

Testowanie typów wiadomości w protokole SIP

Streszczenie

Rozwój sieci IP, telefonii komórkowej oraz rosnące zapotrzebowanie na zintegrowane usługi komunikacyjne sprawiły, że technologie VoIP stały się fundamentem współczesnej telekomunikacji. Tradycyjna telefonia oparta na sieci PSTN została zastąpiona w dużej części rozwiązaniami wykorzystującymi transmisję pakietową, umożliwiającymi realizację połączeń głosowych, wideo oraz usług multimedialnych w oparciu o infrastrukturę internetową.

Kluczową rolę w tym środowisku odgrywa protokół SIP, który odpowiada za zestawianie, modyfikowanie i kończenie sesji komunikacyjnych. To właśnie on stanowi podstawę działania nowoczesnych systemów VoIP oraz architektur operatorskich, takich jak IMS Core. Wraz z rozwojem tych technologii, pojawiła się również potrzeba zapewnienia bezpieczeństwa oraz kontroli, co doprowadziło do powstania rozwiązań typu Session Border Controller (SBC). SBC jak każde oprogramowanie powinno być sprawdzone pod kątem jego jakości, co implikuje potrzebę wykonania testów.

Niniejsza publikacja stanowi propozycję koncepcji testowania wiadomości protokołu SIP wykorzystywanego w technologii VoIP.

1. Wprowadzenie

1.1 Koncepcja VoIP oraz protokół SIP

VoIP (Voice over Internet Protocol) to technologia umożliwiająca prowadzenie rozmów głosowych przez Internet zamiast przez tradycyjną sieć telefoniczną (PSTN). W klasycznej telefonii, głos przesyłany jest jako sygnał elektryczny za pomocą linii telefonicznej. W technologii VoIP głos jest:

- a) *zamieniany z sygnału analogowego na cyfrowy,*
- b) *dzielony na pakiety danych (UDP lub TCP),*
- c) *przesyłany przez sieć IP,*
- d) *odbierany i ponownie składany w dźwięk u rozmówcy.*

Technologia VoIP oferuje niższe koszty połączeń (szczególnie międzynarodowych), brak potrzeby osobnej infrastruktury telefonicznej, integrację z CRM oraz systemami IT oraz mobilność (numer działa w każdym dowolnym miejscu, gdzie dociera sieć internetowa). Jednym z protokołów używanych w VoIP do zestawiania połączeń jest **SIP**, który jest głównym wątkiem niniejszej publikacji.

1.2 Protokół SIP

SIP (*ang. Session Initiation Protocol*) to protokół inicjowania sesji, zaproponowany przez IETF, jako standard dla zestawiania sesji pomiędzy jednym lub wieloma klientami. Obecnie jest on dominującym protokołem sygnalizacyjnym dla VoIP. SIP w swojej definicji powinien dostarczać zestaw funkcji obsługujących połączenia obecne w publicznej tradycyjnej sieci telefonicznej (PSTN). Zawiera on funkcje, które umożliwiają operacje znane z telefonii stacjonarnej: wybieranie numeru, dźwięk dzwonka w aparacie telefonicznym, sygnał zajętości, itp. Jednakże ich implementacja i używana terminologia są odmiennie. Sam SIP umożliwia:

- *zestawianie połączeń głosowych (VoIP),*
- *realizację wideorozmów,*
- *prowadzenie konferencji multimedialnych,*
- *przesyłanie wiadomości błyskawicznych.*

SIP odpowiada za sygnalizację, czyli:

- *lokalizowanie użytkowników,*
- *parametry połączenia,*
- *kontrolę przebiegu sesji za pomocą odpowiednich typów wiadomości,*
- *zakończenie połączenia.*

Sam SIP nie przesyła bezpośrednio danych głosowych czy wideo. W tym celu, podczas rozmów typu VoIP, wykorzystywany jest najczęściej protokół **RTP** (Real-time Transport Protocol). Innymi słowy, jest to funkcjonalność umożliwiająca zestawienie oraz kontrolę połączenia. Wobec tego, nie zawsze istnieje zapotrzebowanie na testowanie pełnych rozmów z pakietami RTP. Możliwe jest testowanie samych tylko wiadomości SIP oraz ich przepływ pomiędzy użytkownikami (najczęściej "rozmówcami"), które zostaną szczegółowo opisane w rozdziale 2.

