МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ АВІАЦІЙНИЙ УНІВЕРСИТЕТ
ФАКУЛЬТЕТ АЕРОНАВІЦІЇ, ЕЛЕКТРОНІКИ ТА ТЕЛЕКОМУНІКАЦІЙ
КАФЕДРА ТЕЛЕКОМУНІКАЦІЙНИХ ТА РАДІОЕЛЕКТРОННИХ СИСТЕМ

ДОПУСТИТИ ДО ЗАХИСТУ
Завідувач кафедри
_____ Роман ОДАРЧЕНКО

“_____”_____ 2023 р.

КВАЛІФІКАЦІЙНА РОБОТА
(ПОЯСНЮВАЛЬНА ЗАПИСКА)

ВИПУСКНИКА ОСВІТНЬОГО СТУПЕНЯ БАКАЛАВР

Тема: “Розробка системи PBX на базі платформи ASTERISK”

Виконавець: _____ Анжеліка ЗДРИЛЮК
(підпис)

Керівник: _____ Маирна МАЛОЄД
(підпис)

Нормоконтролер: _____ Денис БАХТІЯРОВ
(підпис)

Київ 2023

НАЦІОНАЛЬНИЙ АВІАЦІЙНИЙ УНІВЕРСИТЕТ

Факультет аеронавігації, електроніки та телекомунікацій
Кафедра телекомунікаційних та радіоелектронних систем
Спеціальність 172 «Телекомунікації та радіотехніка»
Освітньо-професійна програма «Телекомунікаційні системи та мережі»

ЗАТВЕРДЖУЮ
Завідувач кафедри
_____ Одарченко Р. С.
«_ __ _» _____2023 р.

ЗАВДАННЯ

на виконання дипломної роботи

Здрилюк Анжеліки Олександрівни

1. Тема дипломної роботи: «Використання платформи Asterisk для застосування в системі PBX» затверджена наказом ректора від від 29 березня 2023 р. № 421/ст.

2. Термін виконання роботи: з 22.05.2023 по 25.06.2023

3. Вихідні дані до роботи: теоретичні та методичні дані для дослідження системи PBX та реалізації її функцій.

4. Зміст пояснювальної записки (перелік питань, що їх належить розробити): Розгляд поняття IP-телефонії та PBX. Дослідження найбільш використовуваних IP PBX. Проблема, яку вирішує ASTERISK. Структура ASTERISK та його функції. Процес встановлення ASTERISK. Розробка системи PBX та її дослідження.

5. Перелік обов'язкового графічного матеріалу (з точним визначенням обов'язкових рисунків, діаграм, таблиць тощо): рисунки, що показують структуру IP-телефонії, структуру PBX, архітектуру протоколу SIP, принцип роботи RTP-протокола; представлення функцій «ASTERISK в якості шлюзу офісної АТС», «Знайди мене, йди за мною», «використання ASTERISK для забезпечення підтримки VoIP в застарілій офісній АТС»; рисунок, на якому показано потік управління аутентифікацією по відношеннюдо ASTERISK; рисунки налаштування ASTERISK.

6. Календарний план-графік

№	Завдання	Термін виконання	Відмітка про виконання
1	Розробити деталізований зміст розділів кваліфікаційної роботи	22.05.2023-24.05.2023	Виконано
2	Вступ	25.05.2023	Виконано
3	Розгляд поняття ip-телефонії та pbx, дослідження найбільш використовуваних ip pbx	26.05.2023 – 28.05.2023	Виконано
4	Проблема, яку вирішує asterisk. Структура asterisk та його функції. Процес встановлення asterisk	29.05.2023 – 05.06.2023	Виконано
5	Розробка системи pbx та її дослідження	05.06.2023 – 10.06.2023	Виконано
6	Усунення недоліків дипломної роботи, оформлення та захист кваліфікаційної роботи	12.06.2023 – 25.06.2023	Виконано

7. Дата видачі завдання: «19» травня 2023р.

Керівник дипломної роботи: _____ Марина МАЛОЄД

Завдання прийняла до виконання: _____ Анжеліка ЗДРИЛЮК

РЕФЕРАТ

Пояснювальна записка до дипломної роботи «Розробка системи PBX на базі платформи ASTERISK»: 95 сторінок, 49 рисунків, 1 таблиця, 12 джерел використаної літератури.

ASTERISK, IP-PBX, IP-ТЕЛЕФОНІЯ, ТРАНК, З'ЄДНАННЯ, ЗВ'ЯЗОК.

Об'єкт дослідження – система PBX на базі платформи ASTERISK.

Предмет дослідження – дослідження та налаштування системи PBX на базі платформи ASTERISK

Мета кваліфікаційної роботи – дослідження системи PBX. Розробка системи PBX на базі платформи ASTERISK для її використання в офісах. Розробка лабораторних робіт для студентів на платформі ASTERISK.

Результати роботи: можуть бути використані для подальшого використання системи в офісах. Також розроблені лабараторні роботи можуть бути використані в подальшому студентами для здобуття навичок роботи з системою PBX на базі платформи ASTERISK.

ЗМІСТ

СПИСОК СКОРОЧЕНЬ

АТС — Автоматична телефонна станція

ПК — Персональний комп’ютер

VoIP — Voice-over-IP

IP — Internet Protocol

LAN — Local Area Network

PBX — Private Branch Exhange

TCP — Transmission Control Protocol

SIP — Session Initiation Protocol

IETF — Internet Engineering Task Force

UAC — User Agent Client

UAS — User Agent Server

SDP — Session Description Protocol

RTP — Real-time Transport Protocol

UDP — User Datagram Protocol

TLS — Transport Layer Security

IT — Information Technologies

PCI — Peripheral component interconnect

GUI — Graphical user interface

RAM — Random Access Memory

HTTP — HyperText Transfer Protocol

IVR — Interactive Voice Response

DID — Direct Inward Dialing

NAT — Network Address Translation

ВСТУП

На сьогоднішній день дуже швидкими темпами розвивається наука та техніка. Вченими створюються нові технології та винаходи, доказом цього є поява системи IP-PBX(Private Branch eXchange). При використанні IP-PBX фізичною мережею для передачі голосу є (вже існуюча) локальна мережа та інтернет. Тому впровадження IP-PBX простіше і дешевше, а також дозволяє об'єднати кілька географічно розподілених офісів в єдину систему телефонії, забезпечити безкоштовним зв'язком надомних і мобільних працівників.

Історія інтернет телефонії й технолгії VoIP почалася у 1925 році, коли компанія AT&T, яка тоді була, по суті, єдиним гравцем ринку телефонних комунікацій, відкрила підрозділ Bell Laboratories, в чиї завдання входило винахід і подальша розробка технологій зв'язку, які дозволили б компанії поліпшити свій сервіс. У 1928 році Гомер Дадлі створив перший електронний синтезатор голосу, який отримав назву вокодер. Вокодер аналізував звуки, що видаються ротом і зв'язками людини, і вмів відторювати їх, що виливалося в аналог мови. Принцип роботи вокодера можна порівняти з нинішніми технологіями передачі пакетів, в ході роботи яких на одному телефоні записуються зразки звуку, відтворювані потім на іншому пристрої. Перше серверне PBX-рішення було розроблено в 1996 році каліфорнійською компанією Virtual PBX, в якому реалізовані функції пошуку і проходження (find me / follow m?-e), а також контролю за допомогою веб-порталу. Перший серверний PBX не використав VoIP, а за допомогою мідного дроту з'єднувався з PSTN-мережею, даючи можливість здійснення дзвінків всередині однієї компанії.

Загалом, IP-PBX — це система корпоративної телефонії, основним каналом передачі голосу в якій є VoIP. [7] IP-PBX об'єднує всі офісні телефони в одну інтелектуальну мережу, в якій реалізуються різні голосові сервіси.

Ця тема є актуальною, тому що в майбутньому PBX замінить всі звичайні телефони в офісах.

РОЗДІЛ 1.
РОЗГЛЯД ПОНЯТТЯ IP-ТЕЛЕФОНІЇ ТА PBX, ДОСЛІДЖЕННЯ НАЙБІЛЬШ ВИКОРИСТОВУВАНИХ IP PBX

1.1. Що таке IP-телефонія?

VoIP (Англ. Voice-over-IP — IP-телефонія) – це система зв'язку, яка забезпечує передачу мовного сигналу по мережі Інтернет чи через будь-яккі інші IP-мережі [8]. Сигнал по каналу зв'язку передається в цифровому вигляді і, як правило, перед передачею перетворюється (стискується) з тим, щоб видалити надмірність, властиву людській мові.

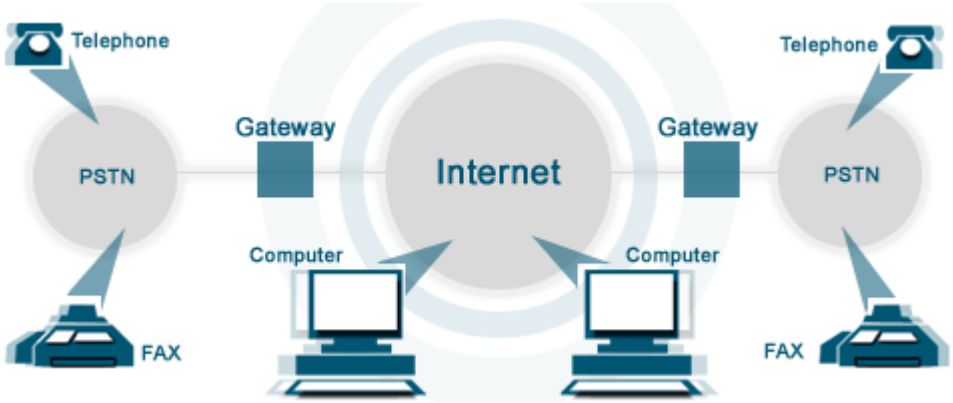

Рис. 1.1. Структура VoIP

При розмові, голосові сигнали (слова, які вимовляються) перетворяться в стислі пакети даних. Після, ці пакети даних надсилаються через мережу Internet іншій стороні. Коли пакети даних досягають адресата, вони декодуються в голосові сигнали оригіналу. [9]

IP-телефонія забезпечує голосовий зв'язок поверх мереж, що використовують Інтернет-протокол (IP). Технологія дозволяє об'єднати безліч розосереджених об'єктів організації, включаючи мобільних працівників, в єдину конвергентну мережу. [8]

IP-телефонія дозволяє досягти економії витрат шляхом об'єднання функцій голосового зв'язку і передачі даних в одну мережу [10], технічна підтримка якої може здійснюватися централізовано, а також шляхом ліквідації витрат на міжміський і міжнародний зв'язок при дзвінках на видалені об'єкти. [10]

Принцип дії IP-телефонії – є перетворення голосового зв'язку в пакети даних. Телефонні апарати повинні бути підключені до портів передачі даних IP-мережі. При цьому, телефонні функції можуть з легкістю виконуватися іншим пристроєм, вже підключеним до мережі. Таку функцію може виконувати ПК.

Телефонія типу «клієнт-сервер» [11], LAN з функціями телефонії, чиста IP-телефонія, конвергентна телефонія і LAN-телефонія – все це терміни, що позначають одну і ту ж базову розподілену архітектуру IP-телефонії.

В протилежність відомому підходу [9], коли окремий багатоканальний телефон розміщується у ПК на кожному робочому місці, IP-телефония використовує програмне забезпечення, яке встановлюється на ПК і виконує функції «програмованого» телефону. На відміну від телефонного устаткування, програмне забезпечення можна легко модернізувати і удосконалити. При цьому не виникає необхідності переривати робочий процес, витрачати кошти на обладнання і навіть підходити до кожного робочого місця.

Концепція передачі голосу поверх мережі даних вельми багатообіцяюча для компаній, які підтримують інтранет-зв'язок з усіма відділеннями і одночасно платять за використання контурів голосового зв'язку АТС в таких відділеннях. Економія витрат сама по собі є привабливим чинником, а параметри безпеки, надійності та якості зв'язку безсумнівно подвигнут менеджерів мереж зробити вибір на користь IP-телефонії.

Типи IP-телефонії.

Способи ведення розмов можна розділити на кілька типів:

Комп'ютер-Телефон — При цьому способі абонентам потрібно мати з одного боку комп'ютер зі звуковою картою і підключеними до неї динаміками [8] або навушниками і мікрофон (також може бути використаний IP-телефон) і з іншого боку самий звичайний телефон, підключений до телефонної лініі.

Комп'ютер-Комп'ютер — При цьому способі абонентам потрібно мати з обох сторін комп'ютер зі звуковою картою і підключеними до неї динаміками або навушниками і мікрофон (також може бути іспольхован IP-телефон). Це найпростіший і дешевий спосіб інтернет телефонії, тому витрати обох співрозмовників на зв’язок зводяться до вартості їх інтернет з'єднання.

Телефон-Телефон — При даному виборі одному з абонентів буде вимушений скористатися однією з численних компаній [10], які надають послуги IP-телефонії, а з іншого боку самий звичайний телефон, що підключений до телефонної лінії.

1.2. PBX

PBX (Private Branch Exhange) [8] — англійський термін, що означає офісну телефонну станцію, яка забезпечує встановлення, підтримання та розрив з'єднань між апаратами, тобто комутацію. PBX дозволяє розділяти обмежені ресурси [10] (міські лінії і номера) між необмеженим числом внутрішніх ліній (користувачів), за допомогою таких телефонних функцій, як внутрішній номерний план, переведення дзвінків, і інших. Саме тому PBX система необхідна будь-якій організації — вона дозволяє ефективно організувати телефонний зв'язок на підприємстві.

Традиційні PBX системи комутують канали (лінії зв'язку), перемикаючи ланцюги електричного струму. Нові PBX системи комутують пакети в мережі TCP/IP, і називаються IP PBX [8]. IP PBX працює на основі протоколів IP телефонії [10]. Також IP PBX можуть підтримувати і традиційні лінії зв'язку — такі IP PBX називаються гібридними. У перехідний період міграції від традиційної телефонії в IP середу саме гібридні IP PBX найбільш затребувані, хоча функцію конвертації традиційних телефонних каналів в IP пакети можна також винести в окремий пристрій — VoIP адаптер або VoIP шлюз, який далі підключається по протоколу IP телефонії до IP PBX.

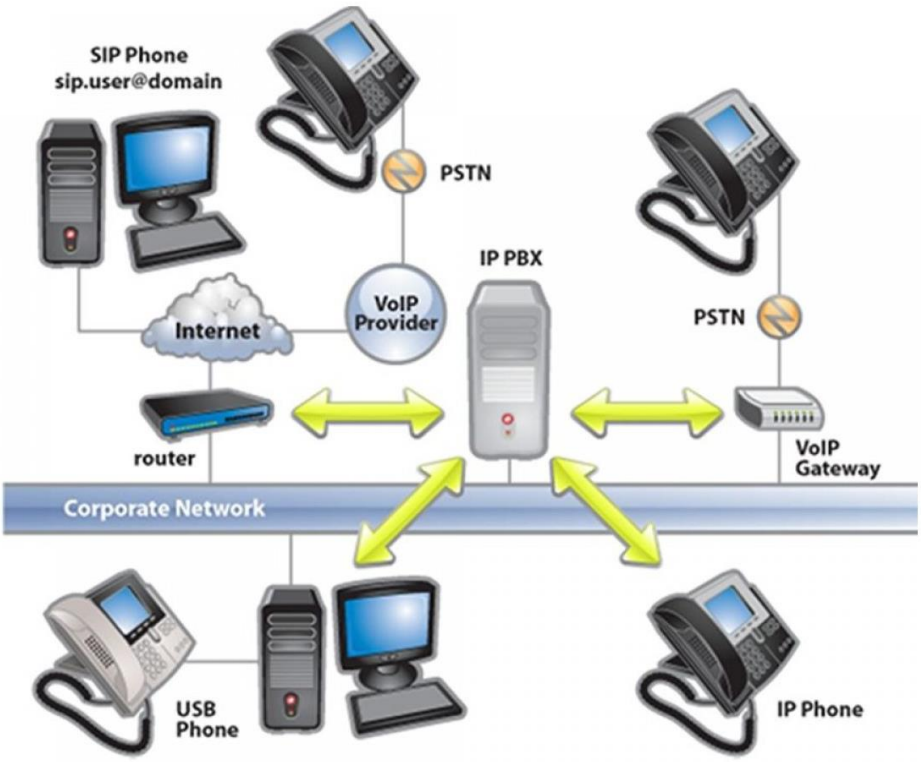

Рис. 1.2. Структура PBX

Нині тільки два протокола IP телефонії набули широкого поширення — H.323 і SIP.

Протокол, а правильніше, стік протоколів H.323, був розроблений міжнародним союзом електрозв'язку (англ. International Telecommunication Union, ITU) — міжнародна організація [8], що визначає рекомендації в галузі телекомунікацій та радіо. Метою створення протоколу була необхідність проведення аудіо і відеоконференцій по сучасних телекомунікаційних мереж, в тому числі цифрових і IP мережі.

1.3. Протокол SIP

SIP (англ. Session Initiation Protocol — протокол встановлення сеансу) — стандарт для способу встановлення і завершення користувацького інтернет-сеансу, що включає обмін мультимедійним змістом (відео- і аудиоконференции, миттєві повідомлення, онлайн-ігри, і ін.). Розробкою протоколу займалася Спеціальна Комісія Інтернет-розробок [9] (Internet Engineering Task Force, IETF) — відкрите міжнародне

співтовариство проектувальників, вчених, мережевих операторів і провайдерів [8], яке займається розвитком протоколів і архітектури Інтернету [9].

Протокол H.323 володіє великим стандартним набором можливостей по роботі з відео конференціями (його створювали телефоністи, а інтернет — одна з його робочих середовищ), а протокол SIP більше пристосований до роботи в мережах TCP / IP, і більш універсальний (його створювали «інтернетчики », і голос і відео — всього лише одні з типів медіа контенту).

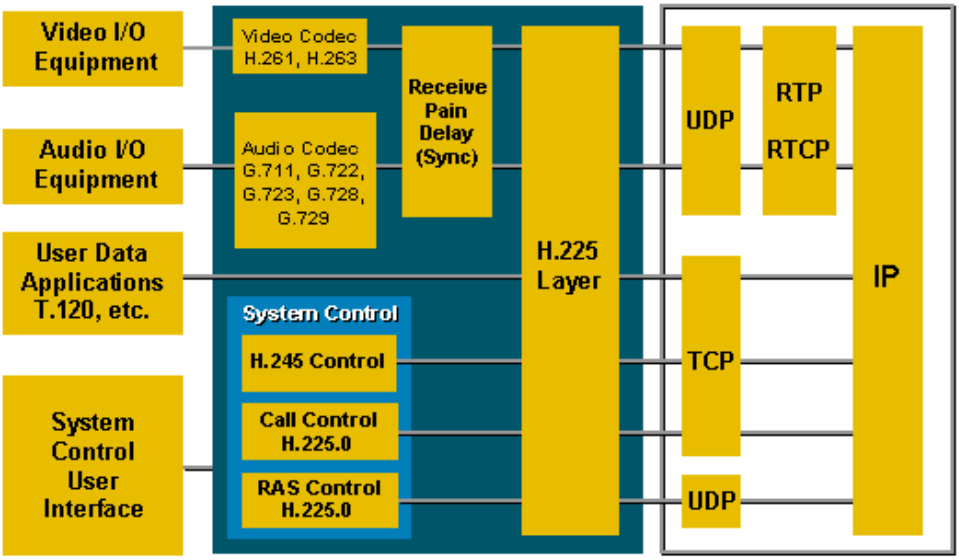

Рис. 1.3. Повний стек протоколів H.323

Інтернет переміг, і в теперішній час стандартом де-факто для IP телефонії вважається SIP, а H.323 протокол використовується в основному в системах багатокористувацьких відео конференцій і для обміну голосовим трафіком по IP між операторами зв'язку, хоча і в цих областях спостерігається тенденція переходу на SIP.

Таким чином, можна з упевненістю зробити висновок, що сучасні IP PBX системи працюють на базі протоколу IP телефонії SIP.

Розглянемо архітектуру SIP докладніше.

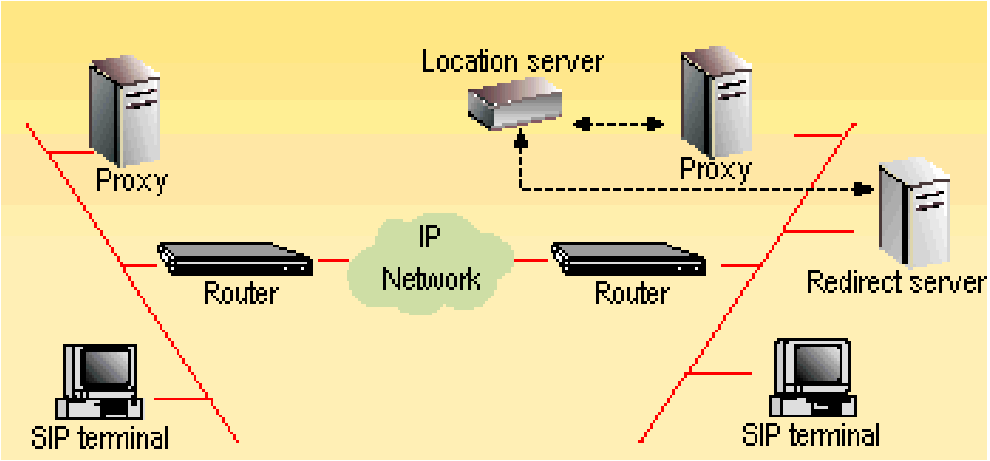

Рис. 1.4. Архітектура SIP

Специфікація протоколу SIP визначає клієнт-серверну архітектуру. Клієнт видає запити, із зазначенням того, що він хоче отримати від сервера. Сервер приймає і обробляє запити, видає відповіді, які містять повідомлення про успішність виконання запиту, повідомлення про помилку або інформацію, запитану клієнтом. Обслуговування виклику розподілено між різними елементами мережі SIP.

Основним функціональним елементом, що реалізує функції управління з'єднанням, є абонентський термінал. Інші елементи мережі можуть відповідати за маршрутизацію викликів, а також надають додаткові сервіси. Перелічимо основні елементи:

- Термінал. Коли клієнт і сервер реалізовані в кінцевому обладнанні і взаємодіють безпосередньо з користувачем, вони називаються користувацьким агентськими клієнтом — User Agent Client (UAC), користувацьким агентськими сервером — User Agent Server (UAS). Якщо в пристрої присутні і UAC, і UAS, то воно називається користувацьким агентом — User Agent (UA), а за своєю суттю є термінальне обладнання SIP. Приклади UA — апаратний або програмний SIP телефон, SIP адаптер. [8]
- Проксі-сервер (від англ. Proxy — «представник») представляє інтереси користувача в мережі. Він приймає запити, обробляє їх і виконує відповідні дії. Проксі-сервер також складається з клієнтської і серверної частин, тому може прий-

мати виклики, ініціювати запити і повертати відповіді. Передбачено два типи проксі-серверів:

- зі збереженням станів (stateful). Такий сервер зберігає в своїй пам'яті всі отримані запити та пов'язані з ним нові сформовані запити до закінчення транзакції.

- без збереження станів (stateless). Такий сервер просто обробляє отримані запити та на його базі реалізувати складні, інтелектуальні послуги неможливо. [8]

• Сервер переадресації використовується для визначення поточного місця розташування користувача [8]. Сервер переадресації НЕ термінує виклики і не ініціює власні запити, а тільки повідомляє адресу необхідного терміналу або проксі-сервера. Для цих цілей він взаємодіє з сервером визначення місцеположення. Для здійснення з'єднання користувач може не використовувати сервер переадресації, якщо він сам знає поточний адресу необхідного користувача. Користувач може переміщатися в межах мережі SIP, тому існує механізм визначення його місця розташування в поточний момент часу. Сервер визначення місцеположення користувачів служить для зберігання поточної адреси користувача і представляє собою базу даних адресної інформації.

Таким чином, специфікація протоколу SIP не визначає нічого, крім механізму встановлення і розриву сесії між клієнтом і сервером, а також пошуку елементів мережі. Тому SIP протокол використовується одночасно з іншими протоколами, які реалізують власні сервіси.

1.3.1. Допоміжні протоколи

Одним з таких допоміжних протоколів є SDP — Session Description Protocol, призначений для опису сесії передачі потокових даних, включаючи телефонію, інтернет-радіо, програми мультимедіа, і потокові додатки. SDP протокол описує формат заголовків і полів, в яких SIP клієнти і сервери перераховують свої сесійні можливості (наприклад, підтримувані алгоритми стиснення — кодеки).

