити маршрутизуються винятково через SIP-проксі-сервер.

```
iptables -A INPUT -i lo -j ACCEPT
iptables -A OUTPUT -o lo -j ACCEPT

iptables -A INPUT -p udp --dport 5060 -j ACCEPT
iptables -A INPUT -p tcp --dport 5061 -j ACCEPT

iptables -A INPUT -p udp --dport 10000:20000 -j ACCEPT

iptables -A OUTPUT -p udp --dport 5060 -j ACCEPT
iptables -A OUTPUT -p tcp --dport 5061 -j ACCEPT

iptables -A INPUT -p udp --dport 5062 -j DROP
iptables -A INPUT -p tcp --dport 5063 -j DROP

iptables -P INPUT DROP
iptables -P FORWARD DROP
iptables -P OUTPUT ACCEPT

apt-get install -y iptables-persistent > /dev/null 2>&1
netfilter-persistent save
```

Рисунок 3.10 – Внесені зміни у фільтрацію мережевого трафіку

Наступним кроком у реалізації моделі Zero Trust буде написання веб-застосунку, із якого легітимні оператори будуть здійснювати дзвінки по кліку, які будуть надходити у софтфон операторів лише через оригінації виклику до АМІ. Реалізується розмежування прав доступу користувачів, що виключає можливість викрадення чутливих даних, як номери телефону користувачів. При цьому, адміністратор системи матиме доступ до бази даних користувача. Додатково, рекомендується створення політики паролів та логінів, аби виключити можливість атак на веб-сервер. Також, на майбутнє допрацювання, рекомендовано інтегрувати у застосунок примусову MFA для усіх користувачів системи, використовуючи інтеграцію із Google автентифікатором, аби

забезпечити безпеку не лише VoIP-рішень, але й ресурсів, які мають до них доступ.

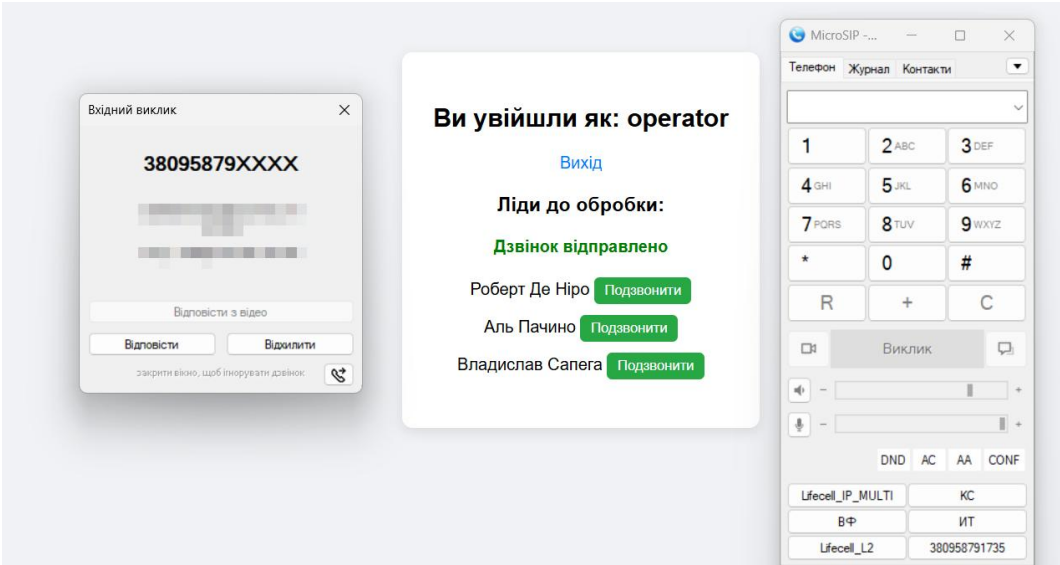


Рисунок 3.11 – Розроблений макет Web-застосунку, у якому реалізоване приховування номера клієнта.

Логування усіх подій, які відбуваються у системі є однією із основних парадигм моделі із нульовою довірою. Кожна дія, зроблена внутрішнім чи зовнішнім користувачем не повинна залишитись непоміченою та повинна бути записана у логи, та відповідно, проаналізована. Тому, спробуємо розгорнути системи моніторингу, які допоможуть аналізувати взаємодію із системою.

Для початку розгорнемо VoIPmonitor - систему пасивного моніторингу VoIP-мережевого трафіку, яка дозволяє здійснювати глибокий аналіз сигналізаційних протоколів та медіа-потоків. Дана система виконує роль мережевого сенсора, що у режимі реального часу фіксує всі спроби встановлення, передачі та завершення голосових викликів, а також аномальні дії, які можуть свідчити про спроби фроду, перехоплення або неправомірного використання інфраструктури.

Datetime	Duration	Caller num/name	SIP source IP	Proxy	Called num	SIP destination IP	Last response	Caller src RTP MOS	Called src RTP MOS	Commands	Note
2025-06-09 14:48:51	00:08	1001@192.168.0.105	192.168.0.105	185.168.129.136	0958791735@192.168.0.105	192.168.0.105 : 5060	200 OK	N/A	N/A	PCAP	
14:49:00	00:08	193.0.216.242 : 42716	193.0.216.242 : 42716	192.168.0.105	0958791735@192.168.0.105	192.168.0.105 : 5060	Q,850 reason: 17	N/A	N/A		

Рисунок 3.12 – VoIPmonitor

На знімку екрану відповідно зображено роботу VoIPmonitor у дії – є можливість переглянути дзвінки, детальніше розглянути сигналізацію під час

дзвінка, є можливість швидко та зручно завантажити дампи виклику (що є зручним, у випадку, коли потрібно передавати приклади провайдеру дзвінка).