1.3 Session Border Controller

Session border controller (SBC) to element sieciowy (oprogramowanie lub oprogramowanie oraz sprzęt) wdrażany w celu ochrony sieci VoIP opartych na protokole **SIP**. Wczesne wdrożenia kontrolerów SBC koncentrowały się na **granicach** między dwiema sieciami dostawców usług w środowisku peeringowym (*ang. peer-to-peer, P2P*). Rola ta rozszerzyła się z czasem, zapewniając tego rodzaju usługi klientom indywidualnym i/lub korporacyjnym.

Termin **granica** odnosi się do punktu rozgraniczającego między jedną częścią sieci a drugą.

Termin **kontroler** odnosi się do wpływu, jaki kontrolery graniczne sesji wywierają na strumieniu danych tworzące **sesje**, gdy przekraczają one granice między jedną częścią sieci a drugą.

Termin **sesja** odnosi się z kolei do komunikacji między dwoma lub większą ilością stron – w kontekście tradycyjnej telefonii byłoby to połączenie. Każde połączenie składa się z jednej lub większej liczby wymian komunikatów sygnalizacji połączenia, które kontrolują połączenie oraz jednego lub większej liczby strumieni multimedialnych, które przenoszą dźwięk, obraz lub inne dane połączenia, wraz z informacjami o statystykach oraz jakości połączenia. Razem strumienie te tworzą sesje.

1.4 System IMS Core

IMS Core, czyli **IP Multimedia Subsystem** lub **IP Multimedia Core Network Subsystem** to framework architektoniczny, który umożliwia zintegrowane przesyłanie danych i głosu w sieciach IP. Architektura IMS oparta jest na protokołach **SIP**, Diameter oraz H.248. IMS Core został pierwotnie opracowany przez organizację zajmującą się standardami bezprzewodowymi, 3rd Generation Partnership Project (3GPP), jako część wizji rozwoju sieci komórkowych poza GSM. IMS wykorzystuje SIP jako **protokół sterujący**. Główne elementy IMS Core to:

- *P-CSCF – pierwszy punkt kontaktu użytkownika z systemem IMS Core,*
- *S-CSCF – centralny serwer sterujący sesjami,*
- *I-CSCF – element pośredniczący między sieciami,*
- *HSS – baza danych abonentów.*

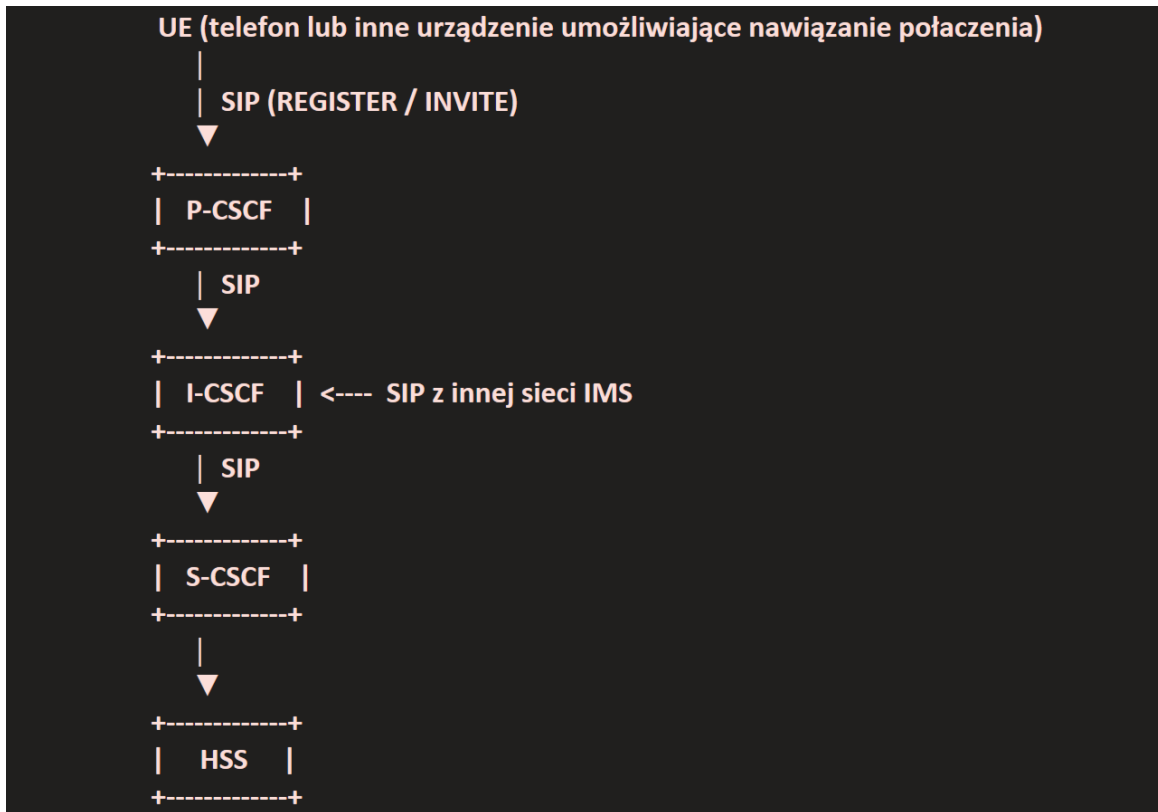
CSCF (*Call Session Control Function*) **to funkcja sterująca sesją połączenia**. W architekturze IMS jest to po prostu ogólna nazwa dla grupy elementów oraz funkcjonalności odpowiedzialnych za kontrolę sygnalizacji SIP oraz zarządzanie sesjami multimedialnymi.