Другим необхідним протоколом є RTP (англ. Real-time Transport Protocol),

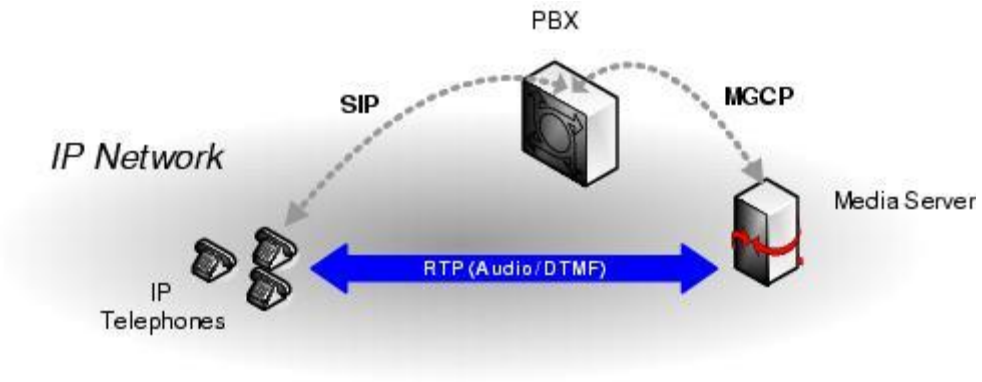

Рис. 1.5. Протокол RTP

який використовується для безпосередньої передачі трафіку реального часу. Протокол RTP в своєму заголовку переносить дані, необхідні для відновлення голосу або відеозображення в приймальному вузлі, а також дані про тип кодування інформації (JPEG, MPEG і т. П.). У заголовку даного протоколу, зокрема, передаються тимчасова мітка і номер пакета. Ці параметри дозволяють при мінімальних затримках визначити порядок і момент декодування кожного пакету, а також інтерполювати втрачені пакети. В якості нижчого протоколу транспортного рівня, як правило, використовується протокол UDP. Встановлення та розрив з'єднання не входить в список можливостей RTP, такі дії виконуються сигнальним протоколом SIP.

Таким чином, робота SIP PBX базується на трьох основних протоколах: SIP, SDP, RTP.

Є ще протоколи, які реалізують додаткову функціональність, наприклад, SIP TLS і Secure RTP, що додають шифрування сигналізації та медіа потоків, і інші, однак основними все ж є SIP, SDP і RTP.

Однак, якщо SIP протокол не визначає ніяких високорівневих функцій і сервісів, тоді що ж таке IP PBX на базі протоколу SIP?

1.4. Що вважати SIP АТС?

В наш час існує досить велика кількість телекомунікаційних програмних продуктів, які відрізняються один від одного архітектурою, цільовими функціями,

підтримуваними протоколами, популярністю, і іншими параметрами. Щоб зрозуміти, чи є вони системою IP PBX, треба розглянути їх відповідно до наступних критеріїв:

- Чи підтримує система функції SIP реєстратора? IP PBX повинна знати про місцезнаходження своїх користувачів, тому повинна реалізовувати функції SIP реєстратора.
- Чи підтримує система функції SIP проксі? IP PBX повинна займатися установкою з'єднань між своїми користувачами, а також підтримувати інформацію про стан цих з'єднань.
- Чи підтримує система механізми контролю над встановленою SIP сесією? IP PBX повинна мати можливість перервати поточну сесію при надходженні більш важливого дзвінка, або для звільнення зайнятої лінії, потрібної керівнику. В архітектурі SIP такі функції виконує так званий Back-to-back User Agent (B2BUA). При використанні B2BU зв'язок встановлюється не безпосередньо між двома користувачами, а між кожним з користувачів і B2BUA, і один дзвінок як би перетворюється в дві повністю незалежні SIP сесії.
- Чи підтримує система функції проксінгу RTP трафіку? IP PBX повинна пропускати через себе медіа потоки, наприклад, з метою запису розмов.
- Які додаткові додатки доступні користувачам? Традиційно PBX системи підтримують такі функції, як голосова пошта, конференц-зв'язок, музика при очікуванні, статистика дзвінків та інші.

Зробимо огляд безкоштовних IP PBX систем, які розповсюджуються в вихідному коді, які я й буду порівнювати відповідно до вищеперелічениї критеріїв. Найбільш популярними та зрілими IP PBX системами з відкритим кодом сьогодні є наступні: *ASTERISK, FreeSWITCH, SipXecs, Yate.*

Розглянемо їх детальніше.

1.4.1. ASTERISK

Проект ASTERISK був ініційований в 1999 році Марком Спенсером, власником і єдиним співробітником американської компанії Linux Support Services. Марк

займався системним адмініструванням і комерційною підтримкою Linux, а також програмував на C.

Один з клієнтів Марка звернувся до нього з питанням забезпечення офісної телефонії, і Марк Виявив, що офісні АТС стоять дуже багато грошей. І вирішив написати свою АТС на базі Linux. Так народився проект під назвою ASTERISK.

Через деякий час Марк заснував компанію Digium, яка стала виробляти плати сполучення ASTERISK з традиційними телефонними мережами (через аналогові і цифрові порти).

Навколо ASTERISK утворилося велике співтовариство користувачів і розробників, проект став активно розвиватися.

В даний час ASTERISK є найпопулярнішою відкритою IP АТС в світі, займаючи майже 85% «ринку» open source PBX.

Назву для ASTERISK (від англійського «зірочка») вибрали дуже вдало. В IT зірочка позначає заміщення будь-якого символу [7], або необмежену кількість символів. Навіть стандартні можливості ASTERISK викликають подив. Модульна архітектура ASTERISK дозволяє легко підключати до комутаційного поля будь-яку бізнес-логіку, написану на практично будь-якій мові програмування, або реалізовану власною мовою діалплана ASTERISK.

Наведемо скорочений список функціональних можливостей ASTERISK:

- Підтримуються як протоколи IP телефонії, так і традиційні лінії зв'язку. У сервер з ASTERISK можна вставити PCI плати Digium з аналоговими і/або цифровими портами в потрібній кількості і поєднанні. [9]
- Підтримуються всі базові та розширені функції АТС: голосове меню, запис розмов, статистика дзвінків, музика на утриманні, голосова пошта, постановка дзвінків в чергу і розподіл по операторам (функції кол-центру), і багато інших. [9]
- Безпосередньо підтримується Skype (драйвер каналу chan_skype від Digium), також є невеликий WEB додаток, що дозволяє викликати Skype користувачів з кнопкових телефонів через короткі номери.
- Підтримується відео зв'язок.
- Існують додатки розпізнавання голосу і генерації мови.

- Останні версії ASTERISK підтримується шифрування розмов.
- ASTERISK володіє в своєму арсеналі простими і добре документованими інтерфейсами для інтеграції з іншими системами [8] (AGI і AMI), що дозволяє легко вбудовувати комунікації в бізнес-процеси і бізнес-додатки.
- Існує велика к-сть всіляких графічних засобів адміністрування ASTERISK, як платних, так і безкоштовних, серед яких найбільш популярним є безкоштовний WEB інтерфейс FreePBX. Також є готові дистрибутиви, які дозволяють розгорнути на звичайному PC сервер IP PBX за лічені хвилини. [8] Найбільш популярними безкоштовними дистрибутивами ASTERISK є TrixBox, Elastix. Слід сказати, що компанія Digium, автор ASTERISK [7], пропонує також комерційне рішення на базі ASTERISK — SwitchVox [8], яке представляє з себе комплексне рішення уніфікованих комунікацій. Крім SwitchVox існує ще кілька десятків як комерційних, так і відкритих систем на базі ASTERISK.
- Нарешті, навколо ASTERISK зібрана дуже велика спільнота користувачів, розробників та інтеграторів, які допомагають один одному пізнавати і використовувати все різноманіття можливостей ASTERISK.

Нині ASTERISK продовжує стрімко розвиватися. Якщо ще кілька років тому комерційну підтримку або індивідуальну розробку під ASTERISK можна було отримати тільки в декількох компаніях, то сьогодні десятки компаній зі всіх надають послуги технічної підтримки і системної інтеграції заснованих на ASTERISK рішеннях, що повністю усунуло ризик використання вільного ПЗ в бізнесі — будь-яка компанія за розумні гроші може швидко отримати гарантовану допомогу фахівців з ASTERISK вищого класу, деякі з яких входять в перші десятки світових ASTERISK розробників.

Велика кількість можливостей ASTERISK і активний розвиток також є і мінусом цього продукту — початківцям складно швидко освоїти великий обсяг інформації.

Підсумовуючи слід сказати, що ASTERISK — це рішення IP PBX для офісу, хоча багато операторів зв'язку намагаються використовувати систему для надання

різних сервісів для своїх клієнтів. Але ASTERISK для цього не дуже підходить, оскільки не дуже добре масштабується.

1.4.2. FreeSWITCH

FreeSWITCH — це програмний комутатор, створення якого було ініційовано одним з колишніх розробників ASTERISK — Ентоні Мінессейлом (Anthony Minessale) в 2006 році. Після численних спроб використання ASTERISK під високим навантаженням, Ентоні висловив ряд зауважень до базової архітектури системи [9], і запропонував її змінити. Однак, автор ASTERISK — Марк Спенсер, відмовився змінювати ядро. Тому Ентоні вийшов зі складу розробників ASTERISK і створив «з нуля» свій продукт, який він назвав FreeSWITCH.

При розробці архітектури FreeSWITCH авторами були враховані всі проблеми існуючих відкритих програмних продуктів для IP телефонії.

Тому одною з головних переваг нового продукту стали стабільність роботи і масштабованість, а також крос-платформеність — FreeSWITCH працює під управлінням як Linux, так і Windows.

Іншою особливістю FreeSWITCH є використання SIP стека sofia-sip від Nokia[7], який вважається найкращою відкритою реалізацією SIP протоколу, поширюваної в вихідному коді. В ASTERISK ж chan_sip реалізований з неповним дотриманням страндартів. SIP є основним протоколом роботи FreeSWITCH, хоча також підтримуються і драйвери PCI плат для інтеграції з традиційною телефонією, а також інші протоколи IP телефонії.

FreeSWITCH може використовуватися як SIP проксі і SIP реєстратор, як Session Border Controller (SBC), транскодуючий Back-to-back User Agent (B2BUA), як сервер конференцій або голосової пошти.

Також FreeSWITCH підтримує і багато функцій IP PBX, такі як перенапрев-лення викликів, перехоплення, запис розмов, прослуховування та інші.

Однак, на сьогоднішній день список додатків IP PBX, доступний для FreeSWITCH, програє аналогічному в ASTERISK.

Основним інтерфейсом конфігурації FreeSWITCH є текстові файли у форматі XML, що ускладнює адміністрування цієї системи, тоді як в ASTERISK застосовуються добре читаємі і зручні .ini файли в форматі секція / опція.

Для FreeSWITCH відсутні готові до використання графічні інтерфейси з управління, що також ускладнює його використання. А існуючі GUI для FreeSWITCH (WikiPBX, FusionPBX, blue.box) далекі за функціональністю від того ж FreePBX для ASTERISK.

Проте, FreeSWITCH активно розвивається. Деякі експерти відкритих програмних продуктів для телекомунікацій називають FreeSWITCH «ASTERISK killer app», інші стверджують, що для обох продуктів є місце на ринку, так як у кожного з них своя унікальна специфіка.

Таблиця 1.1.

Мінімальні вимоги для АТС (до 15 телефонних номерів)

	ASTERISK	**FreeSWITCH**
Processor	Single Core, at least 700MHz	Single Core, at least 1GHz
RAM	512MB	1GB
Storage	10GB	10GB
Operating System	Linux based, 32 or 64 bit	Linux based, 64 bit

1.4.3 SipXecs

В основу продукту SipXecs закладений вихідний код ПО SipXpbx, опублікований у вільний доступ в 2004 році компанією PingTel.

Слід сказати, що фахівцями PingTel був створений один з найперших продуктів, за допомогою яких успішно взаємодіяли SIP пристрої від різних виробників. З тих пір SipXecs вважається найповнішою та правильною реалізацією SIP RFC.

Після старту SipXpbx, компанія PingTel продовжувала розвивати свій комерційний продукт, SIPxchange, періодично викладаючи у відкритий доступ різні частини коду і додаючи їх в SipXpbx.

У міру того, як до відкритого проекту підключалися активні розробники, стало складним підтримувати два різні продукти, так як поточна ліцензійна політика не дозволяла включати відкритий код, написаний ентузіастами, до складу комерційного продукту. Для вирішення цієї проблеми в 2007 році, PingTel змінює структуру проектів, і викладає іншу частину закритого коду в загальний доступ, об'єднуючи його з SipXpbx. Новий проект отримав назву SipXecs.

У 2008 році PingTel поглинається компанією Nortel. Nortel вже займалася постачанням своїм клієнтам продукту SCS (Software Communications System), заснованого на вихідному коді SipXecs. Фахівці Nortel внесли великий вклад як у розвиток свого комерційного продукту SCS, так і у відкритий проект SipXecs.

У 2009 році Nortel оголосила себе банкрутом, і права на комерційний продукт SCS перейшли до Avaya. У березні 2010 року Avaya припинила додавання своїх напрацювань у вихідний код SipXecs. Тоді спільнота користувачів SipXecs, включаючи деяких колишніх співробітників PingTel, об'єдналося під дахом свіжо створеної компанії eZuce, яка в даний час і займається підтримкою та розвитком проекту.

ПЗ SipXecs написано на мові програмування C++ і Java (на Java, зокрема написаний його SIP стек з використанням бібліотеки Jain SIP) і працює на ОС Linux.

Це єдина відкрита IP PBX система, в ядро якої з самого початку був включений WEB інтерфейс з управління. Якщо ASTERISK позиціонується як голосова платформа, то розробники SipXecs вважають свій продукт «коробочним» рішенням уніфікованих комунікацій.

Багатий арсенал ASTERISK знаходиться в великій кількості конфігураційних файлів всіляких модулів, а також у вбудованій командній строці з управління (CLI). SipXecs керується через WEB інтерфейс, і в ньому можливо зробити тільки те, що передбачено розробниками.

ASTERISK підтримує багато різноманітних телефонних інтерфейсів — аналогових, цифрових, кілька протоколів IP телефонії. SipXecs підтримує тільки SIP, будучи чистим SIP рішенням. Весь телефонний функціонал реалізований в рамках специфікації протоколу SIP, а також рознесений на повністю незалежні компоненти, які взаємодіють за протоколами SIP/HTTP/XML-RPC, і які можуть працювати як на

одному, так і на різних серверах, що, до речі кажучи, на новому рівні забезпечує надійність і масштабованість.

Якщо ASTERISK — «багатопротокольна» система, яка бере дзвінки з різних типів каналів, і перетворює їх в свій внутрішній формат з метою обробки та комутації (заміна старих АТС), то SipXecs — це SIP проксі, який займається маршрутизацією SIP транзакцій, не пропускаючи через себе медіа-потоки, а замикаючи їх безпосередньо між агентськими пристроями (IP телефонами).

Однак, з сильних сторін пакета SipXecs випливають і всі його слабкості. Бо ж не проксіруются медіа-потоки [9], що неуможливує реалізувати деякі важливі функції PBX, наприклад, запис розмов. Також, виникає проблема в тому випадку, коли користувач знаходиться всередині мережі з приватними IP адресами — проблема NAT. Також неможливо реалізувати транскодинг там, де це необхідно. Однак, цим проблеми в останніх версіях SipXecs [10] вирішуються за допомогою пакета FreeSWITCH, який органічно вписався в архітектуру SipXecs, виконуючи такі функції, як сервер конференц-зв'язку і IVR сервер.

1.4.4 Yate

Проект Yet Another Telephone Engine (Yate) було розпочато в 2004 році. Операційні системи: Linux, BSD, Windows. Написаний Yate на C ++. Yate не використовує зовнішніх SIP бібліотек, а реалізує SIP стек самостійно.

Yate — це софтсвітч, який містить також багато PBX функції, зокрема:

- переклад, утримання та парковку виклику;
- мелодію при очікуванні;
- конференц зв'язок;
- черги;
- IVR;
- статистику дзвінків.

Однак, Yate в першу чергу — це мультипротокольний комутатор з дуже гнучкими правилами маршрутизації. Yate добре підтримує такі протоколи IP телефонії,

як H323, IAX2, MGCP, різні рівні SS7 (MTP2, SIGTRAN), драйвера потокових цифрових плат різних виробників.

Архітектурно Yate використовує модель мікро ядра і шини повідомлень, а для маршрутизації повідомлень використовуються регулярні вирази з можливістю розміщення будь-яких повідомлень на шині. Така архітектура робить простим додавання нових модулів, не зачіпаючи існуючого коду. Yate — справжнісінький телефонний низькорівневий двигун (engine).

Існує спеціальний вільний дистрибутив з Yate і WEB інтерфейсом з управління — FreeSentral, що включає в себе інтерфейс користувача, де він керує своїми настройками, такими як переадресація, голосова пошта, записна книжка, а також може переглядати статистику своїх дзвінків.

Серед усіх розглянутих продуктів Yate володіє найменшим функціоналом, однак те, що Yate вміє робити, робить дуже добре і стабільно. Ще одним недоліком є недостатня документація.

Найбільш часте застосування Yate — конвертер H323-SIP сигналізації.

1.5 Сильні сторони ASTERISK

Ніколи за всю історію телекомунікацій не існувало системи, настільки відповідающій потребам бізнесу в будь-якій ціновій категорії. ASTERISK — технологія, що надає нові можливості, і, як це було з Linux, скоро навряд чи можна буде знайти підприємство, на якому не використовувалася б одна з версій ASTERISK, хоча б частково, десь в мережі, для вирішення проблем, які здатна вирішити тільки ASTERISK.

Однак схоже, що це визнання відбудеться швидше, ніж було у випадку з Linux, по ряду причин:

- Linux вже проклав шлях до визнання відкритого вихідного коду.
- Телефонія знаходиться в тяжкому становищі, жоден з крупних гравців на цьому ринку не є лідером. ASTERISK ж представляється переконливим, реалістичним і вражаючим проектом.

• Кінцеві користувачі вже ситі по горло несумісними системами з обмеженою функціональністю і жахливої підтримкою. ASTERISK вирішує перші дві проблеми — підприємці та спільнота забезпечать останнім.

Одна з незаперечних сильних сторін системи телефонії ASTERISK — співтовариство ентузіастів, які розробили і підтримують його, керує яким Марк Спенсер, засновник компанії Digium. Спільнота гостро усвідомлює культурну значимість ASTERISK і з захватом дивиться в майбутнє. Найбільш значимий результат діяльності спільноти розробників ASTERISK — об'єднання професіоналів з різних областей: телекомунікацій, мережевих та інформаційних технологій. Незважаючи на традиційну конфронтацію між цими галузями, в співтоваристві ASTERISK ці фахівці захоплюються здібностями один одного. Важливість такого співробітництва не можна недооцінювати.

Проте для реалізації мрії про ASTERISK співтовариство повинне розширюватися. На даний момент однією з основних проблем є швидкий приплив нових користувачів. Дійсні члени спільноти, які поклали початок того, що називається ASTERISK, в загальному, раді новим учасникам, але стурбовані тим, що їм доводиться стикатися з питаннями, відповіді на які можна знайти самостійно, якщо витратити трохи часу на пошуки і експерименти.

Очевидно, що всі нові учасники не можуть бути однаковими. Хтось буде радий проводити години, експериментуючи і читаючи різні блоги, описують чиїсь пригоди і муки. Але багато хто з тих, хто перейнявся цією технологією, абсолютно не зацікавлені вести такі пошуки. Вони хочуть мати просте і зрозуміле покрокове керівництво, яке введе їх в курс справи, супроводжуване деякими корисними прикладами, де описані кращі методи реалізації звичайної функціональності (такої, як голосова пошта, автосекретар та інше).

Експерти спільноти, які вважають ASTERISK свого роду мовою розробки вебдодатків, не сприймають такого підходу. Для них очевидно, що осягнути всі тонкощі ASTERISK можна, тільки повністю занурившись в неї. Хіба комусь спаде на думку просити покрокове керівництво з програмування і сподіватися навчитися по ньому всьому, що пропонує мову програмування?

Звичайно, немає єдиного підходу, який підійшов би всім. ASTERISK абсолютно ні на що не схожа і вимагає зовсім іншого типу мишленя. Знайомлячись з спільнотою, пам'ятайте, що в ньому зібралися люди з різними навичками і характерами. Деякі з цих хлопців НЕ відрізняются особливою стриманістю при спілкуванні з новачками, але це лише тому, що вони дуже трепетно відносяться до предмета, а не тому, що не раді вам.

ВИСНОВОК ДО РОЗДІЛУ 1

1. В даному розділі було розглянуто що таке IP-телефонія та PBX.
2. Розібрали архітектуру протоколу SIP та його допоміжні протоколи SDP та RTP.
3. Стало зрозуміло, що саме вважати за SIP АТС. Також зроблено огляд безкоштовних найбільш популярних IP PBX систем, які розповсюджуються в вихідному коді. Також було розглянуто сильні сторони ASTERISK.
4. Чому ми обрали ASTERISK:

- Моніторинг дзвінків в режимі online і запис розмов, можливість комутувати того чи іншого абонента через панель управління;
- Голосові привітання, динамічні черги розподілу вхідних дзвінків;
- Відсутність необхідності в покупці додаткових ліцензій і плат телефонії;
- Можливість інтегрування зі сторонніми системами використовуючи AMI/API інтерфейс, наприклад CRM системами і базами даних;
- Музика в очікуванні на вибір замовника;
- Розподіл дзвінків, в першу чергу ASTERISK — це IP-АТС з усіма можливостями VoIP телефонії.
- ASTERISK є дуже гнучким продуктом, що дозволяє налаштувати його практично під будь-які завдання компанії. Є дійсно революційним продуктом, що дозволяє не пропустити жодного дзвінка.

РОЗДІЛ 2
ПРОБЛЕМА, ЯКУ ВИРІШУЄ ASTERISK. СТРУКТУРА ASTERISK ТА ЙОГО ФУНКЦІЇ. ПРОЦЕС ВСТАНОВЛЕННЯ ASTERISK

2.1 Проблема, яку вирішує ASTERISK

Методологія розробки вузькоспеціалізованої системи телефонного зв'язку обумовлює наявність в ній безлічі функцій і те, що їх кількість буде багато в чому визначати ціну. Виробники обіцяють, що їхні продукти нададуть сотні функцій, і нікого хвилює, що вам потрібні лише п'ять з них. Але найгірше, що в цій системі може не бути однієї функції, без якої вам дійсно ніяк не обійтися, і це істотно зменшить цінність такої системи, тому що вона не зможе повністю задовольнити ваші потреби.

Той факт, що користувачеві можуть бути необхідні лише 5 з 500 функцій, не враховується, і бажання користувача мати 5 недоступних функцій, що відповідають вимогам його діяльності, відкидається як необґрунтоване. Поки гнучкість не стане нормою, телекомунікація залишиться в минулому столітті, незважаючи на всі технології VoIP в світі.

ASTERISK вирішує саме цю проблему, і вирішує її так, як можуть далеко не всі системи телефонного зв'язку. Це надзвичайно руйнівна технологія, в більшій мірі тому, що вона грунтується на принципах, що знаходять своє підтвердження знову і знову: «світ з закритим вихідним кодом не може виграти еволюційну гонку у спільнот, які дотримуються стратегії відкритого вихідного коду, які мають можливість вкласти у вирішення проблеми на порядки більше часу роботи кваліфікованих фахівців».

- Відкрита архітектура.

Однією з найважливіших проблем традиційних систем зв'язку було відверте небажання співпрацювати один з одним. Телекомунікаційні гіганти існують більш ста років. Принцип закритих вузькоспеціалізованих систем настільки проник в їх

культуру, що навіть спроби відповідати стандартам підірвані їх бажанням обігнати конкурентів, додавши хоча б одну таку функцію, яку більше ніхто не підтримує. Щоб побачити приклад подібного мислення, досить поглянути на VoIP-продукти, пропоновані сьогодні телекомунікаційною галуззю. Незважаючи на заявлене дотримання стандартів, ідея про те, щоб дійсно надати можливість підключення телефону Cisco до комутатора Nortel або інтеграції системи голосової пошти Avaya через протокол IP з офісною АТС Siemens, навіть не обговорюється.