Окрім того, у випадку, якщо кількість дзвінків на день є доволі об'ємною, в програмі реалізовано зручну можливість для пошуку необхідного дзвінка (фільтрація дуже гнучка, можна шукати як за номером, так і за датою дзвінка, окрім того, можна навіть шукати за кодом відповіді, що є зручним, наприклад, для передачі прикладів неуспішних дзвінків провайдеру номера).

Рисунок 3.13 – Фільтрація у VoIPmonitor

Наступним етапом є розгортання системи Elasticsearch - потужного інструменту для зберігання, пошуку та аналітики великих обсягів журналів подій у реальному часі. У контексті реалізованої моделі Zero Trust Elasticsearch виконує роль центрального сховища для логів, що надходять з різних компонентів VoIP-інфраструктури, забезпечуючи централізовану обробку та доступність цих даних для подальшого аналізу.

Для прикладу, після розгортання та налаштування системи спробуємо просканувати систему за допомогою SIPVicious [11]. Як результат – лог потрапляє у Elasticsearch. Окрім того, так само доступні інші логи системи – наприклад, успішної авторизації на Kamailio. Надалі, адміністратор системи може налаштовувати необхідні логи та додатково забезпечити їх передачу у систему Elasticsearch. Також, у майбутньому, є можливість інтегрувати систему із IDS/IPS для автоматичного виявлення та аналізу підозрілої активності системи.

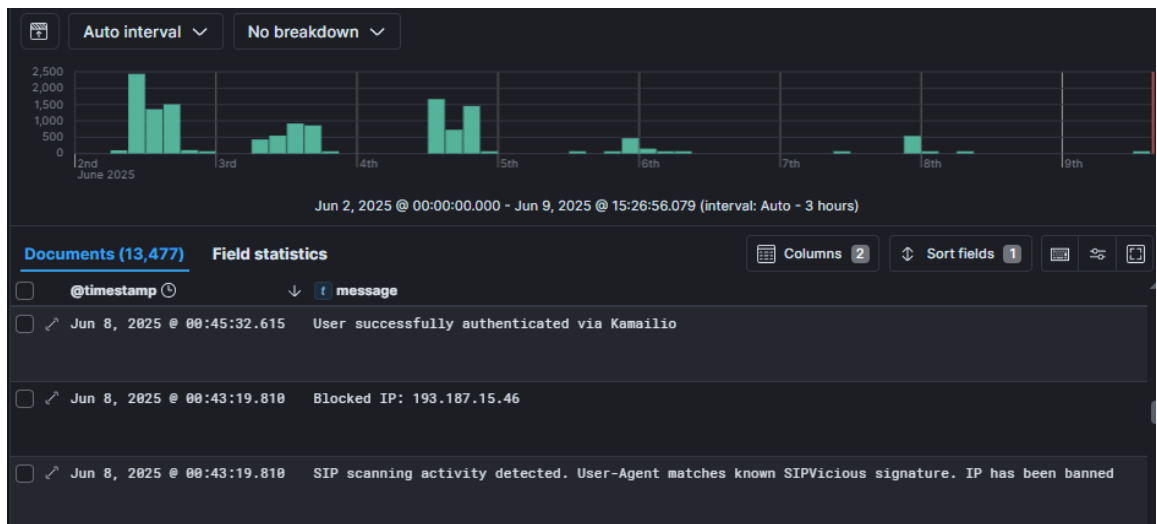


Рисунок 3.14 – Результат логуювання у Elasticsearch

3.2 Порівняльний аналіз системи із моделлю нульової довіри та незахищеної системи

За результатом впроваджених принципів моделі нульової довіри у систему маємо багато переваг у порівнянні із незахищеною системою. Розглянемо та підсумуємо усі плюси та мінуси впровадженого рішення перед незахищеною системою.

Для початку, спробуємо виконати додатково низку атак для порівняння на незахищений медіасервер. Виконаємо сканування медіасервера – реалізацію атаки user enumeration, на меті якої є інформаційна розвідка заведених у систему SIP ліній. Маючи пул робочих місць користувачів та потужний словник паролів, зловмисник може легко провести brute-force атаку та отримати доступ до здійснення дзвінків через номер клієнта що як мінімум може призвести до фінансових витрат шляхом дзвінків на дорогі напрямки, а як максимум – здійсненням фрод дзвінків від імені компанії та значних репутаційних ризиків.

Сканування здійснюється утилітою sipvicious, як результат шляхом сканування пулу робочих ліній було знайдено дійсні SIP-лінії:

```
(venv-sip)-(kali@kali)-[~/sipvicious/sipvicious]
$ python3 svwar.py -e 1000-1010 -m INVITE 192.168.0.105 --port 5060 --force

WARNING:TakeASip:using an INVITE scan on an endpoint (i.e. SIP phone) may cause it to ring
and wake up people in the middle of the night

+-----+-----+
| Extension | Authentication |
+-----+-----+
| 1001      | reqauth       |
+-----+-----+
| 1002      | reqauth       |
+-----+-----+
```