P-CSCF (*Proxy-Call Session Control Function*) to pierwszy element sieci IMS, z którym komunikuje się urządzenie użytkownika po nawiązaniu łączności z operatorem. Można go traktować jako bramę wejściową do rdzenia IMS. Terminal wysyła do niego wszystkie komunikaty sygnalizacyjne SIP, a P-CSCF przekazuje je do odpowiednich elementów sieci. Oprócz samego przekazywania sygnalizacji odpowiada on za bezpieczeństwo połączenia – zestawia zabezpieczenia (np. IPsec), może kompresować sygnalizację oraz współpracuje z mechanizmami kontroli jakości usług (QoS). W praktyce, pełni rolę punktu kontrolnego pomiędzy użytkownikiem a siecią operatora i często współdziała z kontrolerem **SBC** na granicy sieci.

S-CSCF (*Serving-Call Session Control Function*) to główny element sterujący architektury IMS. Odpowiada za rejestrację i uwierzytelnienie użytkownika (we współpracy z HSS) oraz kontrolę sesji, np. VoIP. Analizuje on komunikaty SIP oraz uruchamia odpowiednie usługi, pełniąc rolę „mózgu” sieci IMS (np. wideorozmowa na kamerach).

I-CSCF (*Interrogating-Call Session Control Function*) pełni rolę pośrednika przy ruchu przychodzącym z innych sieci. Ustala, który S-CSCF obsługuje użytkownika, przekazuje do niego sygnalizację oraz ukrywa wewnętrzną strukturę sieci operatora.

W skrócie: P-CSCF – kontakt z użytkownikiem, S-CSCF – kontrola sesji oraz usług, I-CSCF – pośrednictwo oraz kierowanie ruchem między sieciami. Bardzo uproszczony schemat architektoniczny system IMS Core można przedstawić za pomocą poniższego diagramu:



2. Typy wiadomości w SIP

Jak zostało wspomniane wcześniej, protokół SIP jest protokołem sygnalizacyjnym, stosowanym w systemach VoIP do inicjowania, zarządzania oraz kończenia sesji komunikacyjnych pomiędzy użytkownikami w sieciach IP. Komunikacja (rozmowa telefoniczna, rozmowa na kamerach, MMS, itp.) w SIP odbywa się poprzez wymianę wiadomości zwanych metodami, które określają działania wykonywane w trakcie obsługi połączenia. Wiadomości te dzieli się na **metody podstawowe** oraz **metody rozszerzone**.