У комп'ютерній галузі все інакше. Двадцять років тому при покупці IBM-сервера для взаємодії з ним необхідні були IBM-мережа та IBM-термінали. Зараз IBM-сервер, швидше за все, зможе з'єднуватись з терміналами Dellпо мережі Cisco. Можна навести масу подібних прикладів. Якби будь-яка з цих компаній заявила, що ми можемо використовувати їх продукти тільки з тим, на що вони нам вкажуть, їй довелося б піти з ринку.

Телекомунікаційна галузь переживає такі ж зміни, але не поспішає прийняти їх. ASTERISK, з іншого боку, дуже поспішає НЕ тільки прийняти зміни, але активно використовувати їх. IP-телефони Cisco, Nortel, Avaya і Polycom (і це далеко не повний список) були успішно підключені до систем ASTERISK. Сьогодні в світі немає іншої офісної АТС, яка могла б похвалитися цим.

- Миттєва відповідь на нові технології.

Після відвідування першого заходу, присвяченого тестуванню сумісності SIP (SIP Interoperability Test, SIPIT), Марк Спенсер за кілька днів створив для ASTERISK елементарний, але робочий блок, забезпечующий можливість взаємодії з бібліотекою SIP Stack. Це було ще до того, як SIP став кращим протоколом в світі VoIP, але Марк розгледів його цінність і потенціал та забезпечив готовність ASTERISK до його використання.

Таке вміння передбачати і гнучкість типові в співтоваристві розробників продуктів з відкритим вихідним кодом (і дуже нетипові для великих корпорацій).

2.2 Можливості ASTERISK

❖ Шлюз для традиційної офісної АТС

ASTERISK може бути фантастичним мостом від старої офісної АТС в майбутнє. Її можна розмістити перед АТС як шлюз (і виводити користувачів за кордони офісної АТС при необхідності) або за АТС як периферійний сервер додатків. Можна навіть зробити і те й інше одночасно, як показано на рис 2.1.

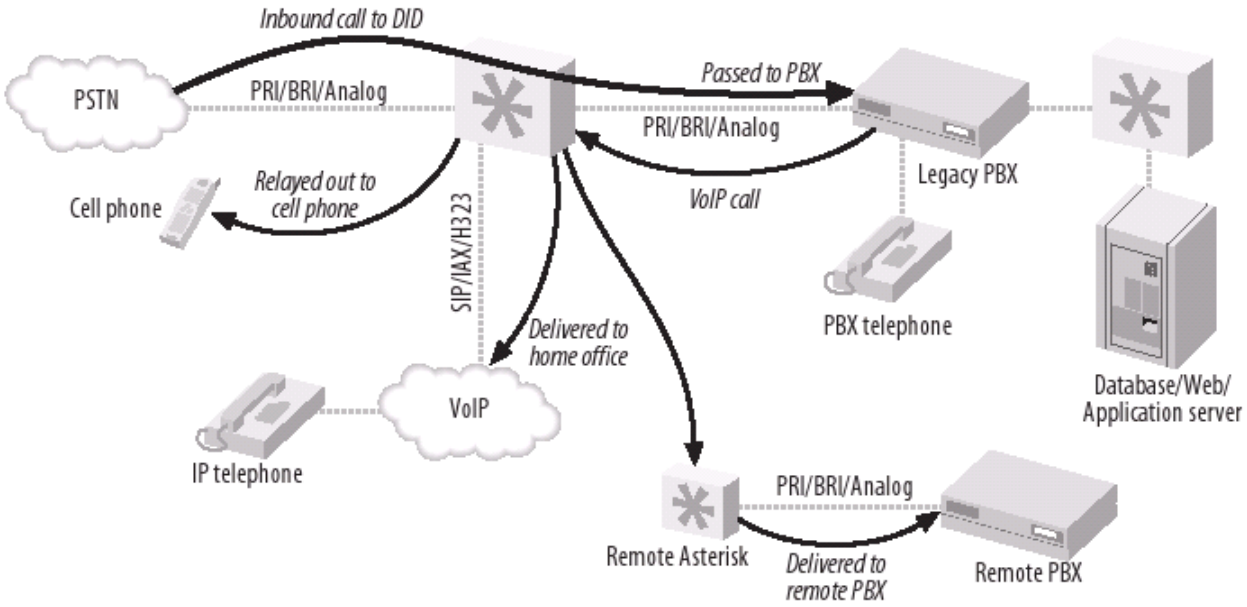

Рис. 2.1. ASTERISK в якості шлюзу офісної АТС

Ось деякі функції з тих, що можна реалізувати:

❖ Зберегти стару офісну АТС, але перейти на IP

Компанії, які витратили в останні кілька років величезні суми на придбання вузькоспеціалізованого обладнання офісних АТС, хочуть вирватися з в'язниці обмежень, але не можуть змиритися з думкою про те, що доведеться позбутися свого обладнання, що працює неефективно. ASTERISK може вирішити будь-які завдання, від заміни системи голосової пошти до надання можливості введення користувачів, підтримуючих IP, понад номінальної ємності системи.

❖ Знайди мене, йди за мною

Надайте офісній АТС список номерів, за якими можна з вами зв'язатися, — і вона буде дзвонити по ним під час надходження виклику на ваш DID (Direct Inward Dialing, тобто телефонний номер). Рис. 2.2. ілюструє цю технологію.

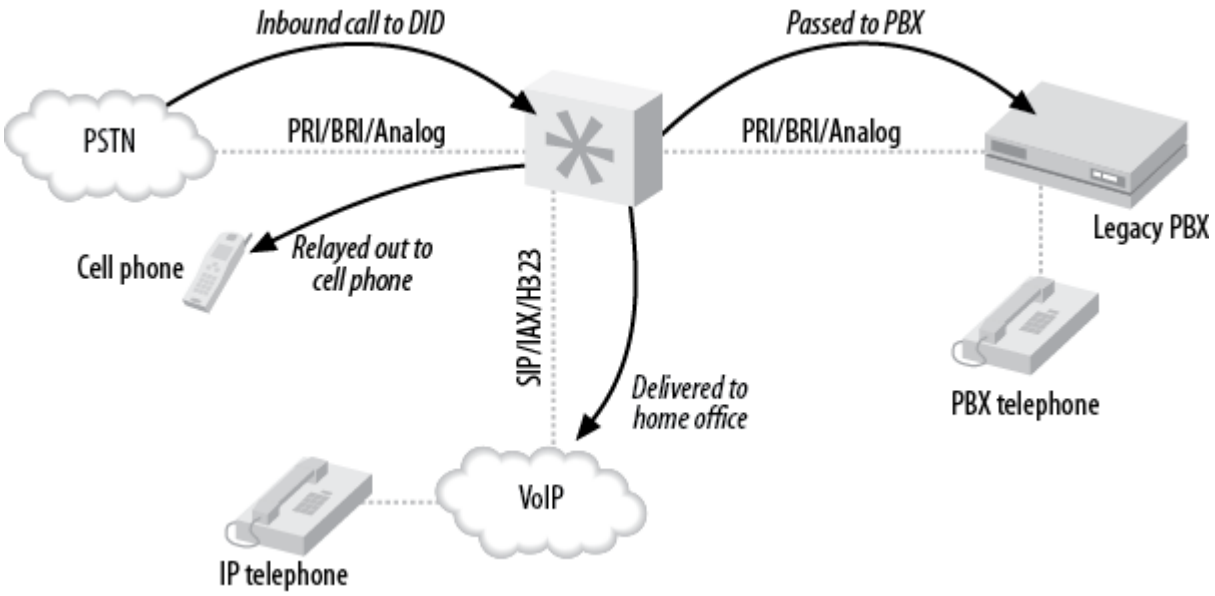

Рис. 2.2. Знайди мене, йди за мною

❖ Дзвінки по VoIP

Якщо можна встановити з'єднання по телефонній лінії між офісною АТС ASTERISK і старою офісної АТС, ASTERISK може забезпечити доступ до сервісів VoIP, тоді як стара офісна АТС продовжуватиме з'єднуватися із зовнішнім світом, як зазвичай. При використанні в якості шлюзу ASTERISK просто треба емулювати функції PSTN — і стара офісна АТС навіть не буде знати, що щось змінилося. На Рис. 2.3. показано, як можна використовувати ASTERISK

для забезпечення підтримки VoIP в застарілій офісній АТС.

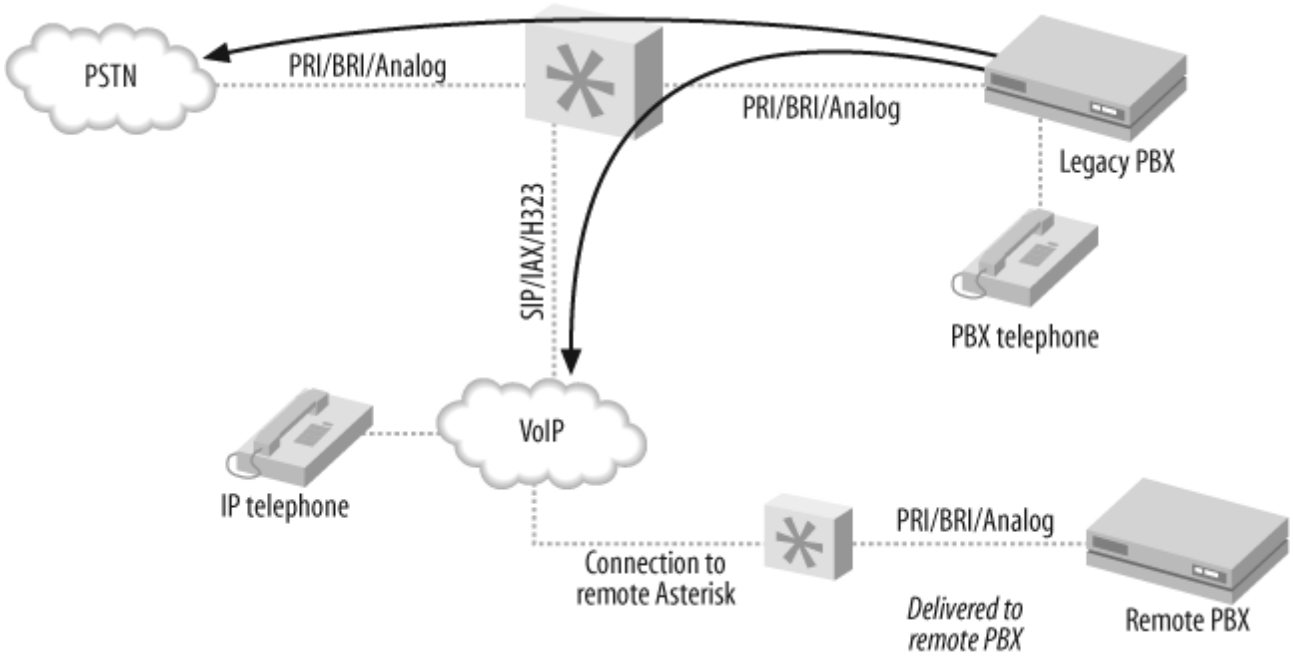

Рис. 2.3. Підтримка VoIP в застарілій офісній АТС

❖ Доступний IVR

Багато хто плутає інтерактивний автовідповідач (Interactive Voice Response, IVR) зі звичайним автовідповідачем (Automated Attendant, AA). Оскільки спочатку IVR використовувався як автовідповідач, то це не дивно. Проте поняття IVR в телефонії набагато ширше, ніж просто автовідповідач. AA є, засобом для перенаправлення абонента на додатковий номер і вбудований в більшість вузькоспеціалізованих систем голосової пошти. Але функціонал IVR може бути набагато більшим.

Системи IVR зазвичай не просто дорогі за ціною, їх конфігурація дуже складне. Для спеціальної системи IVR зазвичай потрібна можливість з'єднання з зовнішньої базою даних або додатком.

ASTERISK — мабуть, ідеальний IVR, оскільки в ній закладена можливість з'єднання з базами даних і додатками на найглибшому рівні.

Ось кілька прикладів щодо простих IVR, для створення яких можна використовувати систему ASTERISK:

❖ Отримання прогнозу погоди

Використовуючи Інтернет, можна отримувати прогноз погоди в текстовому

вигляді з усього світу сотнями способів. Якщо взяти ці звіти і обробити спеціальним синтаксичним аналізатором (сценарій Perl, ймовірно, міг би з цим впоратися), цю інформацію можна зробити доступною для діалплана. Фонотека ASTERISK вже має всі необхідні голосові повідомлення, тому не важко створити

інтерактивне меню для відтворення поточних метеорологічних прогнозів для будь-якої точки земної кулі.

- ❖ Математичні програми

Ед Гай (Ed Guy, архітектор мережі FWD (Free World Dialup — безкоштовні дзвінки по всьому світу) Пулвера) представив на конференції AstriCon 2004 невелику математичну програму, яку створив для своєї дочки. На її написання пішло не більше години. Вона представляла ряд математичних питань, відповіді на які вводилися за допомогою номеронабирача телефону. Після відповіді на всі питання система повідомляла оцінку. Реалізація такого гранично простого ASTERISK-додатку в будь-якої офісній АТС на закритій платформі коштувала б десятки тисяч доларів, якщо взагалі могла б бути виконана. Як це часто буває, то, що просто для ASTERISK, практично неможливо або надзвичайно дорого в будь-який інший системі IVR.

- ❖ Розподілені IVR

Ціна вузькоспеціалізованої системи IVR така, що, якщо компанія, що має безліч торгових точок, хоче забезпечити функціональність IVR, для обробки транзакцій вона змушена перенаправляти абонентів на центральний сервер. З ASTERISK стає можливим поширити додаток на всі сервери й обробляти запити локально. Буквально тисячі маленьких систем ASTERISK, розгорнутих в торгових точках по всьому світу, могли б забезпечувати функціональність IVR так, як не здатна жодна інша система. І не потрібно ніяких віддалених

переадресацій на центральний сервер IVR або використовуваних спеціально для цієї мети гігантських магістральних каналів — більше можливостей за менші гроші.

- ❖ Конференц-зали

Ця маленька можливість в кінці кінців стане однією з найбільш приголомшливих функцій ASTERISK. У співтоваристві розробників ASTERISK конференц-зали використовуються для різних цілей все частіше:

- ❖ Невеликим компаніям необхідний простий спосіб забезпечити можливість діловим партнерам зібратися і обговорити деякі питання.
- ❖ Групи збуту зустрічаються раз на тиждень, при цьому кожен повинен мати можливість зателефонувати звідти, де він знаходиться в даний час.
- ❖ Групи розробки призначають єдине місце і час для передачі один одному інформації про внесені зміни.
- ❖ Побутова автоматизація

ASTERISK як і раніше багато в чому є інструментом для суперфанатів і тому навряд чи може використовуватися в звичайному будинку, але при наявності у вас неабияких знань з Linux і ASTERISK стає можливим наступне.

- ❖ Контроль за дітьми

Батьки, які бажають контролювати няню (або дітей, які залишилися вдома самих), можуть дзвонити в контекст додаткового номера, захищеного паролем. Після аутентифікації буде встановлюватися двосторонній аудіо-зв'язок з усіма IP-телефонами в будинку — це дозволить мамі і татові чути, що там відбувається.

- ❖ Блокування телефонів

Невелика зміна діалплана — і можна дзвонити тільки на телефони екстреного виклику, ваш мобільний телефон і в піцерію. Спроби зателефонувати на будь-який інший телефон приведуть до відтворення запису.

- ❖ Управління системою сигналізації

Дзвінок в свою систему ASTERISK, передача короткого рядка цифр в спеціально створений для цієї мети контекст — і ваша система сигналізації отримує вказівку відключити сигналізацію на зазначений час.

- ❖ Управління дзвінками дітей

Можливо встановити тимчасове обмеження на дзвінки. А щоб використовувати телефон, треба буде ввести свій код доступу. За допомогою Caller ID (ID абонента) можна управляти і вхідними дзвінками.

2.3 Структура системи

«Астеріск, як будь-який додаток, працює на підставі конфігураційних файлів, яких досить велика кількість. На початковому етапі перш за все знадобиться знати і розуміти налаштування, що зберігаються в файлах sip.conf і extensions.conf. » [10] Зазначу, що при стандартній установці всі конфігураційних файли зберігаються в /etc/asterisk/.

Файл sip.conf

Даний файл містить опис базових параметрів роботи протоколу SIP, настройки NAT, кодеків і найголовніше — облікових записів. [5]

Дефолтний конфіг sip.conf дозволяє запускати Астеріск, за умови прописування облікових записів [6], і він буде працювати, слухаючи при цьому вхідні з'єднання на всіх інтерфейсах.

«Облікові записи в sip.conf можуть бути трьох типів:

- peer: SIP запис, яку ASTERISK може використовувати для здійснення вихідних дзвінків (наприклад, SIP провайдер).
- user: SIP запис, через яку виклики можуть надходити ззовні в ASTERISK (телефон, який може тільки здійснювати виклики).
- friend: запис, який одночасно user і peer. Цей тип найбільш підходить для телефонів і інших пристроїв. » [5]

Простий приклад облікового запису виглядає так:

[1001]; ім'я, асоційоване з SIP клієнтом, або це може бути довільне ім'я SIP пристрою, на який можна посилатися з інших конфігураційних фалів

«type = friend; тип облікового запису

host = dynamic; дозволяємо логін з різних IP адрес

username = 1001; ім'я користувача

secret = 1234; пароль

context = default; цей контест повинен бути визначений в extensions.conf

disallow = all; забороняємо всі кодеки

allow = alaw; дозволяємо кодек G.711a (alaw) » [7]

Даного запису досить, щоб зареєструватися на свіжому сервері з параметрами користувача 1001 і паролем 1234.

Створюємо два записи з назвами 1001 і 1002 в файлі sip.conf.

Файл extensions.conf

Даний файл — ключовий файл в системі ASTERISK, так як в ньому описуються правила роботи з будь-яким голосовим трафіком, що з'явився в системі.

Структура файлу цілком проста:

- всередині файлу прописуються глобальні параметри, в тому числі і змінні [5]

- далі файл розбитий на контексти [6], кожен з яких живе своїм життям і правила роботи дзвінків всередині кожного контексту можуть бути своїми.

Контексти потрібні [5], для розмежування здійснення телефонного дзвінка та відділення одних груп користувачів, та їх дзвінків від інших. Контексти дозволяють в межах однієї системи створювати безліч підсистем зі своїми правилами і користувачами (аналог віртуальних машин).

Контексти позначаються як [НАЗВА КОНТЕКСТУ].

«Структура контексту наступна:

exten => МАСКА, ПОРЯДОК, ДІЯ, де

МАСКА — маска номера, або статично заданий номер

ПОРЯДОК — порядок дії для конкретної МАСКИ (можлива послідовність дій)

ДІЯ — команда, яка виконується системою в разі потрапляння дзвінка на дане правило.» [5]

Всі створені тестові користувачі в контексті default, тому, якщо просто запустити Астеріск і зателефонувати на будь-який номер, можна почути тестове демоменю Астеріск.

Відкриваємо для редагування extension.conf, виконуємо пошук по [default] і в наступному після рядка [default] вносимо правило локального дзвінка:

«Статичне правило:

exten => 1001,1, Dial (SIP / 1001)

exten => 1002,1, Dial (SIP / 1 002)

Динамічне правило:

exten => _XXXX, 1, Dial (SIP / $ {EXTEN}); тут X говорить про те що буде набрана будь-яка цифра. » [9]

Дані правила дозволять зробити локальний дзвінок між тестовими користувачами.

2.4 Функції ASTERISK

1. Встановлення музики замість гудка.

Дана функція дозволяє щоб людина, яка нам дзвонить чула музику, а не гудок.

Ми можемо покласти в папку багато музичних файлів, тоді вони будуть програватися по черзі, то один, то інший.

2. Створення інтерактивного (голосового) меню.

Це є інтерактивною довідковою системою, яка здатна розпізнавати запити користувачів і переміщати їх в мовному меню через натискання клавіш мобільного пристрою.

3. Перенаправлення викликів.

Трапляється так, що наприклад секретар отримав дзвінок і секретарю цей дзвінок потрібно направити, наприклад менеджеру. Тоді нам знадобиться функціє перенаправлення викликів.

Існує два способи blind transfer і attended transfer:

- **Blind transfer** використовується для сліпого перенаправлення дзвінків і працює за замовчуванням. Це коли секретар переводить дзвінок менеджеру і секретарю все одно, що трапиться з дзвінком далі. До неї цей дзвінок вже ніколи не повернеться. Тобто це проста переадресація, без зворотного зв'язку.
- **Attended transfer** дозволяє секретарю не просто перенаправити дзвінок, а й контролювати успішність його перенаправлення. Це коли секретар перенаправляє дзвінок менеджеру. Якщо менеджер не відповідає протягом заданого кількості часу, або менеджер просто скинув виклик, то дзвінок повертається назад до секретаря.

4. Запис розмов.

Це дозволяє записувати розмови коли ми звонимо, а також коли дзвонять нам. Запис буде збережений після чого ми зможемо його прослуховувати.

5. Простий автовідповідач.

Автовідповідач запропонує абонентові залишити для голосове повідомлення, якщо ви спілкуєтесь з іншою людиною або ваш телефон знаходиться поза мережею.

6. Встановлення системи перегляду статистики дзвінків.

Це знадобиться, коли потрібно передивитись хто, коли й кому дзвонив.

7. Конференц-зв'язок.

Конференц-зв'язок, це функція, при якій кілька людей можуть розмовляти один з одним одночасно.

У ASTERISK це виглядає наступним чином:

1. Співробітник організації, який хоче підключитися до конференції дзвонить на певний номер;

2. Далі він вводить пароль до конференції;

3. Все. Співробітник є учасником конференції.

Теж саме роблять і інші учасники конференції, таким чином формується кімната конференц-зв'язку.

Притому є така функція: конференція може початися відразу, коли кількість учасників становитиме від двох осіб, або у співробітників, які підключилися до конференції буде грати музика, поки не прийде лідер конференції.

8. Парковка викликів.

Парковка викликів — це:

Вам хтось телефонує і Ви піднімаєте трубку, однак, Вам потрібно поговорити з цією людиною з іншого місця і з іншого телефону. Наприклад, Вас попросили підійти до серверної, а серверна знаходиться в іншому корпусі будівлі, АЛЕ — там є телефон, підключений до ASTERISK, тоді Ви паркуєте виклик. Йдете в серверну і там вже піднімаєте трубку, продовжуючи розмову.

Технічно це відбувається наступним чином:

1) Вам подзвонили і Ви зняли трубку;

2) Далі Ви робите сліпий переклад (blind transfer), тобто просто переводите дзвінок на інший номер;

3) ASTERISK повідомляє паркувальний номер;

4) Ви підходите до іншого телефону, телефонуєте на номер про який повідомив ASTERISK і продовжуєте розмову зі співрозмовником.

9. Переадресація дзвінків.

Переадресація дзвінків працює наступним чином:

Вам телефонують на ваш внутрішній номер. Якщо протягом встановленого часу Ви не відповідаєте, то виклик переводиться на Ваш мобільний.

10. Черга дзвінків.

Приклад черги дзвінків можно спостерігати, якщо зателефонуємо, наприклад в який-небудь Call центр. Там нам кажуть, будь ласка, залишайтеся на лінії, скоро вам дадуть відповідь і ви слухаєте музику в трубці телефону, поки який-небудь оператор не з'єднається з вами.

Така ж функція є і в ASTERISK. Вона підходить великим організаціям в яких є Call центри. Наприклад великий провайдер куди надходить сотні дзвінків протягом дня і на їх обробці сидить кілька людей, або це велика торгова компанія де йде сотні дзвінків від клієнтів.

11. Робота ASTERISK в залежності від дня тижня і часу доби

Існує можливість зробити так, щоб у поза робочий час телефони просто так не дзвонили, а клієнту програвалося повідомлення про те, що зараз не робочий час з можливістю залишити повідомлення на голосову пошту.

Загальний принцип роботи такий:

1) Якщо не робочий час — програємо повідомлення про те, що час не робочий і просимо залишити повідомлення

2) Якщо час робочий — ASTERISK працює в звичайному режимі.

2.5 ASTERISK та VoIP

ASTERISK любить працювати з VoIP, але для цього йому треба знати, яку функцію виконувати: кліента, сервера або і того, й іншого. Одна з найбільш складних концепцій в ASTERISK — схема привласнення імен при аутентифікації вхідних і вихідних дзвінків.