Рисунок 3.15 - Сканування незахищеного медіасерверу, атака user enumeration

При цьому, розпізнати сканування акаунту доволі легко, у пакеті INVITE замість звичайного софтфону у полі User-Agent бачимо надпис friendly-scanner.

```
Session Initiation Protocol (INVITE)
  Request-Line: INVITE sip:1001@192.168.0.105 SIP/2.0
  Message Header
    Via: SIP/2.0/UDP 127.0.1.1:5060;branch=z9hG4bK-1977991895;rport
    Max-Forwards: 70
    To: "1001"<sip:1001@192.168.0.105>
      SIP to display info: "1001"
      SIP to address: sip:1001@192.168.0.105
    From: "1001"<sip:1001@192.168.0.105>;tag=313030310133343030313135313638
      SIP from display info: "1001"
      SIP from address: sip:1001@192.168.0.105
      SIP from tag: 313030310133343030313135313638
    User-Agent: friendly-scanner
    Call-ID: 1635887510
  RFC 3261: User-Agent Header (sip.User-Agent), 30 bytes
```

Рисунок 3.16 - User-Agent у SIP пакеті INVITE при скануванні медіасерверу

Отримавши перелік SIP-акаунтів системи та маючи доступ до об'ємного списку паролів зловмисник може легко реалізувати brute-force атаку та підібравши пароль отримати доступ до робочої SIP-лінії:

```
(venv-sip)-(kali@kali)-[~/sipvicious/sipvicious]
$ python3 svcrack.py -u 1001 -d wordlist.txt udp://192.168.0.105:5060

+-----+-----+
| Extension | Password |
+-----+-----+
| 1001      | 123456   |
+-----+-----+
```

Рисунок 3.17 - Успішна атака brute force на незахищений медіасервер

При цьому, спробувавши виконати що сканування дійсних акаунтів, що атаку типу brute-force отримаємо помилку. Система захищена від подібних атак на кількох рівнях: починаючи від базової ізоляції медіасервера всередині мережі, де Kamailio стає у ролі справжнього захисного бар'єру. У конфігураційному файлі Kamailio також стоїть заборона на використання підозрілих User agent, що миттєво блокується та логується у ELK

```
(venv-sip)-(kali@kali)-[~/sipicious/sipicious]
└─$ python3 swar.py -e 1000-1010 -m INVITE 192.168.0.105 --port 5060 --force
WARNING:TakeASip:using an INVITE scan on an endpoint (i.e. SIP phone) may cause it to ring and wake up people in the middle of the night
WARNING:TakeASip:Bad user - SIP/2.0 401 - swar will probably not work!
WARNING:TakeASip:We got an unknown response
ERROR:TakeASip:Response: 'SIP/2.0 401 Unauthorized\r\nVia: SIP/2.0/UDP 127.0.1.1:5060;rport=5060;received=192.168.0.110;branch=z9hG4bK-3092695847\r\nCall-ID: 2937676646\r\nFrom: "1000" <sip:1000@192.168.0.105>;tag=3130303001333534343831323332\r\nTo: "1000" <sip:1000@192.168.0.105>;tag=z9hG4bK-3092695847\r\nSeq: 1 INVITE\r\nWWW-Authenticate: Digest realm="asterisk",nonce="1749460906/695b87ec9f518173e51c7050d4c98d08",opaque="1alc411417152ab4",algorithm=MD5,qop="auth"\r\nServer: Asterisk PBX 20.6.0-dfsg+cs6.1
3.464314-2b1u5\r\nContent-Length: 0\r\n\r\n'
WARNING:root:found nothing
```

Рисунок 3.18 – Спроба сканування захищеної системи завершилась помилкою
Отриманий лог виглядає наступним чином:

```
@timestamp
Jun 9, 2025 @ 20:43:49.695 Suspicious SIP activity detected: account scanning attempt using User-Agent 'friendly-scanner' from IP 193.34.167.21
```

Рисунок 3.19 – Лог про використання UA friendly-scanner

Аналогічно запобігається та логується інформація про інші інциденти: спроби 3х і більше невдалих автентифікацій, спроби дзвінків напряду без авторизації у застосунку, а інформацію про кожен виклик можливо перевірити та детально розглянути у VoIPmonitor.

Складемо таблицю, у якій буде наведено порівняння нашої системи із типовою системою, складовою якої є незахищений медіасервер Asterisk та софтфони користувачів:

Таблиця 3.1 – Порівняння системи із імплементованою моделлю нульової довіри із незахищеною системою

Параметр	Система з Zero Trust	Незахищена VoIP-система
Автентифікація	Обов'язкова через веб-застосунок з можливістю розширити захищеність через MFA	Робоча лінія одразу заводиться у софтфон
Авторизація	За принципом найменших привілеїв	Всі користувачі мають однаковий рівень доступу
Виявлення вторгнень	Інтеграція з VoIPmonitor та логами Kamailio	Відсутній аналіз трафіку або активності
Шифрування	TLS + SRTP на всіх рівнях	Трафік передається у відкритому вигляді, можливість перехопити вміст розмови

Кінець Таблиці 3.1

Журнали подій	Централізоване логування через ELK	Часткове або повністю відсутнє логування
Відстеження аномалій	Поведінкова аналітика через Elasticsearch	Відсутній моніторинг
SIP-проксі	Kamailio з антифлуд-захистом	Відкрите з'єднання до Asterisk
Ізоляція сегментів	DMZ + мікросегментація трафіку	Всі вузли у загальній мережі
Безпека внутрішніх загроз	Немає довіри навіть до внутрішніх користувачів	Внутрішній трафік не перевіряється
Захист від enumeration	Відповіді Kamailio не розкривають, які користувачі існують	Відповіді дозволяють брутфорс та enumeration
Шифрування паролів	bcrypt + TLS між Kamailio ↔ Flask	plain text у Asterisk
Аудит викликів	Повний лог викликів через VoIPmonitor	Часткове логування
Рольова модель	Ролі admin/operator/user	Всі SIP-аккаунти рівнозначні
Маскування номерів	Приховані останні 4 цифри номеру клієнта, що дозволя виключити витoki клієнтських номерів	Номер абонента передається відкрито
Структура доступу до Asterisk	Тільки з Kamailio (ACL)	Відкритий порт до зовнішньої мережі