2.1 Metody podstawowe SIP

Podstawowe metody wykorzystywane w SIP to:

- **INVITE** – służy do rozpoczęcia sesji komunikacyjnej pomiędzy użytkownikami. W wiadomości tej przesyłane są również informacje o parametrach połączenia (np. kodeki audio oraz video),
- **ACK** – potwierdza otrzymanie odpowiedzi na wiadomość INVITE oraz kończy proces zestawiania połączenia (po zaakceptowaniu połączenia przez drugą stronę, urządzenie wysyła ACK, co umożliwia rozpoczęcie rozmowy),
- **BYE** – służy do zakończenia już istniejącej sesji komunikacyjnej (najczęściej rozmowy),
- **CANCEL** – umożliwia anulowanie próby zestawienia połączenia, zanim zostanie ono nawiązane (odrzućcie połączenia),
- **REGISTER** – pozwala urządzeniu użytkownika zarejestrować swoją lokalizację na serwerze SIP, aby inne urządzenia mogły się z nim połączyć (oprogramowanie typu softphone, WhatsApp, itp.),
- **OPTIONS** – służy do sprawdzania możliwości urządzenia lub serwera, np. obsługiwanych metod czy kodeków (np. kompatybilność parametrów z innym urządzeniem np. czy druga osoba ma kamerę i mikrofon),

Do grupy metod podstawowych należy również **reINVITE**, który jest ponownym użyciem metody INVITE w już istniejącym dialogu SIP. Nie jest to osobna metoda ani metoda rozszerzona – jest to **wariant podstawowej metody INVITE**. Dzięki niej można zmienić parametry połączenia, np. kodek, adres strumienia multimedialnego, dodać/usunąć strumienie wideo czy audio, bez przerywania rozmowy (np. modyfikacja ustawień protokołu SDP).

2.2 Metody rozszerzone SIP

Oprócz metod podstawowych w SIP występują również metody rozszerzone, które zapewniają dodatkowe funkcjonalności SBC:

- **INFO** – umożliwia przesyłanie dodatkowych informacji w trakcie trwania sesji (urządzenie wysyła informacje o statusie połączenia lub o aktywności użytkownika np. „rozpoczęto udostępnianie ekranu”),
- **PRACK** – służy do potwierdzania tymczasowych odpowiedzi serwera (tymczasowe komunikaty serwera otrzymują potwierdzenie PRACK, co zapewnia poprawną synchronizację sesji),
- **UPDATE** – pozwala zmieniać parametry sesji bez konieczności jej ponownego zestawiania (dodanie wideo do istniejącej rozmowy głosowej),
- **SUBSCRIBE** – umożliwia zapisanie się na subskrypcję powiadomień o zdarzeniach (np. status użytkownika, urządzenie subskrybuje informacje o dostępności innych użytkowników w sieci),
- **NOTIFY** – przesyła informacje o zdarzeniach do wcześniej zapisanych klientów (serwer wysyła powiadomienie o statusie użytkownika np. że jest teraz dostępny),
- **MESSAGE** – pozwala wysyłać krótkie wiadomości tekstowe w ramach SIP, (wiadomość tekstowa która może zostać przesłana równolegle do trwającej sesji komunikacyjnej),

- **REFER** – umożliwia przekierowanie połączenia do innego użytkownika (przekierowanie do innego urządzenia lub użytkownika w sieci, np. w przypadku obsługi klienta).

2.3 Kody odpowiedzi w protokole SIP

Protokół SIP używa trójcyfrowych kodów odpowiedzi analogicznie jak protokół HTTP:

Klasa kodów	Zakres kodów	Znaczenie
1xx	100–199	Informacyjne – żądanie przyjęte, trwa przetwarzanie
2xx	200–299	Sukces – żądanie pomyślnie zrealizowane
3xx	300–399	Przekierowanie – potrzebna jest dodatkowa akcja
4xx	400–499	Błąd użytkownika / klienta (User Agent Client) – niewłaściwe żądanie
5xx	500–599	Błąd serwera – problem po stronie serwera
6xx	600–699	Problem globalny – żądanie nie może być obsłużone przez żaden serwer

SIP przyjmuje koncepcję kodów odpowiedzi z HTTP, ale poszerza ją o własne znaczenia i zachowania, szczególnie w kontekście sygnalizacji połączeń multimedialnych.

3. Testowanie wiadomości SIP

Testowanie komunikacji w protokole SIP może być realizowane zarówno manualnie, jak i automatycznie. W obu przypadkach, niezbędne jest przygotowanie odpowiedniego środowiska testowego oraz dobór właściwych narzędzi.