З'єднання, що встановлюються з нами або нами, визначені в файлах iax.conf і sip.conf як user (користувач) та peer (рівноправний учасник). З'єднання, які можуть виконуватися в обох напрямках, можуть бути визначені як friend (друг). При визначенні, в якому напрямку відбувається аутентифікація, завжди важливо подивитися на напрям каналів з точки зору ASTERISK, оскільки з'єднання приймаються і створюються сервером ASTERISK.

- ***З'єднання user***

З'єднання, певне як user, — це будь-яка система/користувач/кінцева точка, якій ми дозволяємо з'єднуватися з нами.Опис user не забезпечує методу виклику цього користувача; тип user використовується просто для створення каналу для вхідних дзвінків. В описі user потрібно задати ім'я контексту для позначення місця діалплана (в файлі extensions.conf), де буде починатися обробка аутентіфіцированних дзвінків.

- ***З'єднання peer***

З'єднання типу peer є вихідним. Уявімо це так: користувачі (users) дзвонять нам, тоді як ми дзвонимо рівноправним учасникам (peers). Оскільки рівноправні учасники не дзвонять нам, опис peer зазвичай не вимагає задання імені контексту. Однак є один виняток: якщо дзвінки, що беруть початок у вашій системі, повертаються в вашу ж систему, вхідні дзвінки (які беруть початок на SIP-проксі, а не на агента користувача) будуть зіставлятися з описом peer. Контекст default повинен обробляти ці вхідні дзвінки відповідним чином, хоча краще, щоб контексти були визначені для кожного peer окремо. Щоб знати, куди відправляти виклик, необхідно мати інформацію про місцезнаходження хоста в Інтернеті (тобто знати його IP-адресу). Місцезнаходження peer може бути визначена або статично, або динамічно.

Динамічний peer конфігурується за допомогою рядка host = dynamic, що розміщується під заголовком опису. Оскільки IP-адреса динамічного peer може змінюватися постійно, він повинен реєструватися на сервері ASTERISK, щоб його IP-адреса була відома й дзвінки могли успішно прямувати до нього. Якщо віддаленим кінцем є інший сервер ASTERISK, необхідно використовувати вираз register, що обговорюється нижче.

Вираз register — це засіб повідомити віддаленого рівноправному учаснику мережі, де в Інтернеті знаходиться ваш сервер ASTERISK. ASTERISK використовує вирази register для аутентифікації в віддалених постачальників сервісів, якщо ви використовуєте динамічні IP-адреси або якщо ваша IP-адреса не зареєстрована у постачальника.

- ***З'єднання friend***

Визначення типу friend є скороченим записом для з’єднання, яке може бути і user, і peer. Однак з'єднання, яке є і user, і peer, не завжди визначаються так, тому що індивідуальний опис кожного напряму створення виклику (використання

двох описів, user і peer) забезпечує можливість більш тонкого налаштування й управління кожним окремо взятим з'єднанням. На рис. 2.4. показаний потік управління аутентифікацією по відношенню до ASTERISK.

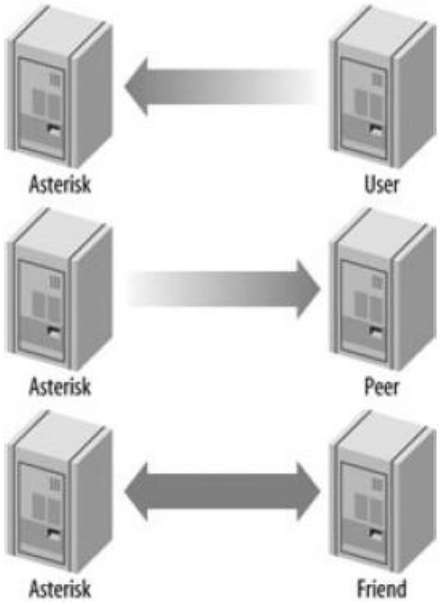

Рис. 2.4. Джерело виклику щодо ASTERISK для з'єднань типу user, peer і friend

2.6 Встановлення ASTERISK

a) Відключаємо систему безпеки SELinux:

sed -i s/SELINUX=enforcing/SELINUX=disabled/g /etc/selinux/config

b) Встановлення необхідних компонентів для встановлення ASTERISK:

yum install -y make wget openssl-devel ncurses-devel newt-devel libxml2-devel kernel-devel gcc gcc-c++ sqlite-devel

c) Завантажуємо вихідний код ASTERISK. Для цього переходимо в папку:

cd /usr/src/

та завантажуємо за допомогою команди wget:

wget http://downloads.asterisk.org/pub/telephony/dahdi-linux-complete/dahdi-linux-complete-current.tar.gz

wget http://downloads.asterisk.org/pub/telephony/libpri/libpri-1.5.0.tar.gz

wget http://downloads.asterisk.org/pub/telephony/asterisk/asterisk-11-current.tar.gz

d) Розпаковуємо скачані архіви:

tar zxvf dahdi-linux-complete*

tar zxvf libpri*

tar zxvf asterisk*

e) Встановлюємо LibPRI

cd /usr/src/libpri*

make && make install

f) Переходимо в директорію, в яку розпакували ASTERISK:

cd /usr/src/asterisk*

g) Запускаємо конфігураційні скрипти для ASTERISK. Для цього, початку дізнаємося якої бітності наш ASTERISK.

Набираємо:

uname –a

Якщо відповідь: 2.6.18-238.12.1.el5 #1 SMP Tue May 31 13:23:01 EDT 2011 i686 i686 i386 GNU/Linux – то значить 32 біти

Якщо відповідь: 2.6.18-238.19.1.el5 #1 SMP Fri Jul 15 07:31:24 EDT 2011 x86_64 x86_64 x86_64 GNU/Linux – то значить 64 біти

В залежності від того, яка бітність ASTERISK, запускаємо конфігураційний скрипт:

Для 32:

./configure && make menuselect && make && make install

Для 64:

./configure --libdir=/usr/lib64 && make menuselect && make && make install

Про успіх встановлення свідчить синє вікно.

h) Додаємо підтримку дзвінків. Справа в тому, що при такій конфігурації ASTERISK начебто як працює, але дзвінки здійснюватися не будуть.

Виникатиме помилка

[Apr 27 21:35:51] ERROR[1225][C-00000009]: rtp_engine.c:259 ast_rtp_instance_new: No RTP engine was found. Do you have one loaded?

Тому, робимо наступне:

yum install uuid uuid-devel libuuid libuuid-devel uuid-c++

після цього:

./configure

make menuselect

та потім:

make

make install

i) Далі встановлюємо образ. Без установки цих образів у нас не з'являться конфігураційні файли sip.conf і extensions.conf

make samples

make config

j) друкуємо

cd

після друкуємо:

reboot

k) Після перезавантаження пишемо

asterisk

В результаті ASTERISK має запуститися. Вітаю! Ми запустили ASTERISK.

- Тепер необхідно фаєрвол в самій CentOS.

Для цього пишемо команди:

service iptables save

service iptables stop

chkconfig iptables off

- Переходимо безпосередньо до налаштування sip.conf:

nano /etc/asterisk/sip.conf

У нас відкривається файл. Пишемо свої конфіги в самий початок файлу. У моєму випадку це визначення двох sip клієнтів (телефонів):

[1001]

type=friend

regexten=1001

secret=1234

context=outcoling

host=dynamic

callerid="1001" <1001>

disallow=all

allow=alaw

allow=ulaw

language=ru

callgroup=1

pickupgroup=1

qualify=yes

canreinvite=yes

call-limit=4

nat=no

[1002]

type=friend

host=dynamic

insecure=invite

username=1002

secret=45678

context=outcoling

disallow=all

allow=alaw

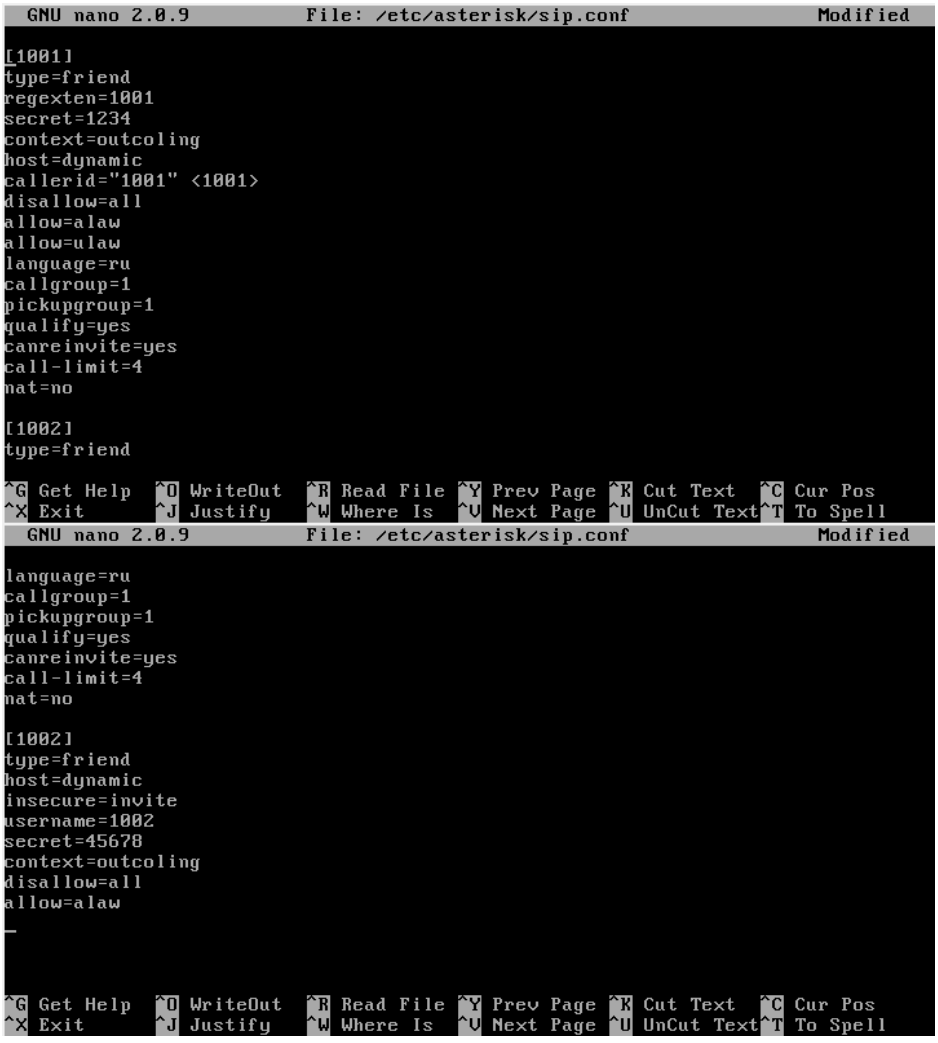

Рис. 2.5. Налаштування sip.conf

Після цього знаходимо секцію [general] і видаляємо її. Так само видаляємо напис «context = public» після напису [general].

Звертаємо увагу на контексти.

Для телефонів (sip клієнтів) [1001] і [1002] це outcoling.

Що таке контекст і навіщо він потрібен? Контекст пов'язує файл sip.conf з файлом extensions.conf. Тобто якщо у [1001] прописаний контекст outcoling, то [1001] буде шукати правило в extensions.conf під назвою outcoling.

Натискаємо ctrl + x, натискаємо y і натискаємо enter. Файл збережений.

- Переходимо до редагування файлу extensions.conf

nano /etc/asterisk/extensions.conf

В кінці файлу пишемо наш діаплан, після чого зберігаємо файл:

[outcoling]

exten => _XXXX,1,Dial(SIP/${EXTEN},,m)

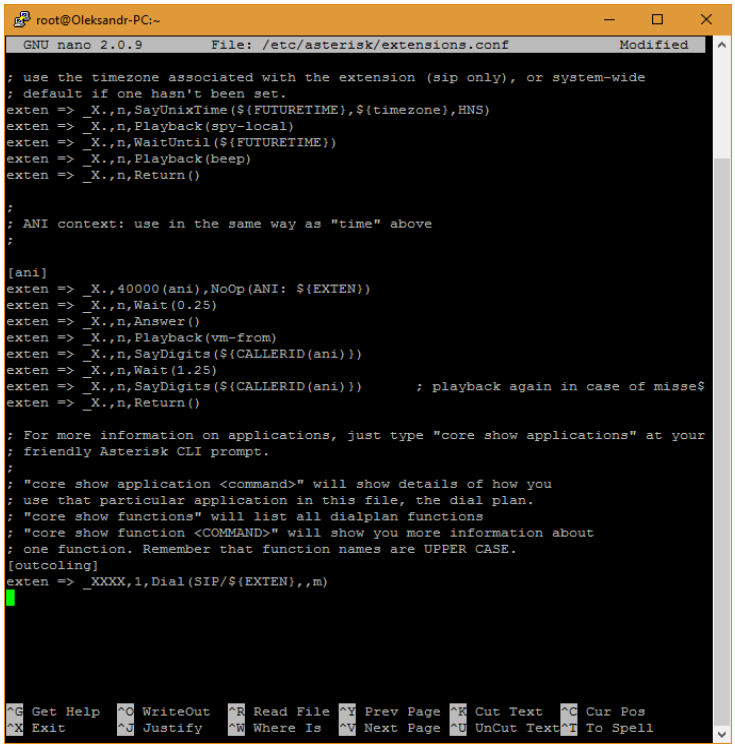

Рис 2.6. Налаштування extensions.conf

- Далі пишемо:

asterisk –r

Це ми з управління Linux перейшли в управління ASTERISK (як нібито відкрили вікно програми і почали з нею працювати).

- Пишемо

core reload

Тим самим ми змусили ASTERISK перерахувати всі конфігураційні файли і прийняти зміни.

Тепер законнектившись софтфоном до Астеріск можно зробити перший телефонний дзвінок між двома внутрішніми абонентами.

ВИСНОВОК ДО РОЗДІЛУ 2

1. Даний розділ був направлений на вивчення структури та функцій ASTERISK.
2. Було наведено перелік проблемм, які вирішує ASTERISK.
3. Також приведено його можливості.
4. Було представлено структуру системи й наведені наступні функції ASTERISK: встановлення музики замість гудка, створення інтерактивного (голосового) меню, перенаправлення викликів, запис розмов, простий автовідповідач, встановлення системи перегляду статистики дзвінків, конференц-зв'язок, парковка викликів, переадресація дзвінків, черга дзвінків, робота ASTERISK в залежності від дня тижня і часу доби.
5. Можно зрозуміти як ASTERISK співрацює з VoIP та їх режими роботи.
6. В кінці було наведено послідовність дій завдяки яким ми можемо встановити ASTERISK на наш ПК.

РОЗДІЛ 3
РОЗРОБКА СИСТЕМИ РВХ ТА ЇЇ ДОСЛІДЖЕННЯ

3.1 Конфігурація ASTERISK на роботу через транк

Прийом дзвінків з зовнішніх телефонів. Для цього ми виконали наступні дії:

a) Укладаємо договір з sip провайдером (провайдером ip телефонії). Отримуємо від нього дані. У нашому випадку це zadarma. Після реєстрації отримуємо свої налаштування SIP:

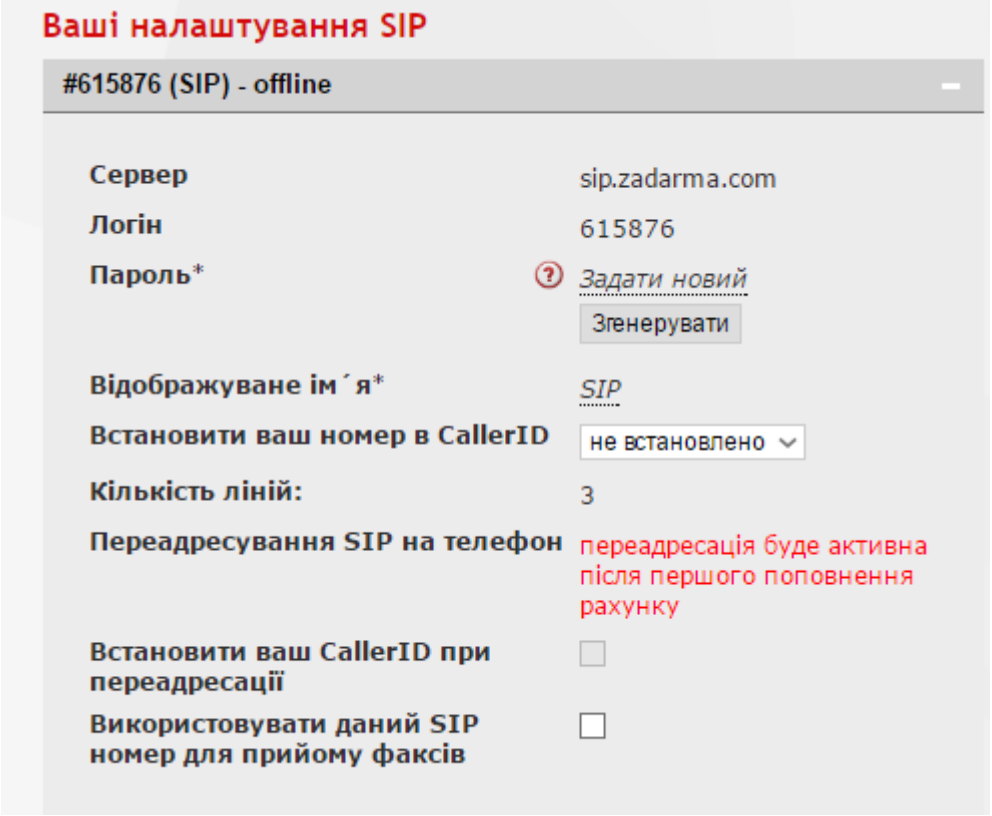

Рис. 3.1. Налаштування SIP

b) Тепер заходимо в файл sip.conf

nano /etc/asterisk/sip.conf

І над sip клієнтами [1001] та [1002] пишу наступний код:

[general]

register => 00000:password@sip.zadarma.com/00000

[zadarma]

type=friend

username=00000

secret=password

fromuser=00000

fromdomain=sip.zadarma.com

host=sip.zadarma.com

nat=yes

insecure=invite

context=incoming

canreinvite=no

де 00000 – це логін на сайті провайдера, а password – пароль звідти ж.

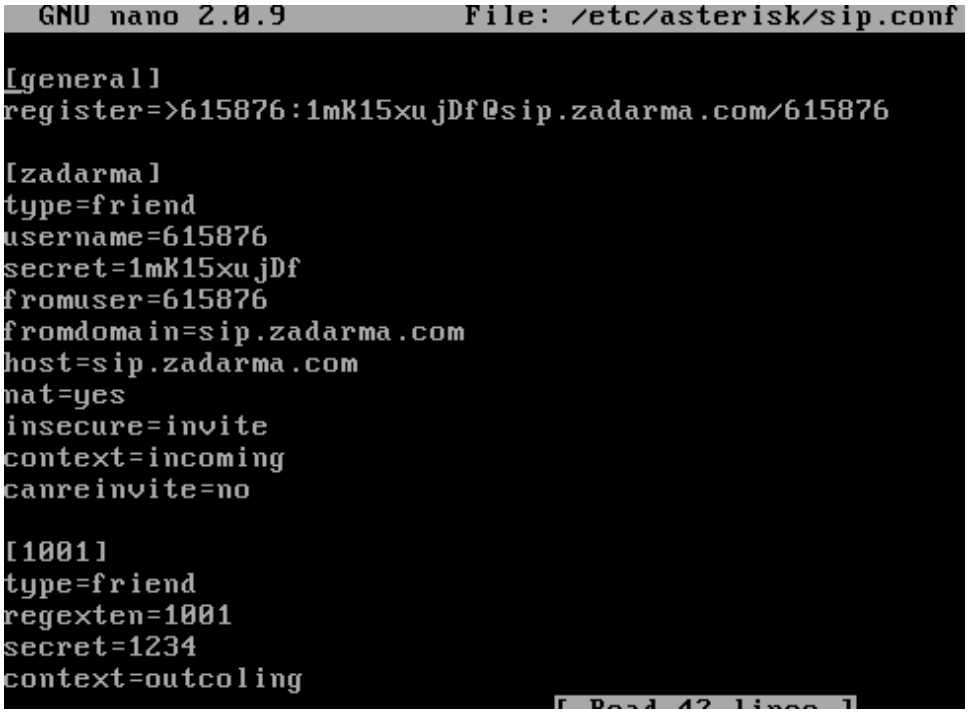

Рис. 3.2. Редагування sip.conf

c) Зберігаємо файл і заходимо в extensions.conf

nano /etc/asterisk/extensions.conf

До контексту [outcoling] додаю наступне:

exten => _XXXXXXXXXXX,1,Dial(SIP/zadarma/${EXTEN})

Крім того, після контексту [outcoling] додаю ще один контекст:

[incoming]

exten => _X.,1,Dial(SIP/1001&SIP/1002,60,m,tT)

Все разом це виглядає так:

[outcoling]

exten => _XXXXXXXXXXX,1,Dial(SIP/zadarma/${EXTEN})

exten => _XXXX,1,Dial(SIP/${EXTEN},,m)

[incoming]

exten => _X.,1,Dial(SIP/1001&SIP/1002,60,m,tT)

Рис. 3.3. Редагування extensions.conf

d) Зберігаємо файл та пишемо: asterisk –r

e) Перезапускаємо нашу ОС та пишемо: sip show registry

Таким чином ми перевіряємо «чи піднявся транк», тобто визначаємо чи з'єднався ASTERISK з провайдером ip телефонії. Якщо все ОК, у відповідь на команду sip show registry отримаємо відповідь: 1 SIP REGISTRATION

Тепер все готово й можна зробити дзвінок з ASTERISK наприклад на мобільний телефон і з мобільного телефону на ASTERISK.

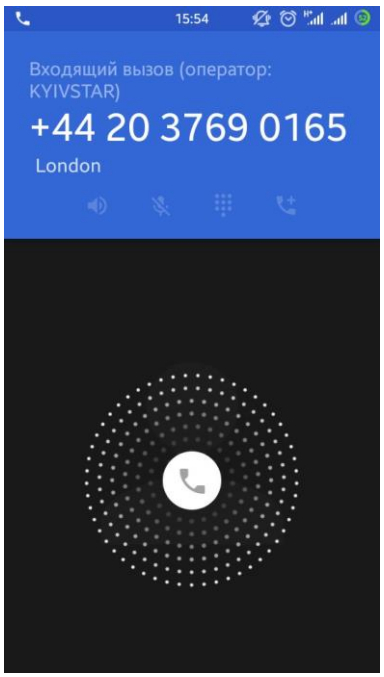

Рис. 3.4. Ілюстрація дзвінка на мобільний через транк

3.2 Реалізація функції перенаправлення викликів

Для цього виконуємо такі дії:

В extensions.conf в Dial додаємо параметр t. Цей параметр означає, що для цього Dial плану дозволено перенаправлення дзвінків.

Це буде виглядати так:

exten => _XXXX,1,Dial(SIP/${EXTEN},,t&m,)

Рис. 3.5. Редагування extensions.conf

Тут можна бачити параметр t & m. Тобто відразу два параметри — t для перенаправлення дзвінків і m для музики.

Отже, в контексті для Dial ми задали параметр t. Тепер у нас працює сліпий трансфер (blind transfer).