Тобто, можемо виділити кілька основних плюсів реалізованої системи у порівнянні із звичайним незахищеним медіасервером:

Уся інформація у нашій системі є зашифрованою. Паролі від робочих ліній зберігаються не у конфігураційному файлі, а у окремій базі даних, у загешованому вигляді. При цьому, вмiсто кожного дзвінка так само є зашифрованим, медіапоток передається протоколом SRTP, що унеможливорює його перехоплення та подальше відтворення. Сигналізація так само зашифрована – злоумисник не зможе оперувати чутливими до атак/підмін даними протоколу SIP. При цьому ж, незахищений сервер можна легко атакувати та отримати усі зареєстровані робочі лінії, якщо пароль до лінії легкий до вгадування – його можна підібрати маючи словник скомпроментованих паролів. Зміст розмов при передачі через «голий» RTP може перехопити та відтворити будь-хто у мережі, що є особливо чутливим, якщо системою користується користувач із зовнішньої незахищеної мережі.

Усі користувачі системи не є довіреними. Перш ніж отримати можливість дзвонити – необхідно підтвердити свою особистість та тільки потім отримати дозвіл на це, врахувавши роль користувача та отримавши найменші дозволи та привілеї. Дзвонити можна лише із внутрішньої мережі, але навіть маючи до неї доступ користувач не є повністю довіреним. За замовчуванням, доступ до номерів абонентів прихований, а отримати його може лише довірений користувач. Самі дзвінки виконуються лише по кліку із створеного веб-сайту, де передбачено приховування номерів, а дзвінок оригiнується на робочу лінію користувача. При цьому, вихідні дзвінки заборонені, маючи зареєстровану лінію ані злоумисник, який скомпроментував акаунт, ані недобросовісний оператор не зможуть. У незахищеній системі, при цьому, будь-хто може виконувати вихідні дзвінки, що робить систему вразливою для зловживання дзвінками на дорогі напрямки та можливістю здійснення фрод дзвінків злоумисниками від імені компанії, що становить величезні фінансові та репутаційні втрати.

Медіасервер захищено від прямої взаємодії із ним, а доступ до нього відбувається лише через проху Kamailio, на якому, при цьому, реалізовано механізми блокування підозрілих активностей. Незахищений медіасервер, до

якого не підв'язано Kamailio, своєю чергою є предметом сканувань зловимників, вразливий до атак типів DOS, brute-force та enumeration.

Логування викликів та дій користувачів є однією із основних переваг системи: усі дії можна відслідкувати, а результати атак знайти та візуалізувати у ELK. Це надає багато переваг для аналітиків безпеки системи та VoIP-інженерів, дампи дзвінків можна легко отримати у VoIPmonitor, де, до того ж, можна зручно переглянути сигналізацію дзвінка та усю детальну інформацію по ньому. Це не просто набагато зручніше ніж Wireshark, це також забезпечує спеціалізовану функціональність, орієнтовану саме на потреби VoIP-аналітики: зокрема, підтримку інтерпретації SIP/SDP-повідомлень, декодування RTP-потоків, автоматичне визначення якості викликів за допомогою MOS-показників, Jitter та Packet Loss, а також побудову графіків дзвінків у реальному часі. При цьому, у базовій реалізації все, що можемо отримати від системи – це базові логи медіасервера та неструктуровану інформацію у Wireshark. Годі й казати про перспективи інтеграції системи із IDS/IPS.

Реалізована система є гарним фундаментом безпечної системи, яка до цього ж, може бути легко масштабована. Маючи робочий Kamailio можна легко додати до нього модуль dispatcher, який дозволить розширити систему на декілька Asterisk, що в кілька разів зможе покращити відмовостійкість системи та балансувати навантаження. Окрім того, маючи свій окремий веб-застосунок для здійснення дзвінків, можна розширити його базовий функціонал до справжньої гнучкої CRM-системи, або ж замінити його на безпечну CRM із реалізованим механізмом оригінації виклику та приховування номерів абонентів.

Висновки до розділу 3

У цьому розділі було реалізовано практичне впровадження VoIP-системи відповідно до моделі нульової довіри (Zero Trust), з урахуванням актуальних

вимог до інформаційної безпеки, типових векторів атак та недоліків традиційних VoIP-архітектур.

В результаті виконано поетапне розгортання базової VoIP-інфраструктури на основі Asterisk та Kamailio. Було налаштовано шифрування сигналізаційного трафіку (TLS) та медіапотоків (SRTP), медіасервер ізольовано від зовнішніх запитів шляхом підняття SIP-proxi Kamailio, що надає захист від атак типу brute-force, enumeration, DoS. Впроваджено централізоване логування за допомогою Elasticsearch та VoIPmonitor, а розробка веб-застосунку з розмежуванням доступу, приховуванням номерів абонентів та механізмом безпечної ініціації дзвінків забезпечує безпеку чутливих даних та додаткову перевірку ідентичності користувачів системи.

Окрім того було проведено порівняльний аналіз захищеної та незахищеної архітектури, який наочно демонструє переваги Zero Trust-моделі.