Для Attended transfer все складніше:

Переходимо до редагування файла features.conf

nano /etc/asterisk/features.conf

Знаходимо рядки:

atxfernoanswertimeout = 15

atxferdropcall = no

atxferloopdelay = 10

atxfercallbackretries = 2

atxfer => *2

та розкоментуємо їх (видаляємо крапку з комою на початку рядка)

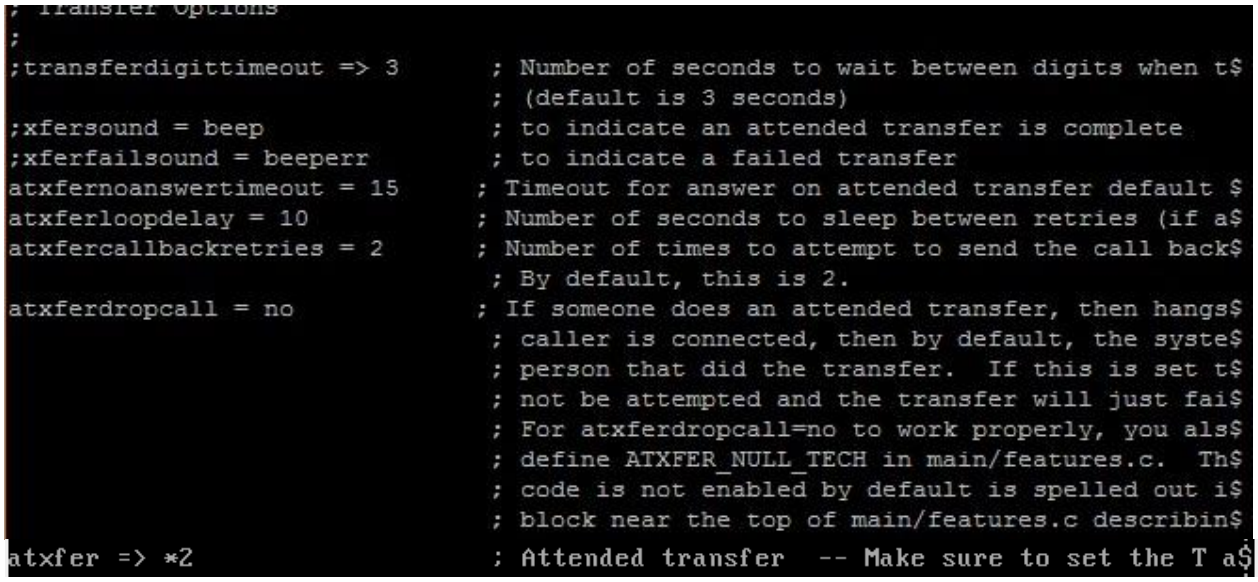

Рис. 3.6. Редагування features.conf

Тепер можна зробити дзвінок і перенаправити його.

1) Дзвонимо з мобільника на Астеріск на номер «people1».

2) Встановлюємо з'єднання. «People1» каже співрозмовнику, щоб почекав, поки він перемкне

3) «People1» набирає на телефоні * 2 і номер, на який хоче перекинути. Ну наприклад 1002

4) Номер 1002 бере трубку. «People1» запитує у номера 1002 чи хоче він розмовляти. Якщо хоче, то «People1» кладе трубку

5) Після того, як «People1» поклав трубку, дзвінок уже переходить до «people2» (1002). Якщо «people2» не відповість протягом 15 секунд або повісить трубку, нас знову відішлють до «people1».

3.3 Реалізація функції встановлення музики замість гудка

Для цього виконуємо наступні кроки:

Закидуємо наш аудіофайл в папку mymusic, яку ми попередньо створили в linux.

Тепер налаштовуємо конфігурації ASTERISK та редактуємо musiconhold.conf

Всередині файлу знаходимо наступне:

[default]

mode=files

directory=moh

Змінюємо значення directory на *directory=moh/mymusic*

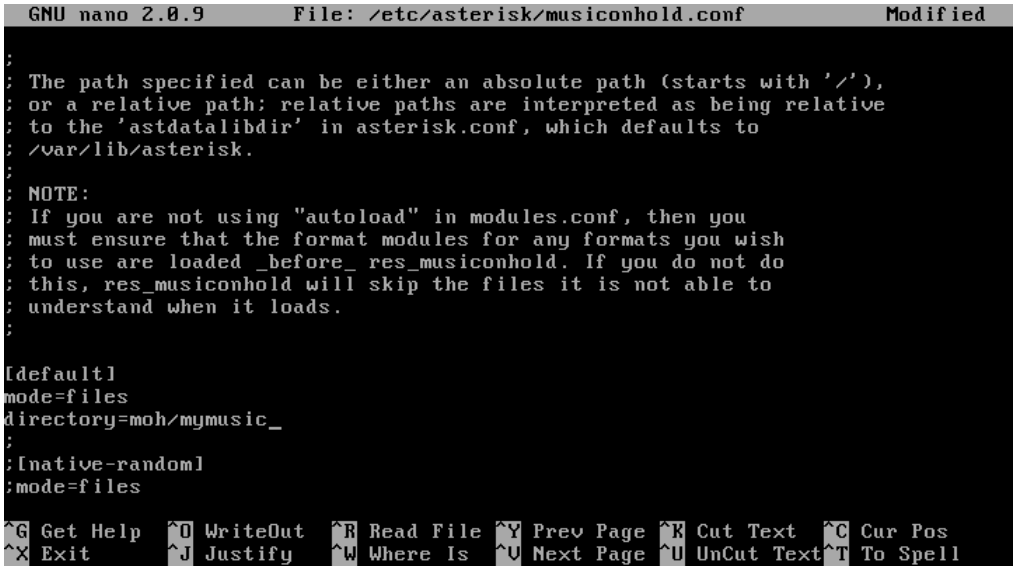

Рис. 3.7. Редагування musiconhold.conf

Та зберігаємо файл.

Якщо ми покладемо в папку mymusic й інші файли, вони будуть програватися по черзі, то один, то інший.

Далі налаштовуємо extensions.conf.

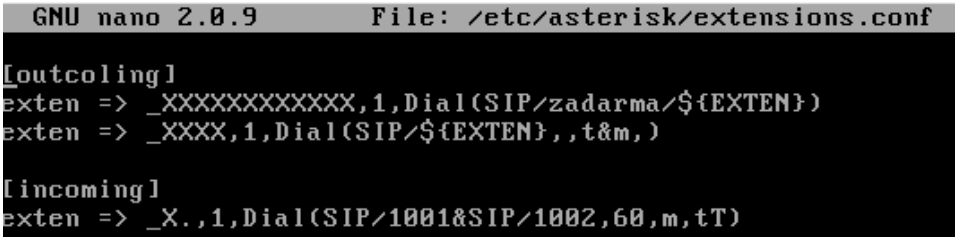

Рис. 3.8. Редагування extensions.conf

Тепер все готово й музика буде програватись, за це відповідає літера m.

3.4 Створення інтерактивного (голосового) меню

Для цього виконуємо наступні кроки:

Перш за все необхідно записати голосове повідомлення. Після закидуємо його в попередньо створену папку voicemail.

Потім створюємо новий номер, наприклад, 7777, до якого не буде підключено жодного телефону, але який буде використовуватись для того, щоб емітувати дзвінок ззовні. Для цього додаємо в sip.conf наступне:

Рис. 3.9. Редагування sip.conf

Тепер редактуємо extensions.conf.

```
[incoming]
exten => _X.,1,Goto(menu,s,1); Якщо нам хтось дзвонить,
то вхідний дзвінок з файлу sip.conf надходить на цей контекст.
Після чого дзвінок переадресовується за допомогою функції Goto на котекст menu
[outcoling]
exten => _XXXXXXXXXX,1,Dial(SIP/zadarma/${EXTEN})
exten => _XXXX,1,Dial(SIP/${EXTEN},,m)
exten => 7777,1,Goto(menu,s,1); Якщо ми ззовні зателефонуємо на цей номер,
то ми зможемо перевірити роботу нашого голосового меню
[menu]
exten => s,1,Background(/var/lib/asterisk/moh/voicemail/voicemenu); Тут ловиться
дзвінок з контексту incoming і програється записане нами вітання
exten => 1,1,Dial(SIP/1001,,m);якщо людина натиснула цифру 1, то дзвонимо до нашого
внутрішнього абонента 1001
exten => 2,1,Dial(SIP/1002,,m);якщо людина натиснула цифру 2, то дзвонимо до нашого
внутрішнього абонента 1002
exten => s,n,Wait(5); Якщо людина не натиснула нічого, чекаємо 5 секунд та
exten => s,n,Dial(SIP/1001&SIP/1002,,m) ;тоді дзвонимо обом абонентам
```

Рис. 3.10. Редагування extensions.conf

Все. Ми створили голосове меню.

3.5 Реалізація функції запису розмов

Для цього виконуємо наступні кроки:

Насамперед, необхідно визначити папку, куди будуть записуватися і зберігатися дзвінки. Вона носитиме назву callrecords.

Запис дзвінків налаштовується в extensions.conf

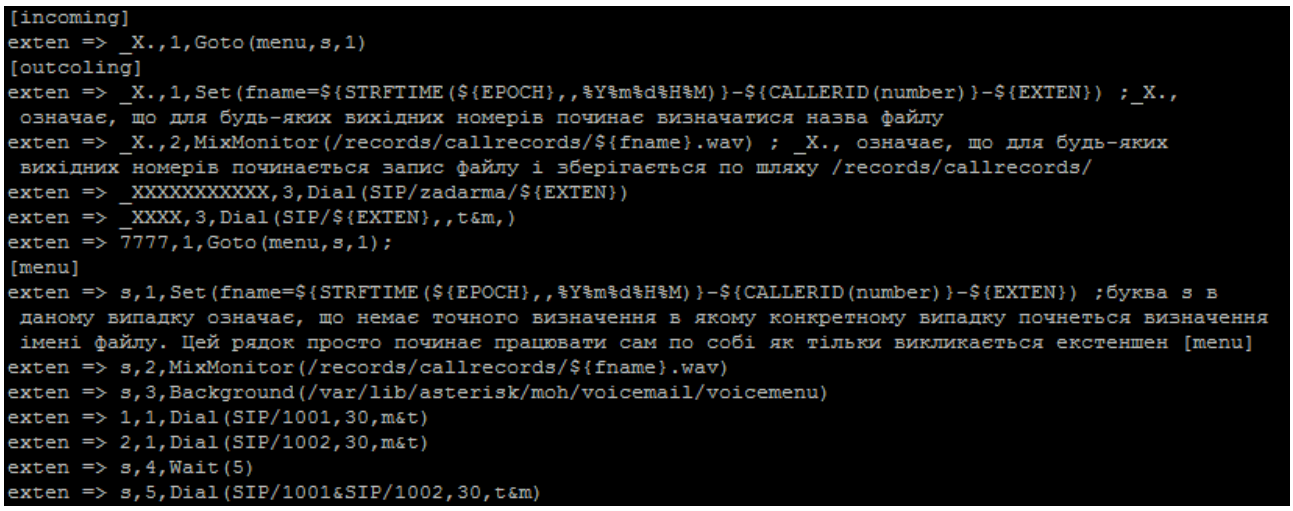

Рис. 3.11. Редагування extensions.conf

Тобто ми тут прописали запис розмов для 2-х випадків:

1) Коли ми дзвонимо (контекст outcoling);

2) І коли нам дзвонять (контекст menu). А контекст menu, в свою чергу викликається з контексту incoming.

Тепер нам необхідно прослухати ці розмови. Якщо ми хочемо прослухати ці розмови із Windows, то нам треба розшарити папку records, яку ми створили в Linux. Для того, щоб розшарити папку в Linux, необхідно встановити і налаштувати сервер Samba, який і буде керувати протоколом Samba. Для цього:

Для встановлення пишемо команду: yum install samba, а після встановлення редагуємо конфігураційний сервер Samba.

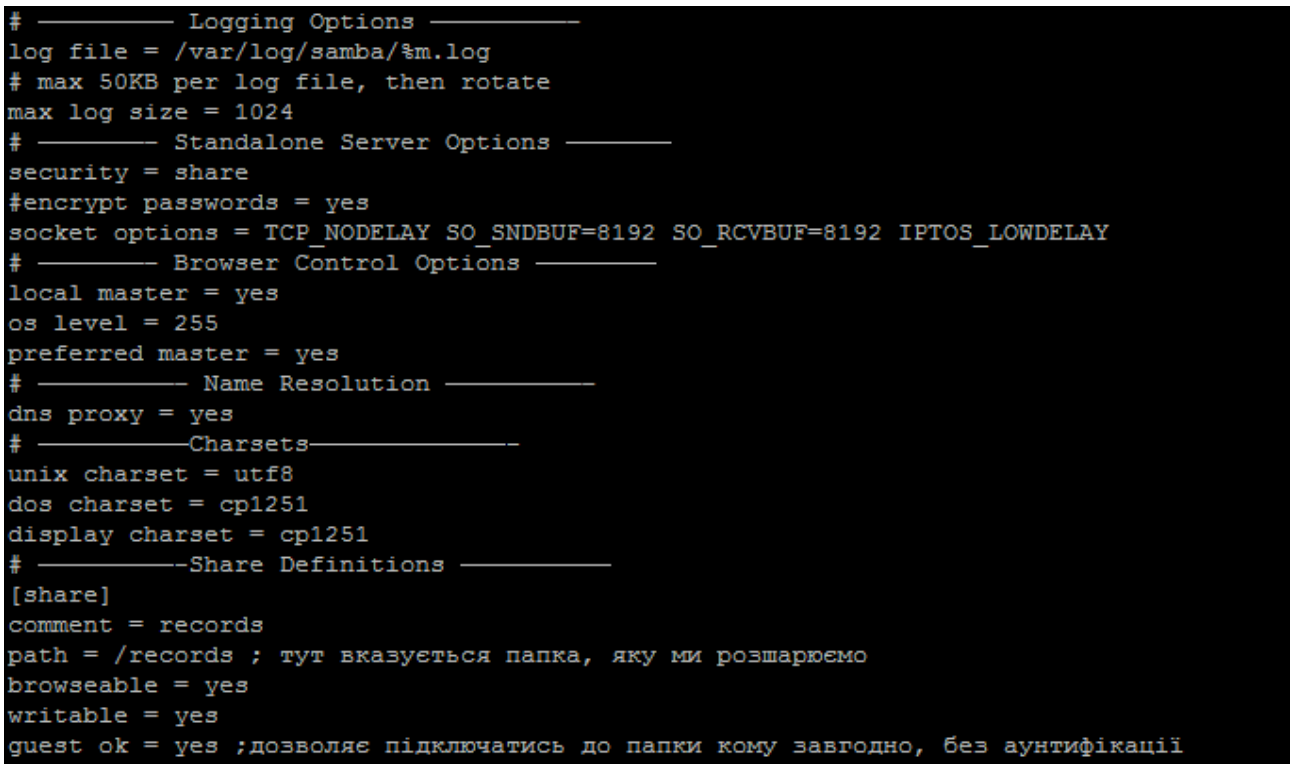

Рис. 3.12. Встановлення сервера Samba

Після цього стартуємо сервер. Прописуємо в консолі /etc/init.d/smb start та додаємо його а автозапуск chkconfig smb on.

Тепер в Windows запускаємо додаток "Виконати" і пишемо \\ip_нашого_Linux_сервера. Ip нашого Linux можна дізнатися набравши в Linux команду ifconfig. Усе! Тепер ми заходимо через Windows в нашу розшарену папку та бачимо там всі наші записані розмови в папці callrecords.

3.6 Реалізація функції автовідповідача

Для цього виконаємо наступне:

Створюємо папку з назвою voicemail, де будуть зберігатися записані файли автовідповідача, а також папку voicebox, для повідомлень-привітань автовідповідача.

Тепер налаштовуємо Dial план в файлі extensions.conf. Він буде виглядати таким чином:

```
[incoming]
exten => _X.,1,Goto(menu,s,1)

[outcoling]
exten => _X.,1,Set(fname=${STRFTIME(${EPOCH},,%Y%m%d%H%M)}-${CALLERID(number)}-${EXTEN})
exten => _X.,2,MixMonitor(/records/callrecords/${fname}.wav,b)
exten => _XXXXXXXXXXX,3,Dial(SIP/zadarma/${EXTEN})
exten => _XXXX,3,Dial(SIP/${EXTEN},,t&m,)
exten => 7777,3,Goto(menu,s,1,t&m)

[menu]
exten => s,1,Set(fname=${STRFTIME(${EPOCH},,%Y%m%d%H%M)}-${CALLERID(number)}-${EXTEN})
exten => s,2,MixMonitor(/records/callrecords/${fname}.wav)
exten => s,3,Background(/var/lib/asterisk/moh/voicemail/voicemenu)
exten => 1,1,Dial(SIP/1001,30,m&t)
exten => 1,2,Goto(autoanswer,s,1) ;Якщо 1001 не відповів або зкинув виклик, перенаправляємо
 на автовідповідач
exten => 2,1,Dial(SIP/1002,30,m&t)
exten => 2,2,Goto(autoanswer,s,1) ;Якщо 1002 не відповів або зкинув виклик, перенаправляємо
 на автовідповідач
exten => s,4,Wait(5)
exten => s,5,Dial(SIP/1001&SIP/1002,30,t&m) ;якщо напротязі 30 секунд не 1001, не 1002 не
 відповіли або зкинули виклик, то викликається autoanswer (автовідповідач)
exten => s,6,Goto(autoanswer,s,1)

[autoanswer]
exten => s,1,Background(/var/lib/asterisk/moh/voicebox/nazvafpryv) ;програється наше записане
 привітання, що всі зайняті
exten => s,2,Set(fname=${STRFTIME(${EPOCH},,%Y%m%d%H%M)}-${CALLERID(number)}-${EXTEN}) ;тут
 виконується визначення імені файла, в який буде записане повідомлення, залишающого повідомлення
 на автовідповідач
exten => s,3,Record(/records/voicemail/${fname}.wav,0,15,X) ;тепер записується сам файл
exten => s,4,Hangup
```

Рис. 3.13. Редагування extensions.conf

Екстеншен autotransfer, викликається в разі, якщо SIP/1001&SIP/1002 зайняті або не відповіли.

3.7 Реалізація функції перехвату викликів

Виконуємо наступне:

В ASTERISK існує спосіб перехопити дзвінок. Уявімо собі ситуацію, що ми знаходимося в одному відділі. І бачимо, що у нашого колеги задзвонив телефон, але його немає на місці.

Якщо ми на своєму телефоні натиснемо зірочку, то ми перехопимо цей дзвінок (переведемо його на свій телефон).

Для реалізації перейдемо до редагування файла sip.conf.

Для [1001] и [1002] допишемо наступне:

callgroup=2

pickupgroup=2

```
[1001]
type=friend
regexten=1001
secret=1234
context=outcoling
host=dynamic
callerid="1001" <1001>
disallow=all
allow=alaw
allow=ulaw
language=ru
callgroup=1
pickupgroup=1
qualify=yes
canreinvite=yes
call-limit=4
nat=no
callgroup=2
pickupgroup=2

[1002]
type=friend
host=dynamic
insecure=invite
username=1002
secret=45678
context=outcoling
disallow=all
allow=alaw
callgroup=2
pickupgroup=2
```

Рис. 3.14. Редагування sip.conf

Тобто тепер внутрішні номери (співробітники з номера) 1001 і 1002 перебувають в callgroup 2 та pickupgroup 2.

Наступним кроком буде редагування файлу features.conf.

Знаходимо рядок

;pickupexten = *8

Та розкоментуємо його (прибираємо на початку рядка крапку з комою) і значення з *8 змінюємо на *

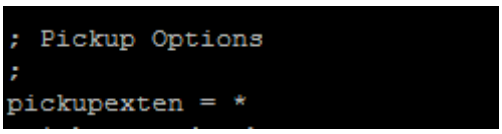

Рис. 3.15. Редагування features.conf

Зберігаємо файл и робимо core reload.

ВИСНОВОК ДО РОЗДІЛУ 3

В даному розділі реалізовано основні функції ASTERISK.

1. Було реалізовано конфігурацію ASTERISK на роботу через транк й тепер можна приймати дзвінки з зовні.
2. Також було реалізовано функцію встановлення музики замість гудка.
3. Створили голосове меню та реалізували функцію запису розмов.
4. Після чого були виконані кроки по налаштуванню автовідповідача й тепер, якщо жоден абонент не відповідає, той хто телефонує може почути, що всі абоненти зайняті й запропонувати залишити повідомлення, яке згодом можна буде прослухати.
5. Також реалізовано функції перенаправлення та перехоплення дзвінків.

ВИСНОВКИ

В даній дипломній роботі було розглянуто системи IP-PBX. Ми розібралися, що саме вважати за SIP АТС. Також зроблено огляд безкоштовних найбільш популярних IP PBX систем, які розповсюджуються в вихідному коді.

Розглянули причини чому ми обрали саме ASTERISK.

Було наведено перелік проблем, які вирішує ASTERISK, а також приведено його можливості. Представлено структуру системи й наведені функції ASTERISK: встановлення музики замість гудка, створення інтерактивного (голосового) меню, перенаправлення викликів, запис розмов, простий автовідповідач, встановлення системи перегляду статистики дзвінків, конференц-зв'язок, парковка викликів, переадресація дзвінків, черга дзвінків, робота ASTERISK в залежності від дня тижня і часу доби. Було розібрано, як ASTERISK співпрацює з VoIP та їх режими роботи. Також було наведено послідовність дій завдяки яким ми можемо встановити ASTERISK на наш ПК.

Було реалізовано конфігурацію ASTERISK на роботу через транк для того щоб можна було приймати дзвінки з зовні. Також було реалізовано функцію встановлення музики замість гудка. Створено голосове меню та реалізовано функцію запису розмов. Також реалізовано функції перенаправлення та перехоплення дзвінків.

Підводячи підсумок, можна сказати, що ASTERISK на даний момент є однією з найкращих IP-PBX систем. Можливості та функції ASTERISK продовжують розвиватися. Оскільки впровадження IP-PBX просте й дешеве, а також дозволяє об'єднати кілька географічно розподілених офісів в єдину систему телефонії, забезпечити безкоштовним зв'язком надомних і мобільних працівників, тому я вважаю, що в майбутньому ASTERISK набере ще більшого поширення серед працівників малих й великих компаній.

Також були розроблені лабараторні роботи для студентів, які містяться в додатках.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Огляд вільно доступних і безкоштовних IP АТС [Електронний ресурс]. – Режим доступу до ресурсу: https://habrahabr.ru/post/122215.
2. What is a PBX Phone System? [Електронний ресурс]. – Режим доступу до ресурсу: https://www.3cx.com/pbx/pbx-phone-system/.
3. *Ніконоров М.* Телефонія Asterisk з нуля / М. Ю. Ніконоров, А. В. Єфременко. – 2014. – 131 с.
4. Understanding H.323 Gatekeepers [Електронний ресурс]. — Режим доступу до ресурсу: http://www.cisco.com/c/en/us/support/docs/voice/h323/5244-understand-gatekeepers.html.
5. FreeSWITCH vs Asterisk [Електронний ресурс]. – Режим доступу до ресурсу: https://www.whichvoip.com/freeswitch-vs-asterisk.htm.
6. https://uk.wikipedia.org/wiki/VoIP
7. https://studfile.net/preview/7835384
8. https://uk.wikipedia.org/wiki/CentOS
9. https://forum.asterisk.ru/viewtopic.php?f=5&t=15932
10. https://forum.asterisk.ru/viewtopic.php?p=79304
11. https://ukrbukva.net/64595-Postroenie-telefonnoiy-seti-malogo-predpriyatiya-na-programmnoiy-ATS-Asterisc.html
12. https://docplayer.net/82804109-Osnovi-infokomunikaciynih-tehnologiy.html

ДОДАТОК 1
ЛАБАРАТОРНА РОБОТА
ВСТАНОВЛЕННЯ І НАЛАШТУВАННЯ CENTOS 6.5

Мета: навчитися встановлювати і налаштовувати CentOS 6.5, ознайомитись з інтерфейсом системи.

Основні теоретичні відомості

CentOS (Community ENTerprise Operating System) — вільно доступний дистрибутив Лінукс на основі комерційного дистрибутивуRed Hat Enterprise Linux компанії Red Hat. Проект доклав зусиль, щоб зібрати 100% двійково сумісний зі своїм предком дистрибутив у плані шляху його розвитку, і старається розвиватися з ним в один бік, не відступаючи від його основної лінії та модифікацій, щоб не змінювати своєї мети.

Red Hat Enterprise Linux складений переважно з вільного тавідкритого програмного забезпечення, але наявний у доступній для вживання, двійковій формі (наприклад, на CD або DVD дисках) лише для передплатних користувачів. Як і вимагається, Red Hat випускає усі сирцеві тексти своїх продуктів під GNU General Public License та іншими вільними ліцензіями. Розробники CentOS використовують цей сирцевий код для створення кінцевого продукту, котрий є дуже подібним до Red Hat Enterprise Linux і вільним для завантаження та використання, однак без відповідної технічної підтримки з боку компанії Red Hat. Існують й інші дистрибутиви, що базуються на сирцевих текстах Red Hat Enterprise Linux, однак жоден з них не досяг такого рівня спільноти; загалом CentOS — єдиний дистрибутив, котрий йде у ногу зі змінами, що вносяться до Red Hat Enterprise Linux.

CentOS віддав перевагу програмному забезпеченню для оновлення на основі yum, хоча підтримка up2date також присутня. Можна використовувати для звантаження та встановлення як додаткові пакунки і залежності, так і спеціальні та періодичні оновлення безпеки з репозиторію на CentOS Mirror Network.

CentOS придатний для використання базованих на X Windowстільницях, однак більш звично використовувати його, як серверну операційну систему для вебхостингу. Багато великих хостингових компаній використовують CentOS разом із cPanel Control Panel для забезпечення стабільної роботи для своїх веб-застосунків.

Нумерація версій

Номер версії CentOS складається з двох частин, головної (major) та незначної (minor). Головна версія рівна відповідній версії Red Hat Enterprise Linux, з якої беруться джерельні пакунки для побудови CentOS. Незначна версія відповідає номеру випуску модифікацій для Red Hat Enterprise Linux, з якого беруться пакунки для побудови відповідної версії CentOS. Наприклад, CentOS 5.3 (актуальна нині) базується на Red Hat Enterprise Linux 5 update 3.

Підтримувані архітектури

CentOS підтримує усі архітектури, підтримку яких має Red Hat Enterprise Linux, а також додатково Alpha та SPARC.Intel x86-сумісні (32 бітна)

- Intel Itanium (64 бітна)
- Advanced Micro Devices AMD64 та Intel EM64T (64 бітна)
- PowerPC/32 (Apple Macintosh PowerMac працює на процесорах PowerPC G3 або G4) (у режимі тестування)
- IBM Mainframe (eServer zSeries та S/390)

Обладнання, що необхідне для виконання роботи: сервер, дистрибутив CentOS, клавіатура, монітор, патч-корд.

Заходи безпеки при виконанні роботи: під час виконання роботи студент повинен чітко виконувати вимоги викладача, не використовувати компю’тер в цілях не пов’язаних з учбовим процесом.

Порядок виконання роботи

Запишіть ISO образ дистрибутива CentOS 6.5 на CD диск або на завантажувальний USB диск, у віртуальну машину Oracle VM VirtualBox CentOS 6.5 сервер можна встановлювати безпосередньо з ISO образу дистрибутива. В BIOS комп'ютера визначте порядок завантаження з диска CentOS.

- Початок завантаження CentOS 6.5 server

Рис. Д1.1. Встановлення CentOS

Перевірку дистрибутива CentOS 6.5 можна пропустити:

Рис. Д1.2. Перевірка дистрибутива CentOS

Рис. Д1.3. Логотип CentOS 6 Community ENTerprise Operating System

- Виберіть мову на час процесу установки(вибираємо англійську)
- Виберіть розкладку клавіатури: (вибираємо англійську)
- Виберіть тип жорсткого диска для установки CentOS 6.5 server:

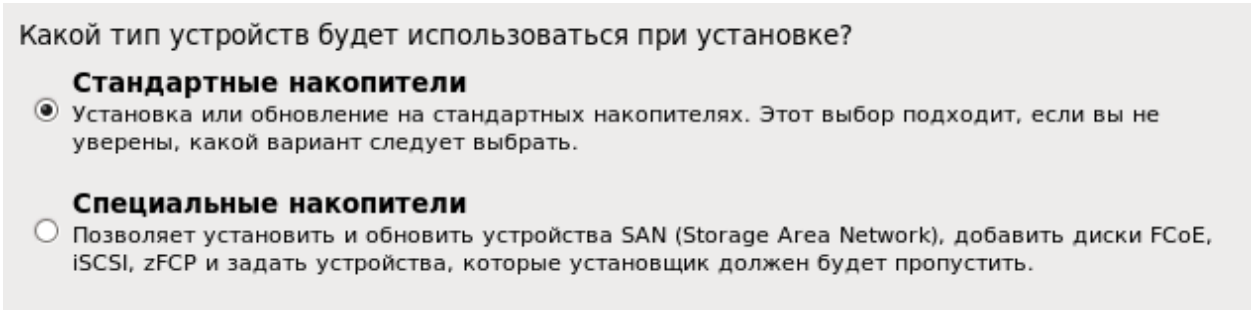

Рис. Д1.4. Встановлення CentOS

- Введіть пароль для облікового запису користувача root:

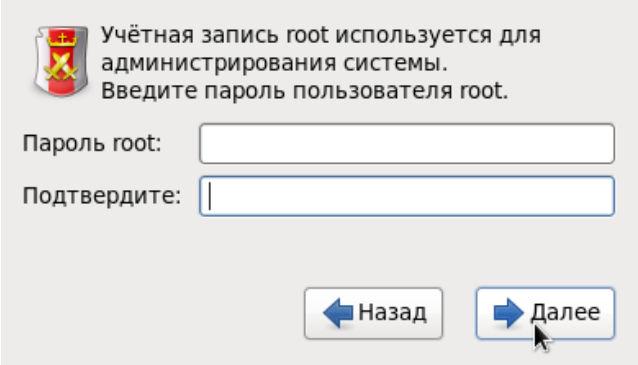

Рис. Д1.5. Встановлення CentOS

- Визначте як використовувати жорсткий диск:

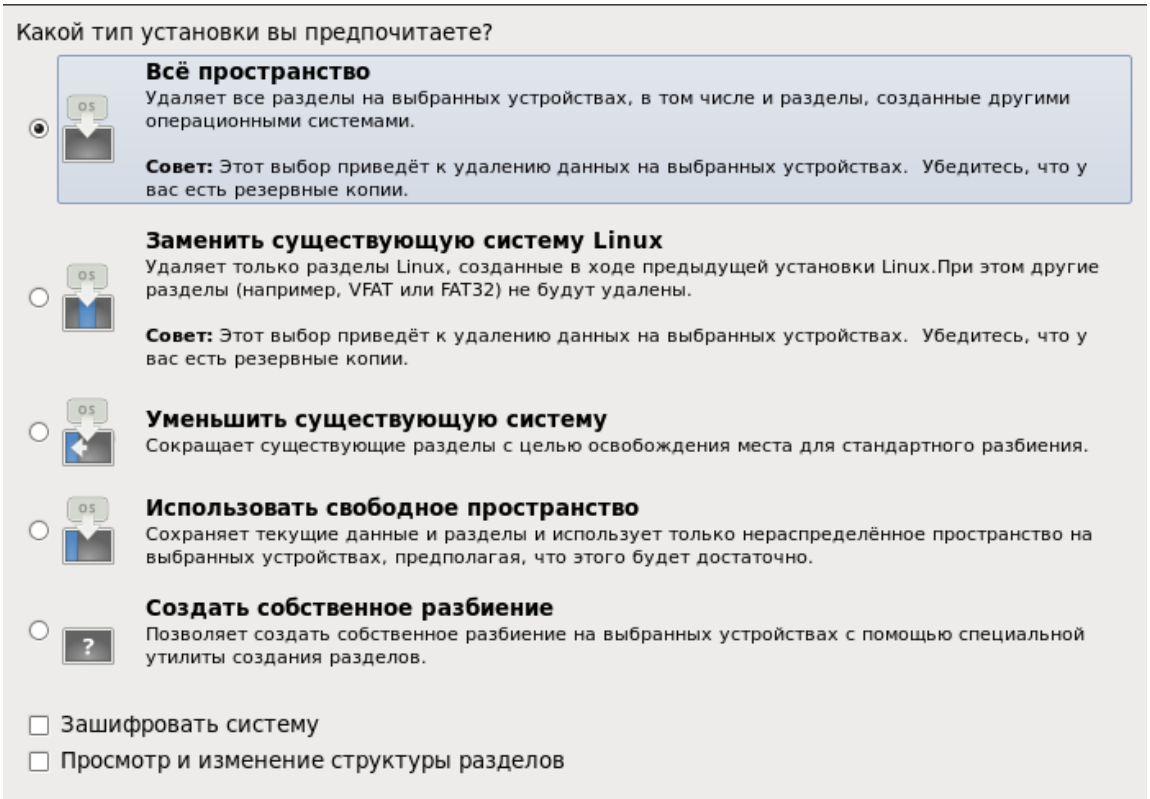

Рис. Д1.6. Встановлення CentOS

- Збережіть всі раніше введені параметри для CentOS 6.5 на жорсткому диску:
-

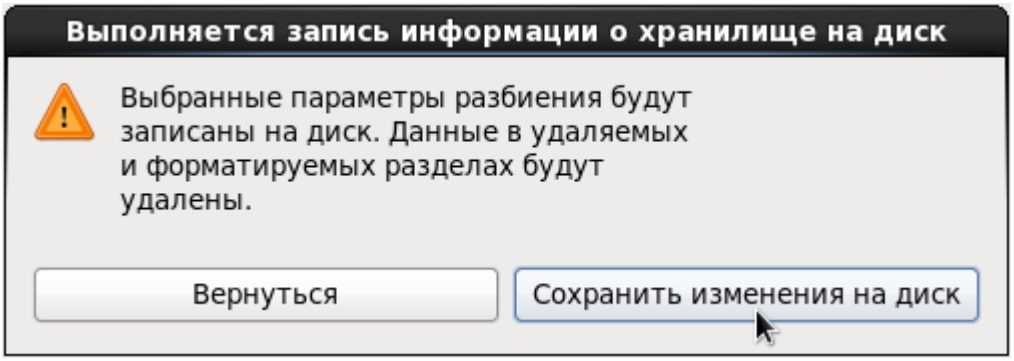

Рис. Д1.7. Встановлення CentOS

- Установка пакетів CentOS 6.5 сервера:

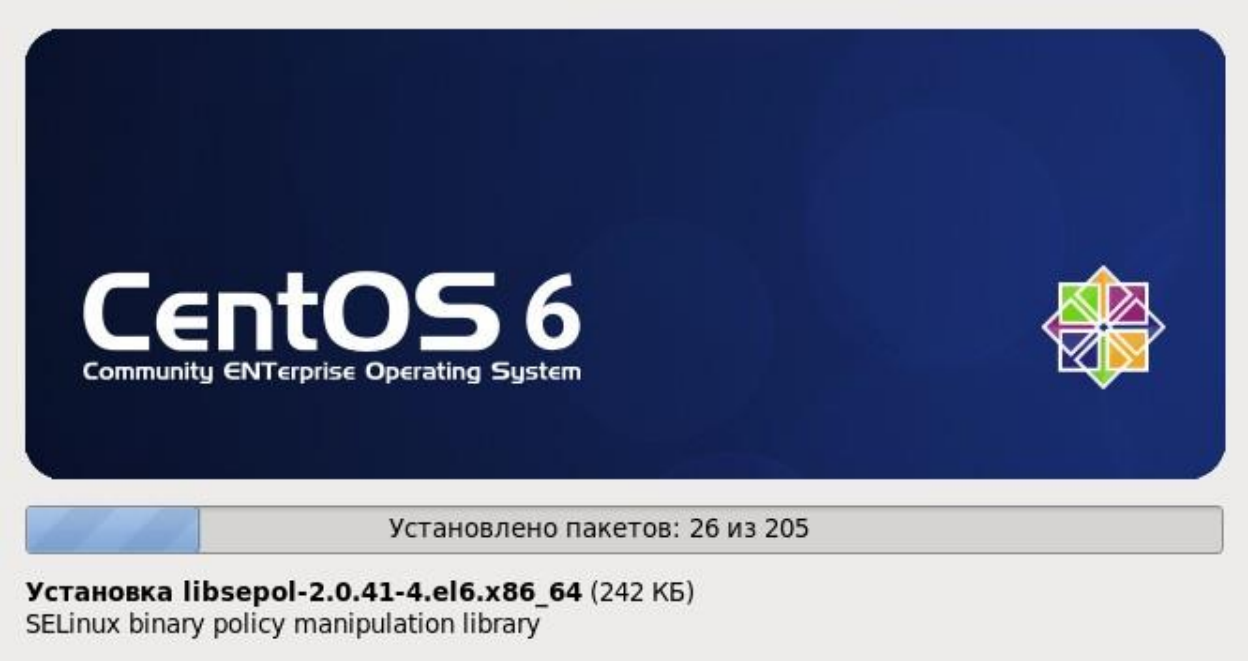

Рис. Д1.8. Встановлення покетів CentOS

- Наприкінці установки CentOS 6.5 server потрібно перезавантажити комп'ютер:

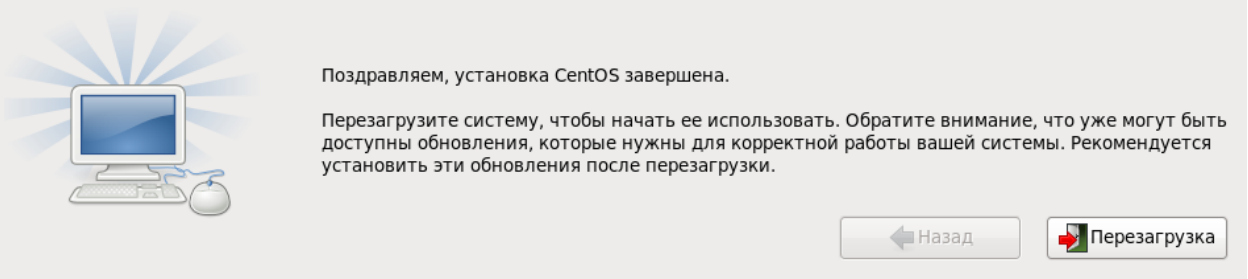

Рис. Д1.9. Встановлення CentOS

- Вийміть диск з дистрибутивом CentOS 6.5 і перезавантажте комп'ютер.

Рис. Д1.10. Встановлення CentOS

- CentOS 6.5 сервер готовий до роботи:

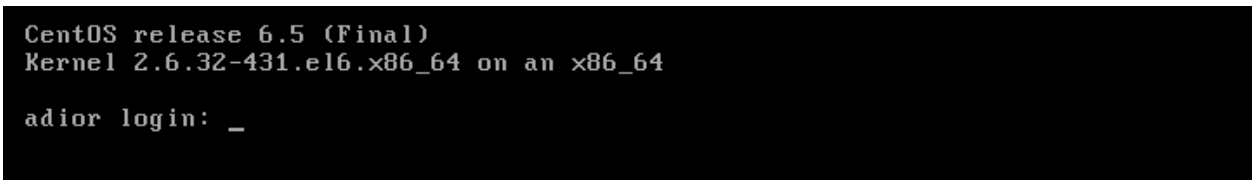

Рис. Д1.11. Встановлення CentOS

Переходимо до налаштування мережі інтернет на нашому сервері.

В консолі прописуємо: **cd /etc/sysconfig/network-scripts/,** потім відкриваємо за допомогою редактора vi даний файл ifcfg-eth0 командою **vi ifcfg-eth0**

Тепер пишемо команду:

ifconfig -a (показує мережеві адаптери)

Якщо ОС бачить мережевий адаптер, то він відобразиться. У мене він називається eth0.

Якщо немає IP-адреси, то для того щоб ОС прикріпила його до мережевої карти пишему команду:

ifup eth0 (визначає ip інтерфейсу eth0)

Знову пишемо команду **ifconfig -a** і бачимо свою ip адресу для інтерфейсу eth0.

Увага — після перезавантаження доведеться знову визначати ip адресу.

Для того, щоб ip адрес чіплявся автоматично при старті CentOS, ми виконуємо наступні дії:

а) встановлюємо текстовий редактор nano

yum install nano

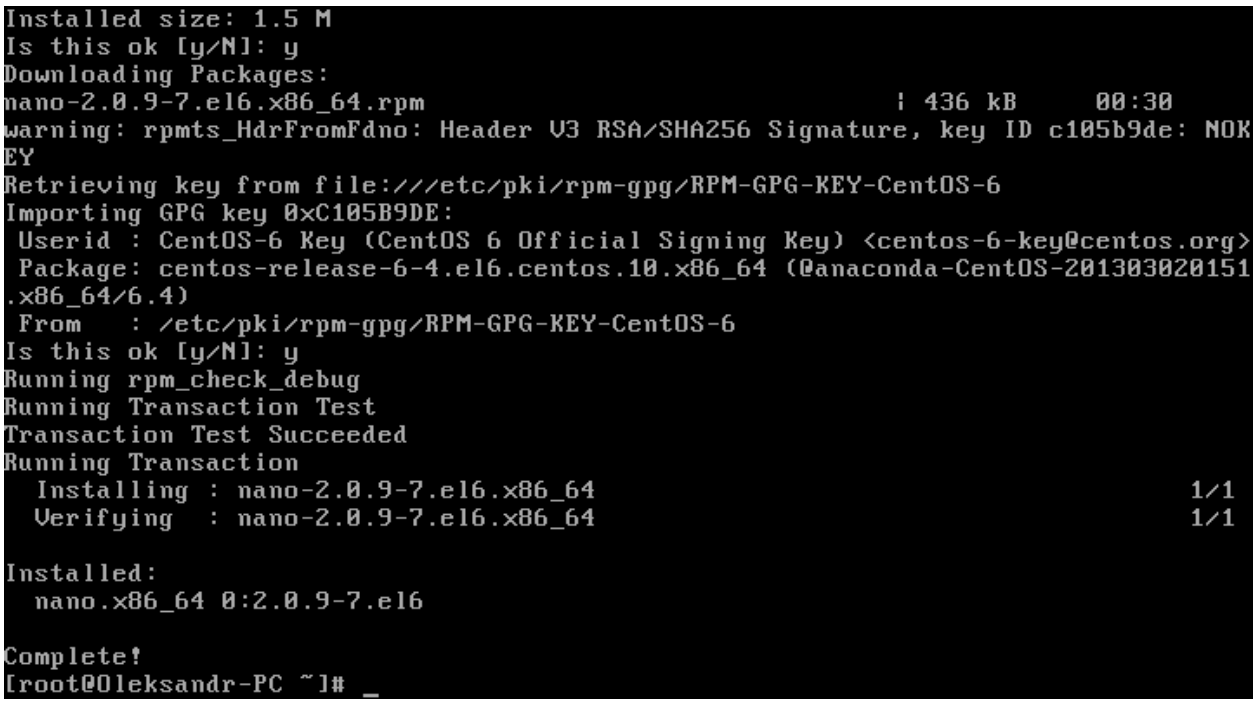

Рис. Д1.12. Встановлення текстового редактора

б) після пишемо команду:

nano /etc/sysconfig/network-scripts/ifcfg-eth0

і в файлі змінну ONBOOT = "no" змінюємо на ONBOOT = "yes".

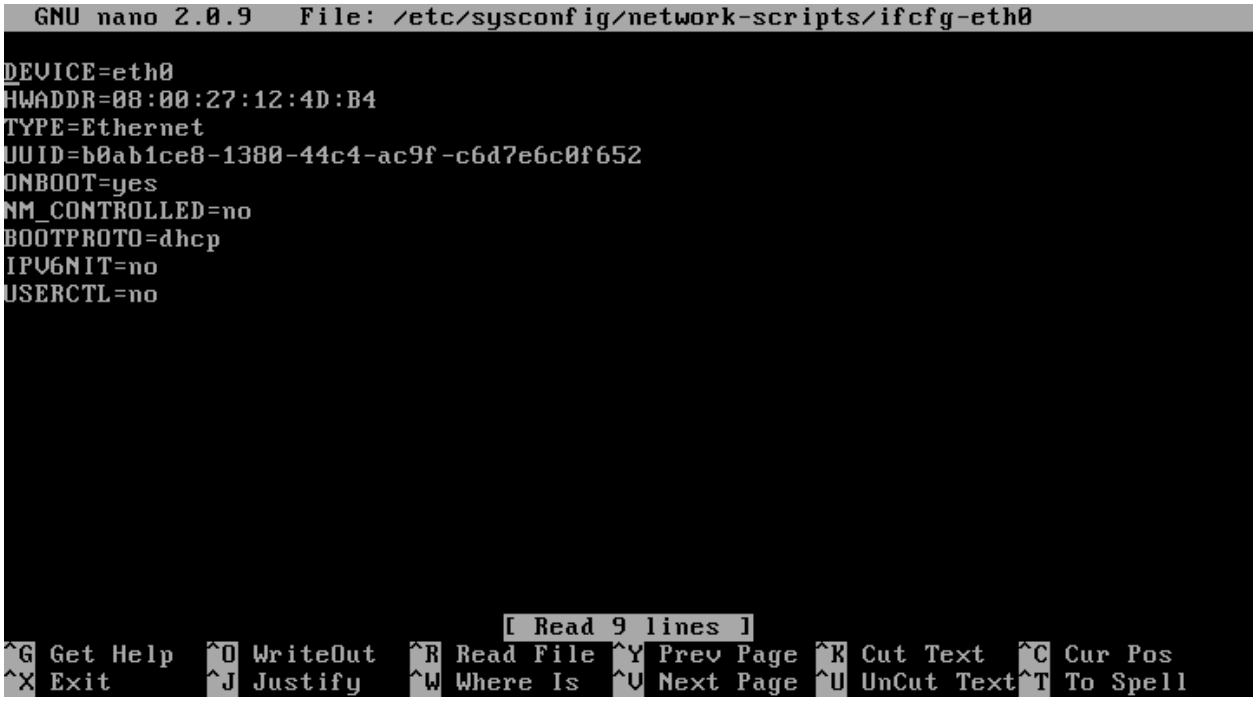

Рис. Д1.13. Налаштування CentOS

Таким чином ip адреса буде чіплятися автоматично при старті CentOS.

Контрольні питання

1) На основі якого дистрибутиві Лінекс був розроблений CentOS
2) Принцип нумерації версій.
3) Які архітектури підтримує?
4) Якому CentOS віддав перевагу програмному забезпеченню для оновлення?
5) Яке ім’я має адміністратор системи?

ДОДАТОК 2
ЛАБАРАТОРНА РОБОТА
БАЗОВЕ НАЛАШТУВАННЯ ASTERISK

Мета: навчитися налаштовувати телефонію на базі ASTERISK, дзвінок в середині мережі.

Основні теоретичні відомості

Революція в телефонії

Для досягнення мети не потрібно мати підтримкубільшості, достатньо кількох шалених і невтомних борців, здатних розпалити полум'я в умах людей.

- Самюель Адамс

Ми є свідками неймовірних революційних подій. Вони очікувалися вже давно, і тепер, коли процес почався, ніщо не в силах зупинити його. Зміни охопили технологічну область, яка сильно відстала від всіх інших галузей промисловості, об'єднаних загальною назвою hi-tech (від англ. high technology -висока технологія). Йдеться про телекомунікації, революцію в яких здійснив продукт з відкритим вихідним кодом для офісної телефонної станції з виходом в загальну мережу (Private Branch e-Xchange, PBX) під назвою ASTERISK™. Телекомунікації — це, напевно, єдина з високотехнологічних галузей, якої не торкнулася революція, пов'язана з появленням відкритого вихідного кода1. Основні виробники в цій області як і раніше створюють необґрунтовано дорогі, несумісні один з одним системи, які використовують архаїчне і заплутане програмне забезпечення і вражаюче своєї інженерної думкою, але безнадійно застаріле обладнання.

Наприклад, Business Communications Manager від компанії Nortel ка кім- то чудом об'єднує в собі кнопковий номеронабиратель 15 — років — ній давності і ПК на базі процесора Celeron з частотою 1,2 ГГц. всі це може стати вашим всього за $5000 — 15 000, не включаючи вартість телефонних апаратів. Якщо хочеться отримати якісь дійсними — але цікаві функції, доведеться доплатити за універсальні додатки з обмеженою функціональністю і закритим вихідним коном. Конфігурація?

Забудьте про неї — вона не входить у функціонал системи. Технології майбутнього і сумісність зі стандартами? Зачекайте пару років — над цим працюють. Всі основні виробники засобів зв'язку пропонують подібні продукти. Виробники не хочуть забезпечити вам можливість вибору або гнучкість, а зацікавлені в тому, щоб споживач був обмежений рамками жорстко фіксованою функціональності їх продуктів. Система ASTERISK вносить докорінні зміни. З ASTERISK ніхто не може диктувати, як повинна працювати телефонна система або яка технологія повинна використовуватися. Вибирайте будь-яку. ASTERISK твердо слід ідеї сумісності зі стандартами, дозволяючи при цьому насолоджуєшся свободою створення власних нововведень. Вибір тільки за вами, ASTERISK не накладає ніяких обмежень. Однак за таку неймовірну гнучкість доводиться платити: ASTERISK не назвеш системою, яку легко конфігурувати. І не тому, що вона нелогічна, заплутана або незрозуміла; навпаки, вона дуже розумно сконструйована і зручна в застосуванні. У людини, вперше який побачив діалплан (робоче середовище) ASTERISK і початківця розуміти його можливості, просто загоряються очі. Але коли є буквально тисячі способів досягнення результату, природно, процес потребує додаткових зусиль. Напевно, це можна порівняти з будівництвом будинку: всі компоненти окремо прості і зрозумілі, але людині, щоб виконати такий проект, доведеться або a) звернутися за допомогою до фахівців, або b) розвинути у себе необхідні навики за допомогою навчання, практики і гарного довідника по даній теми.