Аналіз показав, що впроваджена система значно перевищує незахищену за критеріями безпеки, масштабованості, керованості та прозорості. Особливу роль відіграє структуроване логування та можливість гнучкої аналітики, що надає інструменти як для реагування на інциденти, так і для довгострокового аудиту та оптимізації системи.

Отже, реалізована VoIP-архітектура з Kamailio як SIP-проксі, Asterisk як медіасервером, захищеними каналами зв'язку та централізованою аналітикою відповідає принципам моделі Zero Trust та формує надійну основу для подальшого масштабування, інтеграції з CRM-системами, впровадження MFA та захисту від зовнішніх і внутрішніх загроз.

ВИСНОВКИ

У дипломній роботі було розроблено та реалізовано комплексне рішення щодо підвищення рівня захищеності VoIP-систем на основі концепції моделі Zero Trust. Проведена робота охопила як теоретичне обґрунтування доцільності впровадження моделі нульової довіри у середовищі IP-телефонії, так і поетапну реалізацію безпечної архітектури з відповідними технічними і програмними засобами. Результати продемонстрували ефективність запропонованого підходу та його переваги над типовими, незахищеними VoIP-рішеннями.

На практичному рівні реалізовано сучасну VoIP-інфраструктуру, в основу якої покладено модульну архітектуру з чітким розмежуванням функцій між SIP-проксі сервером Kamailio та медіасервером Asterisk. Kamailio, забезпечує обробку сигналізаційного трафіку, перевірку автентичності клієнтів, захист від флуда, SQL-ін'єкцій, user enumeration, brute-force атак, та є єдиною точкою взаємодії клієнтів із системою. Усі запити, що надходять, проходять перевірку політик безпеки, а доступ до медіасервера Asterisk надається лише після успішної автентифікації та авторизації, виключно з дозволених внутрішніх інтерфейсів. Asterisk, ізольований у внутрішньому сегменті мережі, відповідає за маршрутизацію викликів та обробку RTP-трафіку.

Особливу увагу приділено впровадженню засобів шифрування: SIP-повідомлення передаються через TLS, а голосовий трафік - через SRTP. Це забезпечує конфіденційність сигналізації та медіаданих навіть при потенційному доступі злоумисника до мережевого трафіку. Також реалізовано відмову від зберігання облікових даних у відкритому вигляді - замість цього автентифікація здійснюється через захищену базу даних.

В межах розробленого веб-застосунку реалізовано механізм централізованої автентифікації та авторизації із дотриманням принципу найменших привілеїв. Користувачі системи отримують строго обмежені ролі, що

визначають їхні права доступу до функціоналу. Важливо, що вихідні дзвінки ініціюються виключно із веб-інтерфейсу через АМІ-інтеграцію, без розкриття номерів абонентів операторам, що значно знижує ризики витоку клієнтських даних. Додатково враховано можливість впровадження багатфакторної автентифікації (MFA) для підвищення рівня захисту користувачів системи.

Ключовим елементом забезпечення безпеки стала система централізованого логування. Усі події, пов'язані з автентифікацією, викликами, виявленням підозрілої активності логуються у реальному часі та надсилаються до Elasticsearch. У доповнення до цього впроваджено систему VoIPmonitor, яка забезпечує детальний контроль за SIP-сесіями.

У результаті проведеного порівняльного аналізу впроваджена архітектура демонструє суттєві переваги над незахищеними VoIP-системами. Система є універсальною платформою для подальшого розвитку та інтеграції з CRM, засобами автоматизації бізнес-процесів, службами виявлення загроз, а також готова до адаптації під специфічні потреби організацій різного масштабу.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАНЬ

1. VoIP Protocols Overview [Електронний ресурс]. – Режим доступу: <https://getvoip.com/blog/voip-protocols/>
2. VoIP Security Best Practices [Електронний ресурс]. – Yeastar Blog. – Режим доступу: <https://www.yeastar.com/blog/voip-security-best-practices/>
3. Rosenberg, J., Schulzrinne, H., Camarillo, G. та ін. SIP: Session Initiation Protocol : RFC 3261 [Електронний ресурс]. – June 2002. – Режим доступу: <https://datatracker.ietf.org/doc/html/rfc3261>
4. Kamailio SIP Server – Security Guidelines [Електронний ресурс]. – Kamailio Wiki. – Режим доступу: <https://www.kamailio.org/wiki/tutorials/security/kamailio-security>
5. Basic Vulnerability Issues for SIP Security [Електронний ресурс]. – Режим доступу: <https://startrinity.com/VoIP/Resources/basic-vulnerability-issues-for-sip-security.pdf>
6. A Look at VoIP Vulnerabilities [Електронний ресурс]. – SciSpace. – Режим доступу: <https://scispace.com/pdf/a-look-at-voip-vulnerabilities-6c391bcjbs.pdf>
7. Perimeter Security vs. Zero Trust – Cybersecurity Transformation [Електронний ресурс]. – Tufin Blog. – Режим доступу: <https://www.tufin.com/blog/perimeter-security-vs-zero-trust-cybersecurity-transformation>
8. VoIP and SIP Security Slides [Електронний ресурс]. – SlidePlayer. – Режим доступу: <https://slideplayer.com/slide/12328661/>
9. Zero Trust vs. Traditional Perimeter Security [Електронний ресурс]. – AZTech IT Blog. – Режим доступу: <https://www.aztechit.co.uk/blog/zero-trust-vs-traditional-perimeter-security>
10. Oladoja, T., & Alimi, A. (2024). End-to-End Security in Converged VoIP Systems: Addressing Risks and Implementing Best Practices [Електронний ресурс]. – Режим доступу: https://www.researchgate.net/publication/388614977_End-to-

End_Security_in_Converged_VoIP_Systems_Addressing_Risks_and_Implementing_Best_Practices

11. SIPVicious Blog [Электронный ресурс]. – Режим доступа:
<https://blog.sipvicious.org/>

ДОДАТОК А

Реалізований додаток із розмежуванням доступу

```

from flask import Flask, request, jsonify, make_response, render_template_string,
redirect, url_for, flash
from flask_httpauth import HTTPTokenAuth
import pymysql
import jwt
import datetime
from functools import wraps
import asyncio
from panoramisk.manager import Manager
import bcrypt
import logging
from logging.handlers import RotatingFileHandler

app = Flask(__name__)
app.config['SECRET_KEY'] = 'my_secret_key'

log_file_path = '/var/log/flask_login_attempts.log'
handler = RotatingFileHandler(log_file_path, maxBytes=10000, backupCount=1)
handler.setLevel(logging.INFO)
formatter = logging.Formatter('%(asctime)s %(levelname)s: %(message)s [in
%(pathname)s:%(lineno)d]')
handler.setFormatter(formatter)
if not app.logger.handlers:
    app.logger.addHandler(handler)
app.logger.setLevel(logging.INFO)

auth = HTTPTokenAuth(scheme='Bearer')

```

```

@app.before_request
def before_request():
    token = request.cookies.get('token')
    if token:
        request.headers = {**request.headers, 'Authorization': f'Bearer {token}'}

def create_connection():
    return pymysql.connect(
        host='localhost',
        user='dbuser',
        password=123456,
        database='mydb',
        cursorclass=pymysql.cursors.DictCursor
    )

def hash_password(password):
    return bcrypt.hashpw(password.encode('utf-8'), bcrypt.gensalt())

def check_password(hashed_password, plain_password):
    return bcrypt.checkpw(plain_password.encode('utf-8'),
        hashed_password.encode('utf8'))

def authenticate(username, password):
    conn = create_connection()
    with conn.cursor() as cursor:
        cursor.execute("SELECT * FROM users WHERE username = %s", (username,))
        user = cursor.fetchone()
    conn.close()
    if user and check_password(user['password'], password):
        return user

```

```
return None
```

```
def generate_token(user):
```

```
    token = jwt.encode({
        'user_id': user['id'],
        'role': user['role'],
        'internal_number': user['internal_number'],
        'username': user['username'],
        'exp': datetime.datetime.utcnow() + datetime.timedelta(hours=1)
    }, app.config['SECRET_KEY'], algorithm='HS256')
    return token
```

```
@auth.verify_token
```

```
def verify_token(token):
```

```
    try:
        return jwt.decode(token, app.config['SECRET_KEY'], algorithms=['HS256'])
    except jwt.ExpiredSignatureError:
        return None
    except jwt.InvalidTokenError:
        return None
```

```
def requires_role(role):
```

```
    def decorator(f):
```

```
        @wraps(f)
```

```
        def decorated_function(*args, **kwargs):
```

```
            token_data = auth.current_user()
```

```
            if token_data and token_data['role'] == role:
```

```
                return f(*args, **kwargs)
```

```
            return jsonify({'message': f'Access forbidden: requires {role} role'}), 403
```

```
        return decorated_function
```

```
return decorator
```

```
login_html = ""
<!doctype html>
<html>
<head><title>Увійти</title></head>
<body style="display: flex; justify-content: center; align-items: center; height: 100vh;
background: #f0f2f5;">
<div style="background: white; padding: 30px; border-radius: 10px; text-align:
center;">
<h2>Увійти</h2>
<form method="post">
  <input type="text" name="username" placeholder="Логін"><br><br>
  <input type="password" name="password" placeholder="Пароль"><br><br>
  <input type="submit" value="Увійти">
</form>
</div>
</body>
</html>
""
```

```
dashboard_html = ""
<!doctype html>
<html>
<head><title>Панель</title></head>
<body style="display: flex; justify-content: center; padding-top: 50px; background:
#f0f2f5;">
<div style="background: white; padding: 30px; border-radius: 10px; text-align:
center;">
<h2>Ви увійшли як: {{ username }}</h2>
```

```

<a href="/logout">Вихід</a>
<h3>Контакти:</h3>
{% with messages = get_flashed_messages() %}
    {% if messages %}<div style="color: green">{{ messages[0] }}</div>{% endif %}
{% endwith %}
<ul style="list-style: none; padding: 0;">
    {% for contact in contacts %}
        <li>
            {{ contact.name }}
            <form method="post" action="/exec_initiate-call" style="display:inline;">
                <input type="hidden" name="contactId" value="{{ contact.id }}">
                <button type="submit">Подзвонити</button>
            </form>
        </li>
    {% endfor %}
</ul>
</div>
</body>
</html>
'''

```

```

@app.route('/', methods=['GET', 'POST'])
def login():
    if request.method == 'POST':
        username = request.form['username']
        password = request.form['password']
        user = authenticate(username, password)
        if user:
            token = generate_token(user)

```

```

        response = make_response(redirect('/dashboard'))
        response.set_cookie('token', token)
        return response
    else:
        app.logger.info(f'Failed login attempt for user: {username}')
        return 'Невірний логін або пароль'
    return render_template_string(login_html)

@app.route('/dashboard')
@auth.login_required
def dashboard():
    token_data = auth.current_user()
    conn = create_connection()
    with conn.cursor() as cursor:
        cursor.execute("SELECT id, name FROM contacts")
        contacts = cursor.fetchall()
    conn.close()
    return render_template_string(dashboard_html, username=token_data['username'],
contacts=contacts)

@app.route('/exec_initiate-call', methods=['POST'])
@auth.login_required
@requires_role('user')
def exec_initiate_call():
    contact_id = request.form['contactId']
    token_data = auth.current_user()
    conn = create_connection()
    with conn.cursor() as cursor:

```

```

        cursor.execute("SELECT phone, name FROM contacts WHERE id = %s",
(contact_id,))
        contact = cursor.fetchone()
        conn.close()
        if not contact:
            flash('Контакт не знайдено')
            return redirect('/dashboard')

        phone_number = contact['phone']
        contact_name = contact['name']
        loop = asyncio.new_event_loop()
        asyncio.set_event_loop(loop)
        loop.run_until_complete(initiate_call(phone_number,
token_data['internal_number'], contact_name))
        flash("Дзвінок ініційовано")
        return redirect('/dashboard')

async def initiate_call(num_to_call, internal_number, contact_name):
    if num_to_call and internal_number:
        manager = Manager(loop=asyncio.get_event_loop(), host='192.168.0.105',
port=5061, username='deniro', secret='deniro')
        await manager.connect()
        action = {
            'Action': 'Originate',
            'Channel': f'PJSIP/{internal_number}',
            'Context': 'default',
            'Exten': num_to_call,
            'Priority': '1',
            'CallerID': contact_name,
            'Async': 'true'

```

```

    }

    try:
        response = await manager.send_action(action)
        logging.info(f'AMI Response: {response}')
    except Exception as e:
        logging.error(f'Exception occurred while sending action: {e}')
    await manager.close()

@app.route('/logout')
def logout():
    response = redirect('/')
    response.delete_cookie('token')
    return response

if __name__ == '__main__':
    app.run(debug=True)

```

ДОДАТОК Б

Конфігураційний файл kamailio.cfg

```

#!KAMAILIO

# --- Активация потрібних можливостей ---
#!define WITH_MYSQL    # Підключення до бази MySQL
#!define WITH_AUTH     # Підтримка автентифікації
#!define WITH_TLS      # TLS шифрування для SIP
#!define WITH_ANTIFLOOD # Захист від флуда
#!define WITH_NAT      # Обхід NAT
#!define WITH_TOPH     # Маскування заголовків

# --- Модулі Kamailio ---
loadmodule "db_mysql.so"    # Підключення до MySQL
loadmodule "sl.so" |       # Stateless SIP replies
loadmodule "tm.so"         # Транзакції SIP
loadmodule "rr.so"         # Record-Route обробка
loadmodule "pv.so"         # Псевдозмінні Kamailio
loadmodule "maxfwd.so"     # Захист від SIP-петель
loadmodule "usrloc.so"     # Локаційна база користувачів
loadmodule "registrar.so"  # Обробка REGISTER
loadmodule "textops.so"    # Робота з текстом
loadmodule "textopsx.so"   # Додаткові текстові функції
loadmodule "xlog.so"       # Логування
loadmodule "kex.so"        # Kernel extensions
loadmodule "siputils.so"   # SIP утиліти
loadmodule "sanity.so"     # Перевірка коректності SIP-повідомлень
loadmodule "htable.so"     # Хеш-таблиці в пам'яті
loadmodule "pike.so"       # Антифлуд-захист

```

```

loadmodule "topoh.so"      # Маскування заголовків
loadmodule "auth.so"      # Базова автентифікація
loadmodule "auth_db.so"   # Автентифікація з БД
loadmodule "domain.so"    # Підтримка SIP-доменів
loadmodule "tls.so"       # TLS
loadmodule "secfilter.so"  # Розширені безпекові фільтри
loadmodule "geoip2.so"     # Геолокаційний фільтр

# --- Налаштування підключення до БД ---
modparam("auth_db", "db_url", "mysql://kamailio:kamailiorw@localhost/kamailio")
modparam("usrloc", "db_mode", 3)
modparam("registrar", "method_filtering", 1)
modparam("rr", "enable_full_lr", 1)
modparam("tm", "fr_timer", 3000)
modparam("tm", "fr_inv_timer", 12000)
modparam("tm", "restart_fr_on_each_reply", 0)

# --- TLS шифрування ---
modparam("tls", "config", "/etc/kamailio/tls.cfg")

# --- Маскування SIP заголовків ---
modparam("topoh", "mask_key", "ChangeThisSecretKey")
modparam("topoh", "mask_ip", "10.0.0.1")
modparam("topoh", "mask_callid", 1)

# --- Захист від флуда (PIKE) ---
modparam("pike", "sampling_time_unit", 2)
modparam("pike", "reqs_density_per_unit", 20)
modparam("pike", "remove_latency", 4)

```

```

# --- Хеш-таблиці для бану IP та користувачів ---
modparam("htable", "htable", "ipban=>size=8;autoexpire=300;")
modparam("htable", "htable", "authban=>size=8;autoexpire=3600;")

# --- Налаштування автентифікації ---
modparam("auth", "nonce_expire", 60)
modparam("auth", "nonce_auth_max_drift", 2)
modparam("auth", "qop", "auth")
modparam("auth", "nonce_count", 1)
modparam("auth", "auth_checks_register", 11)
modparam("auth", "auth_checks_no_dlg", 9)
modparam("auth", "auth_checks_in_dlg", 15)
modparam("auth_db", "calculate_ha1", yes)
modparam("auth_db", "password_column", "password")
modparam("auth_db", "load_credentials", "")

# --- Secfilter: фільтрація заголовків, IP, SQL-ін'єкцій ---
modparam("secfilter", "db_url", "mysql://kamailio:kamailiorw@localhost/kamailio")
modparam("secfilter", "dst_exact_match", 0)

# --- Загальні налаштування сервера ---
server_signature=no
sip_warning=0
server_header="Server: MS Lync"
user_agent_header="User-Agent: My SIP Server"

# --- Інтерфейси прослуховування (UDP+TLS) ---
listen=udp:0.0.0.0:5060
listen=tls:0.0.0.0:5061 advertise PUBLIC_IP:5061