VoIP: наведення мостів між традиційною і мережевої телефонією.

VoIP (англ. voice over IP) — технологія передачі медіа даних в реальному часі за допомогою сімейства протоколів TCP/IP. IP-телефонія — система зв'язку, при якій аналоговий звуковий сигнал від одного абонента дискретизується (кодується в цифровий вигляд), компресія і пересилається по цифрових каналах зв'язку до другого абонента, де проводиться зворотна операція — декомпресія, декодування і відтворення аналогового сигналу.

Протоколи

Частина протоколів із сімейства VoIP затверджується Інтернет співтовариством в якості RFC (англ. request for comments), частина — міжнародними організаціями (IETF тощо).

Основу технології VoIP складає протокол RTP (real time protocol, RFC 1889, RFC 3550), побудований поверх протоколів UDP/IP, а також протоколи (методи) кодування медіа даних (для кодування голосу це протоколи G.711, G.723, G.729, GSM, Speex та інші, для кодування відео це протоколи RFC ???).Існують розширення (профілі) протоколу RTP, такі як SRTP (secure RTP) та інші (RFC 1890, RFC 2198, RFC 3711 тощо).

Хоча передача голосу по IP-протоколу (Voice over IP, VoIP) часто розглядається як свого роду безкоштовна міжміський телефонний зв'язок, справжня цінність VoIP в тому, що з його допомогою голос становиться всього лише звичайним додатком у мережі передачі даних. Здається, ми забули про те, що призначення телефону — дозволити людям спілкуватися. Це проста мета насправді, і ми повинні мати можливість реалізовувати її набагато більш гнучко і творчо, ніж це пропонується зараз. Оскільки галузь продемонструвала небажання прагнути до цієї мети, рішенням задачі зайнялися ентузіасти. Складність полягає в тому, що галузь, яка практично не змінилася за останні сто років, не проявляє особливого інтересу до цього і зараз.

Для широкомасштабних змін необхідна гнучка технологія. Найуспішніша в світі мала АТС має конструктивне обмеження, про усунення якого користувачі благають ось вже протягом 15 років: при визначенні того, скільки разів продзвонить телефон, перш ніж виклик буде перенаправлено на голосову пошту, надається можливість вибрати 2, 3, 4, 6 або 10 дзвінків. Чи знаєте ви, скільки людина просили про внесення можливості вибору п'яти дзвінків? Здавалося б, потрібно внести проста зміна, але, скільки б не просили користувачі, виробники не можуть зрозуміти, що це дійсно є проблемою.»Вона так працює, — відповідають вони, — і користувачам треба просто змиритися з цим». Інший подібний приклад: ім'я в телефонній книзі може бути довжиною не більше семи сімволов2. Наприкінці 1980 -х, коли ця система розроблялася, оперативна пам'ять була дуже дорогою і зберігання семи символів

для десятків телефонних апаратів означало гігантські витрати на обладнання. А яке цьому виправдання може бути сьогодні? Його немає. Чи планується змінити ситуацію? Навряд чи, питання навіть не визнаний проблемою офіційно. Це лише два приклади, а галузь рясніє ними. Ми розглянули одну систему, але реальний стан справ такий, що недоліки є у всіх існуючих офісних АТС. Неважливо, наскільки багату функціональність пропонує телефонна станція, врахувати всі і передбачати винахідливість користувача неможливо. Декільком користувачам може знадобитися маленька незвичайна можливість, про яку група розробки або не подумала або вирішила не займатися нею через невиправданість витрат на її розробку, а оскільки код системи закритий, користувачі не зможуть самостійно реалізувати необхідну функціональність. Якби всякого роду правила та комерційні інтереси стримували розвиток Інтернету, він ніколи не отримав би такого широкого розповсюдження. Відкритість Інтернету означає, що кожен бажаючий може взяти участь у його розробці. У результаті спільного праці десятків тисяч умів отриманий продукт, який не міг би вийти зі стін жодної корпорації. Як і для багатьох проектів з відкритим вихідним кодом, таких як Linux та Інтернет, імпульсом до розробки ASTERISK були мрії тих, хто знав, що має існувати щось більше, ніж пропонується в даній галузі. Сила спільноти в тому, що його становлять не службовці, вирішальні поставлені перед ними конкретні завдання, а люди зі всіляких областей діяльності з абсолютно різним досвідом і різним розумінням гнучкості та відкритості. Ці люди знали: якщо зуміти виділити найкраще, що є в різних АТС, в окремі компоненти, які можна різним чином з'єднувати між собою, подібно блокам LEGO, почнуть з'являтися ідеї, які не пройшли б традиційний в корпораціях процес аналізу ризиків. до тих пір поки ні в кого немає повної картини того, як усе має виглядіти, нестачі в думках і ідеях нет1. Багато людей, вперше стикаючись з ASTERISK, вважають її незакінченою проектом. Напевно, їх можна порівняти з відвідувачами ізостудії, які очікували побачити тут підписані та пронумеровані репродукції. Часто вони розчаровуються, дізнавшись, що ASTERISK – це які очікують їх чисте полотно, тюбики з фарбою і нові кісті2. Навіть на цьому ранньому етапі, на якому вже вдалося досягти успіху, проектом ASTERISK займаються більше майстрів, ніж будь-який інший офісної АТС. У більшості компа-

ній- виробників над какимлибо продуктом трудяться лише кілька розробників ; у розробці ASTERISK беруть участь дуже багато людей. Для обслуговування більшості комерційних АТС у всьому світі знайдеться лише кілька десятків справжніх експертів ; у випадку з ASTERISK їх сотні. Глибина і широта експертних знань, вкладених у цей продукт, що не має аналогів в телефонній галузі. ASTERISK має віддаених шанувальників серед» старих» з Telco, колишніх свідками розквіту телефонних апаратів з дисковими номеронабирачами, співробітників великих телекомунікаційних компаній, які пам'ятають часи, коли голосова пошта була наймоднішою новітньою технологією, і фахівців з передачі даних, що допомагали створювати Інтернет. Всі ці люди вірять в одне — телекомунікаційна промисловість потребує належних революційних ізмененіях1. ASTERISK — це каталізатор.

ASTERISK: офісна АТС, створювана хакерами

Телекомунікаційні компанії, які вирішили ігнорувати ASTERISK, надходять ризиковано. Надана Aste ASTERISK risk гнучкість забезпечує можливості, про які кращі комерційні системи можуть тільки мріяти, бо ASTERISK — це АТС, створена хакерами. Якщо вас попросять не застосовувати слово» хакер», не звертайте уваги. Цей термін не є власністю засобів масової інформації. Вони привласнили його і спотворили значення, називаючи так»зловмисних зломщиків». Прийшов час відновити справедливість. Хакери розробили механізм передачі даних по мережі, тобто Інтернет. Хакери створили Apple Macintosh і операційну систему UNIX. Хакери працюють і над телефонною системою майбутнього. Не треба лякатися, всі вони відмінні хлопці і зможуть побудувати систему, набагато більш безпечну, ніж всі існуючі сьогодні. Вони не стануть створювати ненадійні і легко піддаються злому засоби безпеки для закритих систем, хакери зможуть швидко реагувати на зміну тенденцій у забезпеченні безпеки і налаштовувати телефонну систему відповідно політиці корпорації і найкращій практиці галузі. Як і інші системи з відкритим вихідним кодом, As ASTERISK terisk зможе розвинутися в набагато більш безпечну платформу, ніж будь-яка комерційна система, не всупереч своїм хакерським корінням, а, швидше, завдяки їм.

ASTERISK: офісна АТС, створювана професіоналами

Ніколи за всю історію телекомунікацій не існувало системи, настільки відповідає потребам бізнесу в будь-якій ціновій категорії. ASTERISK — технологія, що надає нові можливості, і, як це було з Linux, скоро навряд чи можна буде знайти підприємство, на де не використовувалась б одна з версій ASTERISK, хоча б частково, десь в мережі, для вирішення проблем, які здатна вирішити тільки ASTERISK. Однак схоже, що це визнання відбудеться швидше, ніж було у випадку з Linux, з ряду причин: Linux вже проклав шлях до визнання відкритого вихідного коду. ASTERISK йде второваною дорогою. Телефонія знаходиться в тяжкому становищі, жоден з великих гравців на цьому ринку не є лідером. ASTERISK ж видається переконливим, реалістичним і вражаючим проектом. Кінцеві користувачі вже ситі по горло несумісними системами з обмеженою функціональністю і жахливої підтримкою. ASTERISK вирішує перші дві проблеми — підприємці та спільнота забезпечать останнє.

Файли extensions.conf, sip.conf

Конфігурація плану набору міститься у файлі конфігурації ASTERISK — extensions.conf. Знаходиться за адресою /etc/asterisk/extensions.conf. Це один з найважливіших конфігураційних файлів. У ньому визначається обробка та маршрутизація вхідних і вихідних дзвінків. Цей файл керує поведінкою всіх з'єднань які проходять через Вашу АТС. Зміст файлу»extensions.conf" розбито на секції, в яких можуть бути або визначені статичні настройки і визначення або виконувані команди плану набору, в цьому випадку вони називаються контекстами. Секції, призначені для статичних налаштувань, називаються general і globals, а імена контекстів визначаються системним адміністратором системи. Спеціальний тип контекстів — це макрос, позначений користувальницький певним ім'ям з префіксом macro -. Це багато разів виконувані шаблони, подібні процедурам в мові програмування. Кожна секція у файлі extensions.conf починається з рядка з ім'ям секції, укладеного в квадратні дужки. Це робить файл extensions.conf, за форматом, схожим за на традиційні. Ini файли в системі Windows.

У файлі конфігурації sip.conf в секції [general] додайте визначення register : формат:

register => user [: secret [: authuser]] @ host [: port] [/ extension] приклад :

; Зареєструвати 2345 у sip провайдера, як номер 1234 на нашому боці.

register => 2345 : password@mysipprovider.com / 1234

user — ідентифікатор користувача, який використовується для SIP сервера (наприклад, 2345)

authuser — не обов'язкове ім'я користувача для авторизації на SIP сервері

secret — пароль користувача

host — ім'я домену або хоста SIP сервера. Цей SIP сервер повинен бути визначений в своїй секції файлу sip.conf, де повинні бути задані його параметри (mysipprovider.com).

port — на який номер порту посилати запити на реєстрацію на сервері host. За замовчуванням — 5060

/ 1234 — номер екстеншена для прийому викликів у Вашому ASTERISK. 1234 — вставляється в SIP заголовок contact, SIP запиту на реєстрацію. Цей екстеншен використовується віддаленим SIP сервером, коли йому необхідно здійснити виклик в сторону Вашого ASTERISK. Дивись приклади, наведені нижче. За замовчуванням, використовується контекстний»s". Це, звичайно, все добре, але використання незашифрованих паролів в текстовому файлі — не найвдаліша ідея, але що ж можна ще зробити тепер. Вам необхідно реєструватися, тільки якщо : a) повинна бути можливість зателефонувати до Вас, і b) одна зі сторін має динамічний IP адресу. Перевірити, чи вдало зареєструвався Ваш сервер, можна за допомогою CLI команди:» SIP SHOW REGISTRY», аналогічно, можна отримати список клієнтів, зареєстрованих на Вашому сервері, за допомогою команди:» SIP SHOW PEERS». Ви можете переглянути більш детальну інформацію про зареєстроване клієнті, за допомогою команди:» SIP SHOW PEER <NAME>«. Виконайте команду» HELP SIP» в CLI консолі, щоб отримати список додаткових команд.

Визначення сервера, для здійснення вихідних дзвінків, має бути приблизно таким :

[mysipprovider — out]

type = peer

secret = password

username = 2345

host = sipserver.mysipprovider.com

fromuser = 2345

fromdomain = fwd.pulver.com

nat = yes

context = from — mysipprovider ; цей контест повинен бути визначений в extensions.conf

У файлі extensions.conf, для здійснення вихідних дзвінків, у Вас повинне бути присутнім правило набору, приблизно такого вигляду:

exten => _9. , 1, Dial (SIP / $ { EXTEN : 1 } @ mysipprovider — out, 30, r)

Зверніть увагу, що конструкція $ { EXTEN : 1 } витягує весь вміст змінної, в якій міститься викликається екстеншен (який співпав з шаблоном), за винятком першої цифри, в даному випадку : 9 + набір цифр. Зверніться до розділу по роботі з підрядками в описі змінних ASTERISK, для більш докладної інформації.

Далі приводитися секція (файлу extensions.conf), яка приймає виклики від sip провайдера і направляє його в потрібне Вам місце:

[from — mysipprovider]

exten => 1234,1, Answer ; 1234 — екстеншен з контактної інформації, за замовчуванням -» s»

exten => 1234,2, Dial (SIP/111, 25, Ttr) ; вхідний дзвінок перенаправляємо на SIP телефон з номером 111

exten => 1234,3, Hangup

Конфігурація SIP — секція general

Секція [general], файлу sip.conf, включає в себе наступні змінні:

allow = <codec> : Дозволені кодеки, порядок вибору кодека, задається порядком їх опису в цій команді (Спочатку використовуйте: DISALLOW = ALL, перед тим, як дозволити якісь конкретні кодеки)

allowtransfer ? = yes | no : З'явилася, починаючи з версії 1.4.0. При установці в значення ' no ' — забороняє всі види перекладу викликів. (за винятком тих, що дозволені в описі налаштувань користувача).

disallow = all : Забороняє використання всіх кодеків (глобальна настройка)

allowguest = yes (за замовчуванням) | no : Дозволити або заборонити гостьові виклики (за замовчуванням — yes, як значення можна вказати параметр ' osp ', якщо ASTERISK зібраний з підтримкою OSP)

Autocreatepeer = yes | no : Якщо дозволено, хто завгодно може використовувати сервер в якості бенкету (без перевірки на можливість доступу ; може бути зручно, при роботі з SER SIP проксі).

bindaddr = 0.0.0.0 : IP адреса, на який ASTERISK прийматиме IP пакети SIP викликів

bindport = 5060 : номер порту, на який ASTERISK прийматиме IP пакети SIP викликів

callerid = < рядок > : Інформація для Caller * Id, використовується, коли немає нічого з того, що можна було б використовувати як цього значення. За замовчуванням -» ASTERISK». (Можливість перепризначити значення за замовчуванням є у версії ASTERISK 1.0.9. Насчет інших версій немає повної впевненості.)

canreinvite = update | yes | no (глобальна настройка). З деяких причин, значення за замовчуванням — ' Yes ', будьте уважні...

context = <contextname> : Це використовуваний контекст за замовчуванням, який використовується, коли для клієнта не визначений свій контекст. Цей контекст, певний для клієнта, використовується для маршрутизації викликів від цього клієнта до потрібного місця призначення. Вміст контексту описується в файлі плану набору — extensions.conf.

defaultexpiry = 120 : Тривалість періоду вхідної або вихідної реєстрації.

dtmfmode : inband | info | rfc2833 (глобальна настройка)

domain = domains : Список доменів, розділених комами, за які відповідає сервер ASTERISK. (з'явилося в ASTERISK 1.2.x)

externip = 200.201.202.203 : IP адреса, який буде використовуватися в SIP повідомленнях, якщо наш сервер перебувати за NAT.

externhost ? = Hostname.tld : (новий параметр в ASTERISK 1.2.x)

externrefresh ? = Xxx : Цим параметром ми визначаємо, як часто буде проводитись пошук запису в DNS імені хоста, визначеному в параметрі ' externhost ' (новий параметр в ASTERISK 1.2.x)

localnet = 192.168.1.0/255.255.255.0 : визначення локальної мережі та її маски.

fromdomain = <domain> : Установка домену за умовчанням в полі From: SIP повідомлень, при роботі в якості SIP ua (клієнта).

maxexpiry = 3600 : Максимально дозволена тривалість реєстрації.

minexpiry = 60 : (Мінлива з'явилася, починаючи з версії ASTERISK 1.4.0) мінімально дозволена тривалість реєстрації.

nat = yes | no (установка для з'єднань з бенкетами) Зверніть увагу, що в ASTERISK 1.0.x, параметр nat може приймати значення: no | never | route | yes.

notifymimetype = text / plain : Дозволяє перевизначити mime type в повідомленнях MWI NOTIFY, що використовуються в повідомленнях, які відправляються системою голосових ящиків ASTERISK.

pedantic ? = Yes | no : Включити повільну, педантичну перевірку полів Call — ID і всіх рядків у багаторядковому заголовку SIP повідомлення і кодованих URI заголовків

port = <portno> : SIP порт, використовуваний за замовчуванням. (це не той порт, на якому Ваш ASTERISK чекає IP пакети. Дивись параметр : bindport)

promiscredir = yes | no : Включення підтримки повідомлень 302 Redirects ; (No — буде переадресувати всі до локального екстеншену, який отримано в поле Contact, а не до екстеншену, який вказаний в полі призначення виклику.)

realm = realm (Цією налаштуванням можна змінити realm для авторизації, зі значення за замовчуванням — ASTERISK, на будь-який обраний Вами. Працює, починаючи з ASTERISK версії 1.x)

register ? => <username> @ <sip Client/peer id in sip.conf> / <contact> : Зареєструватися на сервері SIP провайдера.

srvlookup = yes | no : Включити DNS SRV пошук для викликів

tos = <value> : Даний параметр видалений, починаючи з версії ASTERISK 1.4.0. Установка параметрів IP QoS ? для вихідних медіапотоків.

t1min = <value> : (Мінлива з'явилася, починаючи з версії ASTERISK 1.4.0) Мінімальна затримка проходження повідомлень до хоста і назад (roundtrip time), для якого відстежується стан (monitored host). Значення за замовчуванням — 100 ms.

tos_sip = <value> : (Мінлива з'явилася, починаючи з версії ASTERISK 1.4.0) Установка параметрів IP QoS ? для SIP пакетів.

tos_audio = <value> : (Мінлива з'явилася, починаючи з версії ASTERISK 1.4.0) Установка параметрів IP QoS ? для пакетів, що передають аудіодані по протоколу RTP.

tos_video = <value> : (Мінлива з'явилася, починаючи з версії ASTERISK 1.4.0) Установка параметрів IP QoS ? для пакетів, що передають відео дані по протоколу RTP.

videosupport = yes | no : Включає підтримку SIP відео

useragent : Параметр дозволяє змінити значення SIP заголовка»User — Agent».

trustrpid = yes | no : Якщо включено, то отриманого значення Remote — Party — ID ? можна довіряти.

Порядок виконання роботи

1) Встановлення Putty.

Отже, ми встановили операційну систему, визначили ip адресу, тепер потрібно поставити саму систему.

Зручніше буде працювати через putty, бо в ньому можна працювати з буфером обміну.

Завантажуємо, запускаємо putty і чіпляємо його до нашої centOS.

Завантажити можна тут:

http://www.chiark.greenend.org.uk/~sgtatham/putty/download.html

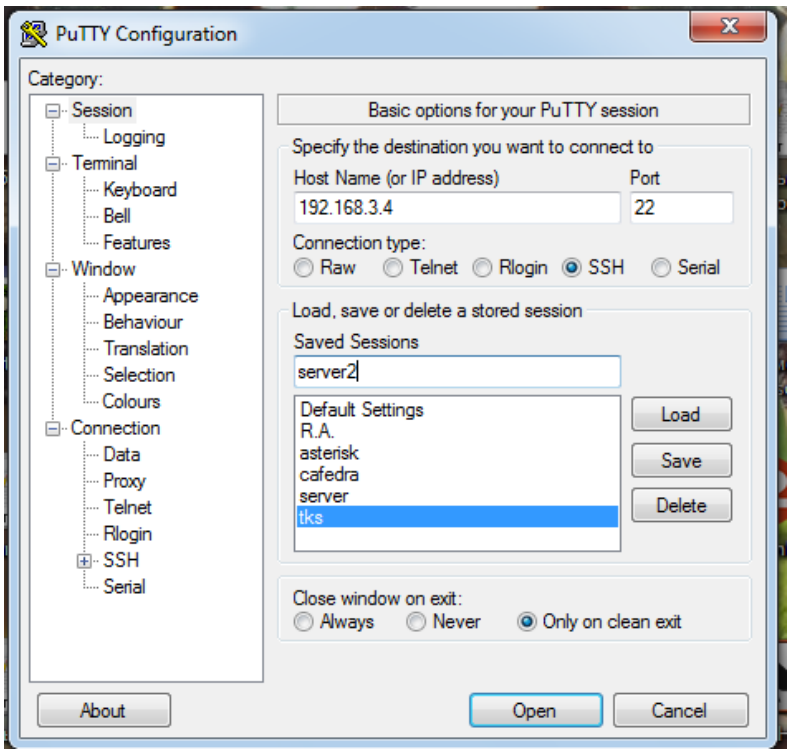

Рис. Д2.1. Налаштування putty

Зберігаємо і запускаємо, при авторизації вести пароль і логін який скаже викладач.

2) Встановлення ASTERISK.

a) Відключаємо систему безпеки SELinux:

sed -i s/SELINUX=enforcing/SELINUX=disabled/g /etc/selinux/config

b) Встановлення необхідних компонентів для встановлення ASTERISK:

yum install -y make wget openssl-devel ncurses-devel newt-devel libxml2-devel kernel-devel gcc gcc-c++ sqlite-devel

c) Завантажуємо вихідний код ASTERISK. Для цього переходимо в папку:

cd /usr/src/

та завантажуємо за допомогою команди wget:

wget http://downloads.asterisk.org/pub/telephony/dahdi-linux-complete/dahdi-linux-complete-current.tar.gz

wget http://downloads.asterisk.org/pub/telephony/libpri/libpri-1.5.0.tar.gz

wget http://downloads.asterisk.org/pub/telephony/asterisk/asterisk-11-current.tar.gz

d) Розпаковуємо скачані архіви:

tar zxvf dahdi-linux-complete*

tar zxvf libpri*

tar zxvf asterisk*

e) Встановлюємо LibPRI

cd /usr/src/libpri*

make && make install

f) Переходимо в директорію, в яку розпакували ASTERISK:

cd /usr/src/asterisk*

g) Запускаємо конфігураційні скрипти для ASTERISK. Для цього, спочатку дізнаємося якої бітності наш ASTERISK.

Набираємо:

uname –a

Якщо відповідь: 2.6.18-238.12.1.el5 #1 SMP Tue May 31 13:23:01 EDT 2011 i686 i686 i386 GNU/Linux – то значить 32 біти

Якщо відповідь: 2.6.18-238.19.1.el5 #1 SMP Fri Jul 15 07:31:24 EDT 2011 x86_64 x86_64 x86_64 GNU/Linux – то значить 64 біти

В залежності від того, яка бітність ASTERISK, запускаємо конфігураційний скрипт:

Для 32:

./configure && make menuselect && make && make install

Для 64:

./configure --libdir=/usr/lib64 && make menuselect && make && make install

Увага! Може виникнути проблема, при якій після виконання останньої команди виникне помилка. Буде лаятися на .xml файл. Тоді необхідно додати рядок до цієї команди, після чого для 64-біта буде виглядати так:

./configure --(команда, яку запропоновує астеріск) --libdir=/usr/lib64 && make menuselect && make && make install

Про успіх встановлення свідчить синє вікно.

h) Додаємо підтримку дзвінків. Справа в тому, що при такій конфігурації ASTERISK начебто як працює, але дзвінки здійснюватися не будуть. Виникатиме помилка

[Apr 27 21:35:51] ERROR[1225][C-00000009]: rtp_engine.c:259 ast_rtp_instance_new: No RTP engine was found. Do you have one loaded?

Тому, робимо наступне:

yum install uuid uuid-devel libuuid libuuid-devel uuid-c++

після цього:

./configure

make menuselect

та потім:

make

make install

i) Далі встановлюємо образ. Без установки цих образів у нас не з'являться конфігураційні файли sip.conf і extensions.conf

make samples

make config

j) друкуємо

cd

після друкуємо:

reboot

k) Після перезавантаження пишемо

asterisk

В результаті ASTERISK має запуститися. Вітаю! Ми запустили ASTERISK.

- Тепер необхідно фаєрвол в самій CentOS.