```

```

# --- Основна логіка маршрутизації SIP-повідомлень ---
request_route {
    # --- Захист від циклічних SIP запитів ---
    if (!mf_process_maxfwd_header("10")) {
        sl_send_reply("483","Too Many Hops");
        exit;
    }

    # --- Відповідь на Keepalive ---
    if (is_method("OPTIONS") && uri==myself) {
        sl_send_reply("200","Keepalive");
        exit;
    }

    # --- Перевірка на IP-бан ---
    if ($sht(ipban=>$si)!=null) {
        xlog("L_WARN", "Request from banned IP $si\n");
        exit;
    }

    # --- Антифлуд-захист ---
    if (!pike_check_req()) {
        $sht(ipban=>$si) = 1;
        xlog("L_ALERT","PIKE: blocking $rm from $si:$sp\n");
        exit;
    }

    # --- Детектори сканерів SIP ---
    if ($ua =~ "friendly-scanner|sipvicious|sipcli|pplsip|iWar|VaxSIP") {
        xlog("L_WARN", "Scanner detected $ua from $si, blocking\n");
    }
}

```

```

    exit;
}

# --- Захист від SQL-ін'єкцій ---
if (is_method("REGISTER|INVITE") && $au =~ "(=)|(--)|(|#|%27|%24)" && $au
!= $null) {
    xlog("L_WARN", "SQLi attack detected from $si\n");
    exit;
}

# --- Блокування користувачів з повторними спробами ---
if ($sht(authban=>$au)!= $null) {
    xlog("L_WARN", "Blocked auth user $au from $si due to repeated failures\n");
    exit;
}

# --- Перевірки безпеки ---
route(SEcurity_CHECKS);

# --- Обробка реєстрації ---
if (is_method("REGISTER")) {
    route(AUTH);
    save("location");
    exit;
}

# --- Пошук користувача для дзвінка ---
if (is_method("INVITE")) {
    if (!lookup("location")) {
        sl_send_reply("403", "Forbidden");
    }
}

```

```

        exit;
    }
}

# --- Релеювання решти запитів до Asterisk по TLS ---
$du = "sip:192.168.0.106:5061;transport=tls";
if (!t_relay()) {
    sl_reply_error();
}
}

# --- Обробка автентифікації з баном після 3-х спроб ---
route[AUTH] {
    if (!www_authorize("kamilio.org", "subscriber")) {
        $var(attempts) = $sht(authfail=>$au);
        if ($var(attempts) == $null) $var(attempts) = 0;
        $var(attempts) = $var(attempts) + 1;
        $sht(authfail=>$au) = $var(attempts);

        if ($var(attempts) >= 3) {
            # Банімо користувача, юзер-агент та IP на годину
            $sht(authban=>$au) = 1;
            $sht(ipban=>$si) = 1;
            xlog("L_ALERT", "User $au permanently banned from $si after 3 failed
attempts\n");
        }

        www_challenge("kamilio.org", "0");
        exit;
    }
}

```

```

# Скидуємо лічильник при успішній автентифікації
$sht(authfail=>$au) = 0;
consume_credentials();
if (!lookup("location")) {
    sl_send_reply("401", "Unauthorized");
    exit;
}
}

# --- Розширена перевірка безпеки (заголовки, IP, гео) ---
route[SECURITY_CHECKS] {
    if (geoip2_match("$si", "src")) {
        secf_check_country($gip2(src=>cc));
        if ($? == -2) {
            xalert("$rm from $si blocked because Country '$gip2(src=>cc)' is blacklisted");
            exit;
        }
    }
}

secf_check_ip();
if ($? == -2) {
    xalert("$rm from $si blocked because IP address is blacklisted");
    exit;
}

secf_check_sqli_all();

secf_check_ua();
if ($? == -2) {
    xalert("$rm from $si blocked because UserAgent '$sua' is blacklisted");
}

```

```

    exit;
}

secf_check_from_hdr();
switch ($?) {
    case -2:
        xalert("$rm to $si blocked because From user '$fU' is blacklisted");
        exit;
    case -3:
        xalert("$rm to $si blocked because From domain '$fd' is blacklisted");
        exit;
    case -4:
        xalert("$rm to $si blocked because From name '$fn' is blacklisted");
        exit;
}

secf_check_to_hdr();
switch ($?) {
    case -2:
        xalert("$rm to $si blocked because To user '$tU' is blacklisted");
        exit;
    case -3:
        xalert("$rm to $si blocked because To domain '$td' is blacklisted");
        exit;
    case -4:
        xalert("$rm to $si blocked because To name '$tn' is blacklisted");
        exit;
}

secf_check_contact_hdr();

```

```

switch ($?) {
    case -2:
        xalert("$rm to $si blocked because Contact user '$ct' is blacklisted");
        exit;
    case -3:
        xalert("$rm to $si blocked because Contact domain '$ct' is blacklisted");
        exit;
}

if (is_method("INVITE")) {
    secf_check_dst($rU);
    if ($? == -2) {
        xalert("$rm from $si blocked because destination $rU is blacklisted");
        exit;
    }
}
}

```