Для цього пишемо команди:

service iptables save

service iptables stop

chkconfig iptables off

- Переходимо безпосередньо до налаштування sip.conf:

nano /etc/asterisk/sip.conf

У нас відкривається файл. Пишемо свої конфіги в самий початок файлу. У моєму випадку це визначення двох sip клієнтів (телефонів):

[1001]
type=friend
regexten=1001
secret=1234
context=outcoling
host=dynamic
callerid="1001" <1001>
disallow=all
allow=alaw
allow=ulaw
language=ru
callgroup=1
pickupgroup=1
qualify=yes
canreinvite=yes
call-limit=4
nat=no

[1002]
type=friend
host=dynamic
insecure=invite
username=1002
secret=45678
context=outcoling
disallow=all

allow=alaw

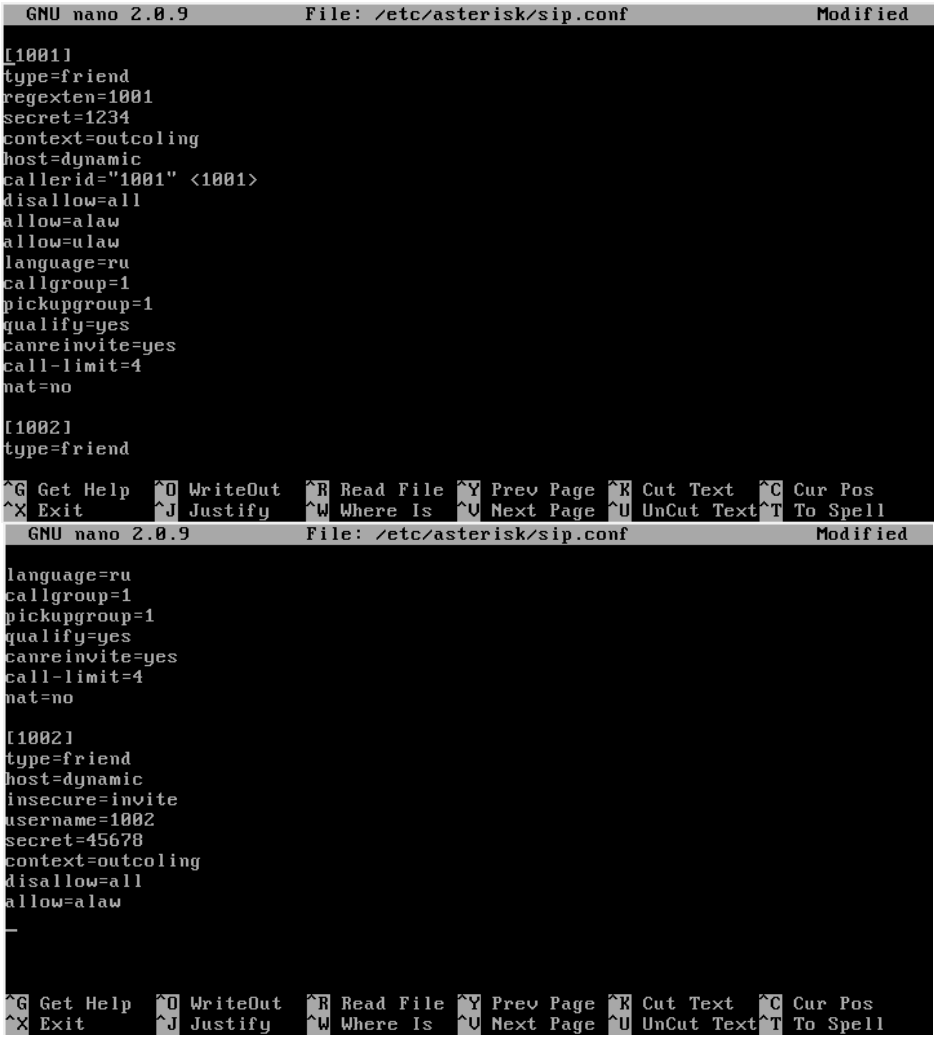

Рис. Д2.2. Налаштування sip.conf

Після цього знаходимо секцію [general] і видаляємо її. Так само видаляємо напис «context = public» після напису [general].

Звертаємо увагу на контексти.

Для телефонів (sip клієнтів) [1001] і [1002] це outcoling.

Що таке контекст і навіщо він потрібен? Контекст пов'язує файл sip.conf з файлом extensions.conf. Тобто якщо у [1001] прописаний контекст outcoling, то [1001] буде шукати правило в extensions.conf під назвою outcoling.

Натискаємо ctrl + x, натискаємо y і натискаємо enter. Файл збережений.

- Переходимо до редагування файлу extensions.conf

nano /etc/asterisk/extensions.conf

В кінці файлу пишемо наш діаплан, після чого зберігаємо файл:

[outcoling]

exten => _XXXX,1,Dial(SIP/${EXTEN},,m)

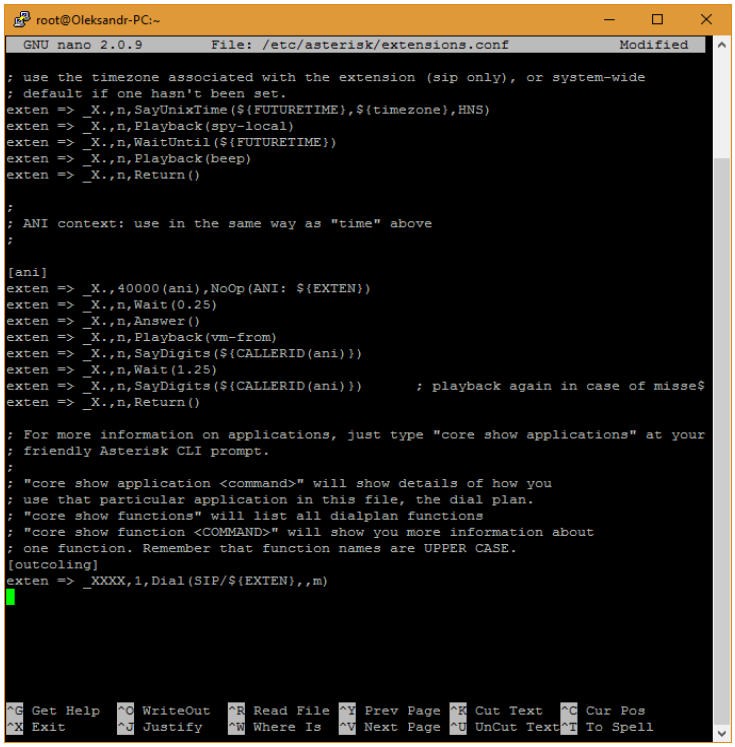

Рис Д2.3. Налаштування extensions.conf

- Далі пишемо:

asterisk –r

Це ми з управління Linux перейшли в управління ASTERISK (як нібито відкрили вікно програми і почали з нею працювати)

- Пишемо

core reload

Тим самим ми змусили ASTERISK перерахувати всі конфігураційні файли і прийняти зміни.

Коннектимо софтфон до ASTERISK і пробуємо зробити перший телефонний дзвінок між двома внутрішніми абонентами.

Контрольні запитання

1. За якою адресою знаходиться файл конфігурації ASTERISK — extensions.conf
2. Що таке VoIP?
3. Розкажіть про протоколи VoIP
4. Складова абонента в sip.conf
5. Складова extensions.conf

ДОДАТОК 3
ЛАБАРАТОРНА РОБОТА
НАЛАШТУВАННЯ ASTERISK НА РОБОТУ ЧЕРЕЗ ТРАНК

Мета: навчитися налаштовувати ASTERISK на роботу через транк та перенаправлення викликів.

Основні теоретичні відомості

SIP trunk — це віртуальний канал зв'язку між провайдером (оператором) IP-телефонії та офісної IP-АТС клієнта, що дозволяє підключити, будь-яку кількість телефонних номерів. У свою чергу кожен такий номер, може мати необмежену кількість каналів (кількість одночасних розмов по одному номеру).

SIP trunk для передачі викличної сигналізації використовує протокол SIP 2.0, VoIP телефонії, описаний в RFC 3261 і його доповнення (RFC 2976, RFC 3262, RFC 3265, RFC 3428, RFC 3515, RFC 3903).

Підключивши SIP транк від провайдера телефонії, компанія не обмежує себе в обміні даними (голос, відео і чат), на відміну від традиційної аналогової телефонії, це дозволяє відмовитися від багатопарних фізичних кабельних ліній з'єднують її з міською АТС. Більш того ваш провайдер надає зовнішній номер, може перебувати територіально в іншій державі.

Другим видом SIP транка, є віртуальний канал між двома IP-АТС, в разі потреби, територіально рознесені фізично. Це дає можливість організувати спільну внутрішню, номерну ємність обох АТС і використовувати загальні зовнішні лінії, для вихідного і вхідного зв'язку.

Кожен телефонний номер в транке називається і ідентифікується як DID (Direct Inward Dialing), який вам призначає оператор VoIP телефонії.

Переваги SIP транків

Перехід компанії від аналогової телефонії до SIP транка надає компанії цілий ряд переваг:

- Низька абонентська плата за канал і DID номер — абонентська плата за голосові канали та DID номера істотно нижче, ніж плата за ту ж кількість аналогових ліній.
- Висока якість послуг, що надаються — для встановлення з'єднання і передачі голосу, відео використовуються цифрові протоколи передачі даних, що дає високу якість голосу і високу швидкість з'єднання.
- Зміна офісу зі збереженням єдиного номера — ваш SIP транк не прив'язаний до вашого офісу. Ви можете переїхати в інший офіс, зберігши свій колишній номер. Відповідно не потрібно повідомляти клієнтів про новий номер, змінювати рекламу, візитки і платити оператором за установку переадресації і підключати нові номери.
- Не потрібно купувати VoIP шлюз — SIP транки позбавляють вас від необхідності купувати і підтримувати VoIP шлюзи. Виклики з телефонної мережі приходять безпосередньо по SIP протоколу. Крім того, відсутність перетворення технологій (з аналогової в IP) забезпечує більш високу якість зв'язку.
- Масштабованість ємності каналів — якщо у вас зросла кількість дзвінків, ви можете швидко і просто додати додаткові голосові канали до існуючого SIP транка. Порівняйте це з витратами часу на замовлення і прокладку додаткових аналогових ліній, установку плат розширення і програмування аналогової АТС.

Порядок виконання роботи

1) Укладаємо договір з sip провайдером (провайдером ip телефонії). Отримуємо від нього дані. У моєму випадку це zadarma.

Після реєстрації ви отримуєте свої налаштування SIP:

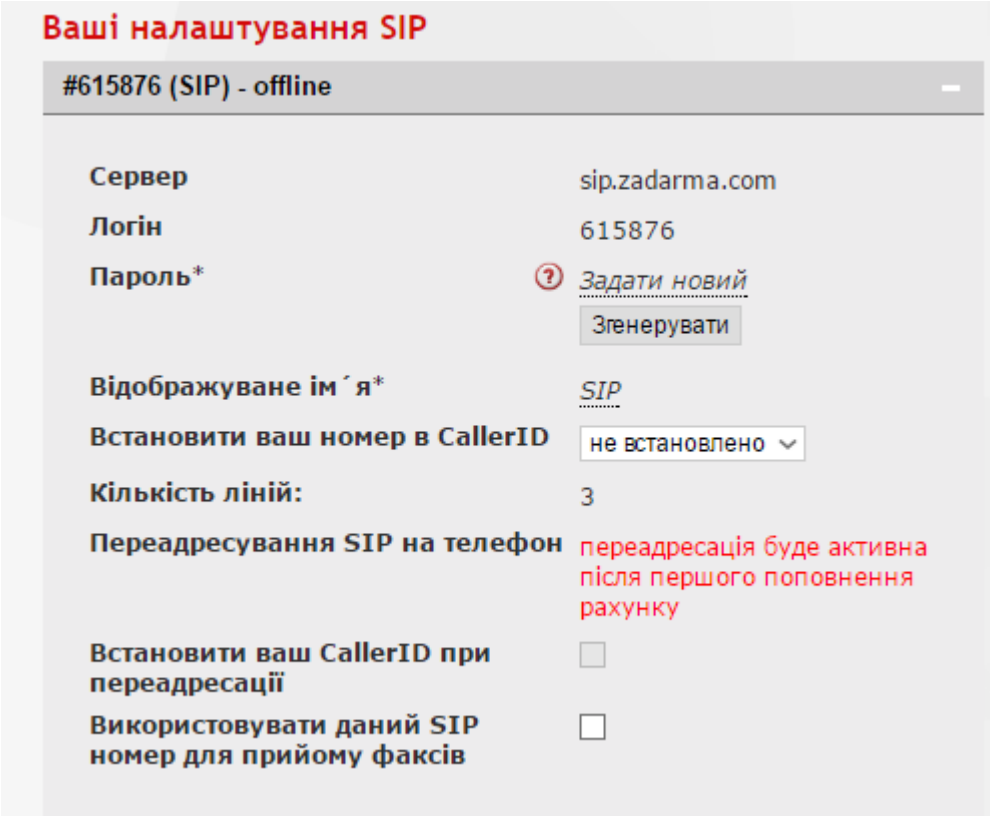

Рис. Д3.1. Налаштування SIP

2) Заходимо в файл sip.conf

nano /etc/asterisk/sip.conf

І над нашими sip клієнтами [1001] та [1002] пишемо наступний код:

[general]

register => 00000:password@sip.zadarma.com/00000

[zadarma]

type=friend

username=00000

secret=password

fromuser=00000

fromdomain=sip.zadarma.com

host=sip.zadarma.com

nat=yes

insecure=invite

context=incoming

canreinvite=no

де 00000 – це ваш логін на сайті провайдера, а password – пароль звідти ж.

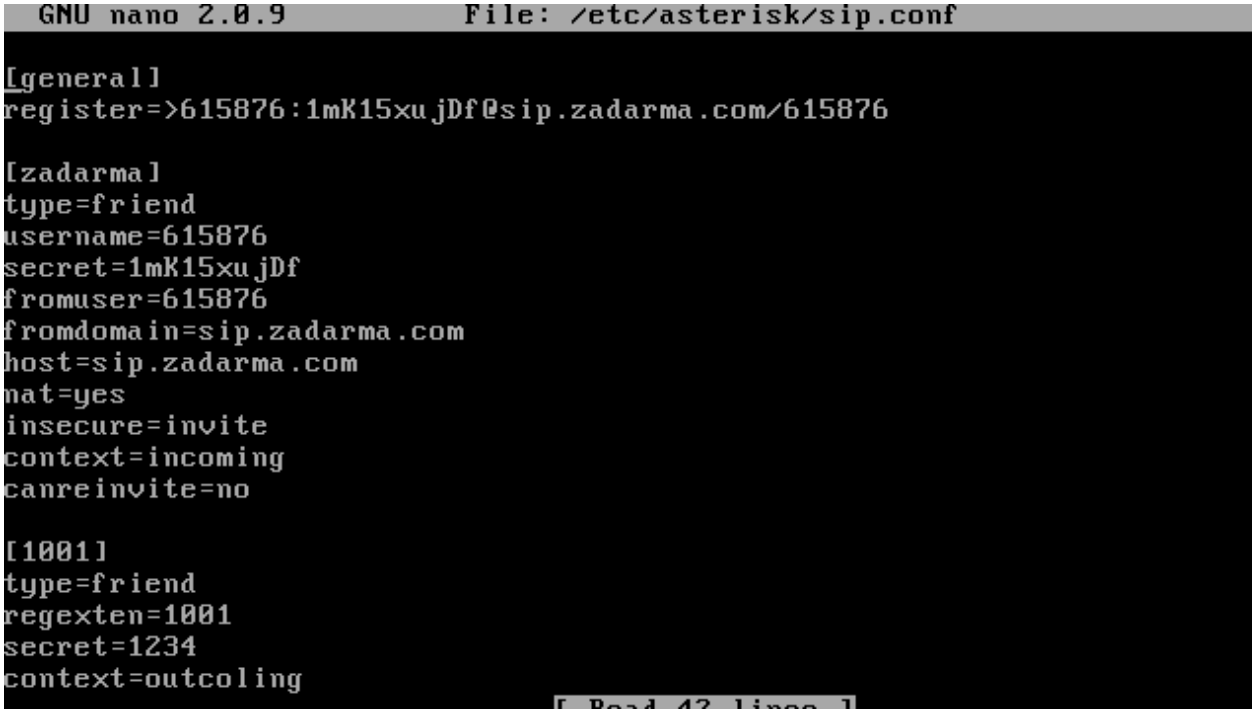

Рис. Д3.2. Редагування sip.conf

Після того, як ми вставили цей текст, нижче знайдемо ще один [context] і видалимо його. Так після [context] буде рядок: context = public — видалимо його.

3) Зберігаємо файл і заходимо в extensions.conf

nano /etc/asterisk/extensions.conf

До контексту [outcoling] додаємо наступне:

exten => _XXXXXXXXXXX,1,Dial(SIP/zadarma/${EXTEN})

Крім того, після контексту [outcoling] додаємо ще один контекст:

[incoming]

exten => _X.,1,Dial(SIP/1001&SIP/1002,60,m,tT)

Все разом це виглядає так:

[outcoling]

exten => _XXXXXXXXXXX,1,Dial(SIP/zadarma/${EXTEN})

exten => _XXXX,1,Dial(SIP/${EXTEN},,m)

[incoming]

exten => _X.,1,Dial(SIP/1001&SIP/1002,60,m,tT)

Рис. Д3.3. Редагування extensions.conf

4) Зберігаємо файл та пишемо:

asterisk –r

5) Перезапускаємо нашу ОС та пишемо:

sip show registry

Таким чином ми перевіряємо «чи піднявся транк», тобто визначаємо чи з'єднався наш ASTERISK з провайдером ip телефонії. Якщо все ОК, у відповідь на команду sip show registry ми отримаємо відповідь:

1 SIP REGISTRATION

6) Спробуємо зробити дзвінок з ASTERISK наприклад на мобільний телефон і з мобільного телефону на ASTERISK.

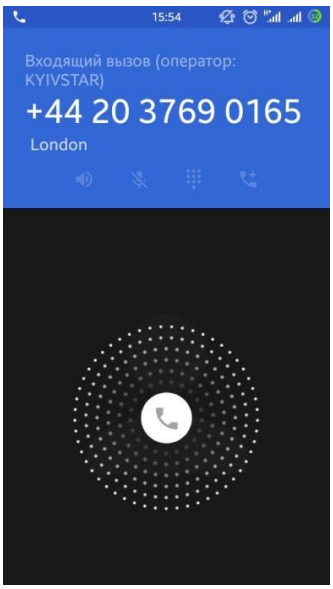

Рис. Д3.4. Ілюстрація дзвінка на мобільний через транк

7) Трапляється так, що наприклад секретар отримав дзвінок і секретарю цей дзвінок потрібно направити, наприклад менеджеру. Тому реалізуємо цю функцію. Для цього виконуємо такі дії:

в extensions.conf в Dial додаємо параметр t. Цей параметр означає, що для цього Dial плану дозволено перенаправлення дзвінків.

Це буде виглядати так:

exten => _XXXX,1,Dial(SIP/${EXTEN},,t&m,)

```
 GNU nano 2.0.9       File: /etc/asterisk/extensions.conf

[outcoling]
exten => _XXXXXXXXXXX,1,Dial(SIP/zadarma/${EXTEN})
exten => _XXXX,1,Dial(SIP/${EXTEN},,t&m,)

[incoming]
exten => _X.,1,Dial(SIP/1001&SIP/1002,60,m,tT)
```

Рис. Д3.5. Редагування extensions.conf

Тут ми бачимо параметр t & m. Тобто відразу два параметри — t для перекладу дзвінків і m для музики.

Тепер потрібно розібратися в поняттях blind transfer і attended transfer.

Blind transfer використовується для сліпого перенаправлення дзвінків і працює за замовчуванням. Що означає сліпе перенаправлення дзвінків?

Це коли секретар переводить дзвінок менеджеру і секретарю все одно, що трапиться з дзвінком далі. До неї цей дзвінок вже ніколи не повернеться. Тобто це проста переадресація, без зворотного зв'язку.

Коли хтось подзвонив і БУЛО таких з'єднань секретарем, то секретар натискає на #, вводить внутрішній або будь-який зовнішній номер телефону, на який хоче перевести абонента, що телефонує (ну наприклад на менеджера) і все.

Інша справа attended transfer. Attended transfer дозволяє секретарю не просто перевести дзвінок, але і контролювати успішність його перенаправлення. Уявімо ситуацію: секретар перенаправляє дзвінок менеджеру. Якщо менеджер не відповідає протягом заданого кількості часу, або менеджер просто натиснув на червону трубку (скинув), то дзвінок повертається назад до секретаря. А там секретар вже скаже — вибачте, менеджера зараз немає або він зайнятий.

Давайте реалізуємо. Отже, в контексті для Dial ми задали параметр t. Тепер у нас працює сліпий трансфер (blind transfer) через #.

8) Для Attended transfer все складніше:

- Переходимо до редагування файла features.conf

nano /etc/asterisk/features.conf

Знаходимо рядки:

atxfernoanswertimeout = 15

atxferdropcall = no

atxferloopdelay = 10

atxfercallbackretries = 2

atxfer => *2

та розкоментуємо їх (видаляємо крапку з комою на початку рядка)

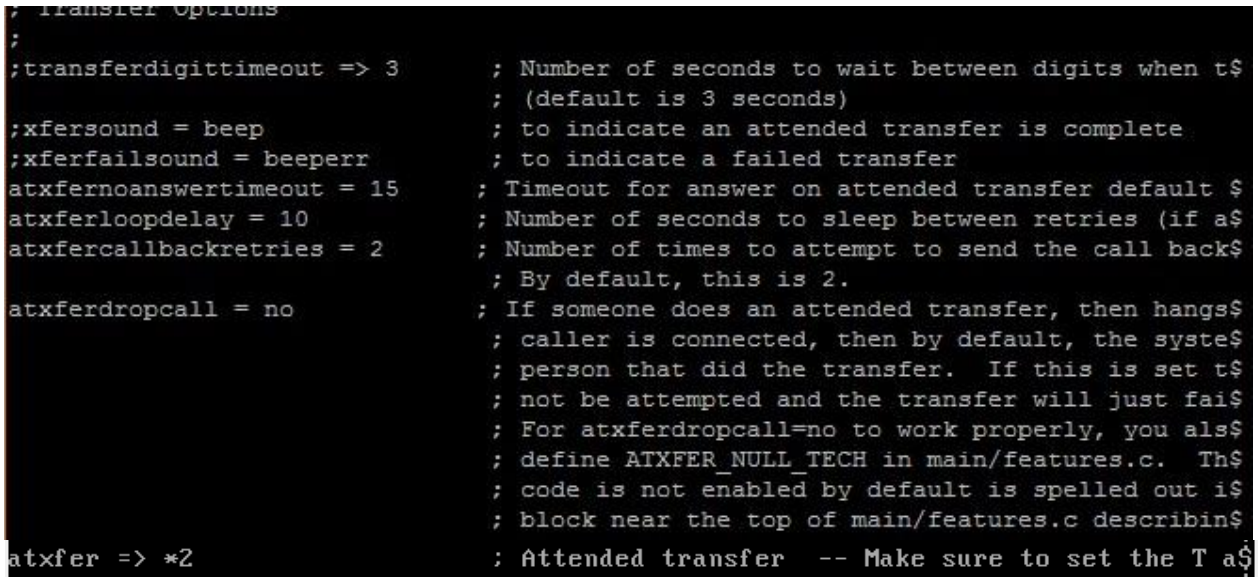

Рис. Д3.6. Редагування features.conf

Тепер спробуємо зробити дзвінок і перенаправити його.

1) Дзвонимо з мобільника на Астеріск на номер секретаря.

2) Встановлюємо з'єднання. Секретар каже співрозмовнику, щоб почекав, поки вона перемкне

3) Секретар набирає на телефоні * 2 і номер, на який вона хоче перекинути. Ну наприклад 1002

4) Номер 1002 бере трубку. Секретар запитує у номера 1002 чи хоче він розмовляти. Якщо хоче, то секретар кладе трубку

5) Після того, як секретар поклав трубку, дзвінок уже переходить до менеджера (1002). Якщо менеджер не відповість протягом 15 секунд або повісить трубку, нас знову відішлють до секретаря.

Зверніть увагу на пункт 4. Якщо номер 1002 не відповів чи скинув виклик, секретар знову почне розмовляти з мобільником (скаже що менеджер зараз зайнятий або відсутній).