W przypadku testów automatycznych konieczne jest zaprojektowanie i implementacja dedykowanego frameworka testowego, analogicznie do rozwiązań stosowanych w innych obszarach IT. Ze względu na to, że przepływ sygnalizacji SIP odbywa się na poziomie backendu i nie jest widoczny bezpośrednio dla użytkownika, testy te mają zazwyczaj charakter zbliżony do testów programistycznych – opierają się na analizie komunikatów oraz logiki wymiany sygnałów.

Testy manualne w obszarze VoIP oraz SIP realizowane są najczęściej przy użyciu następujących narzędzi:

- Softphone, np. Zoiper,
- analizatory ruchu sieciowego,
- fizyczne telefony IP.

W niniejszej publikacji zostaną zaproponowane scenariusze testowe bez pokazywania kodu, wewnętrznej struktury wiadomości oraz dedykowanych narzędzi. Ważna jest tutaj przyjęta koncepcja oraz zrozumienie kontekstu testowania. Przyjęta zostanie strategia podziału na testy pozytywne, negatywne oraz brzegowe. Zaprezentowane scenariusze testowe nie wyczerpują wszystkich możliwości. Oprócz standardowych rozmów, istnieją także rozmowy alarmowe oraz rozmowy peeringowe, ale ich testowanie będzie przebiegało analogicznie jak testowanie standardowych rozmów. Ważne jest tutaj podejście, które zostanie przedstawione.

Wraz z rozmowami SIP-owymi mogą istnieć dodatkowe powiązane funkcjonalności, jak np. liczniki zdarzeń, liczniki błędów, logowanie, alarmy, itp.

Tego rodzaju funkcjonalności będą wymagały dodatkowych testów, ale nie są one tematem niniejszej publikacji.

3.1 Wiadomości REGISTER

Rozdział ten przedstawia przypadki testowe z rejestracją użytkownika.

Założenia testów:

- 1) System: serwer SIP,
- 2) Istnieje użytkownik, którego można rejestrować,
- 3) Cel: weryfikacja poprawnej obsługi rejestracji użytkownika.

Testy pozytywne:

TC-REGISTER-001 – Poprawna rejestracja

Opis: Użytkownik wysyła poprawny REGISTER

Kroki:

1. Wyślij REGISTER z poprawnymi nagłówkami (To, From, Contact)
2. Podaj poprawne dane uwierzytelniające

Oczekiwany rezultat:

- 200 OK
- Kontakt zapisany w bazie lokalizacji

TC-REGISTER-002 – Rejestracja z uwierzytelnieniem (401 Unauthorized)

Kroki:

1. Wyślij REGISTER bez Authorization
2. Odbierz 401 Unauthorized
3. Wyślij ponownie REGISTER z nagłówkiem Authorization

Oczekiwany rezultat:

- 200 OK po drugim żądaniu

TC-REGISTER-003 – Odświeżenie rejestracji przed wygaśnięciem

Kroki:

1. Zarejestruj użytkownika (Expires: 3600)
2. Przed wygaśnięciem wyślij REGISTER ponownie

Oczekiwany rezultat:

- 200 OK
- Odświeżony czas rejestracji

TC-REGISTER-004 – Derejestracja (Expires: 0)

Kroki:

1. Zarejestrowany użytkownik istnieje
2. Wyślij REGISTER z Expires: 0

Oczekiwany rezultat:

- 200 OK
- Usunięcie kontaktu z bazy lokalizacji

Testy negatywne:

TC-REGISTER-005 – Błędne dane logowania

Kroki:

1. Wyślij REGISTER z niepoprawnym hasłem

Oczekiwany rezultat:

- 403 Forbidden lub 401 Unauthorized

TC-REGISTER-006 – Brak nagłówka Contact

Kroki:

1. Wyślij REGISTER bez Contact

Oczekiwany rezultat:

- 400 Bad Request

TC-REGISTER-007 – Nieznany użytkownik

Kroki:

1. Wyślij REGISTER dla nieistniejącego użytkownika

Oczekiwany rezultat:

- 404 Not Found

Testy brzegowe

TC-REGISTER-008 – Bardzo krótki czas Expires

Kroki:

1. REGISTER z Expires: 1

Oczekiwany rezultat:

- Akceptacja lub wymuszenie minimalnej wartości

TC-REGISTER-009 – Bardzo długi Expires

Kroki:

1. REGISTER z Expires: 999999

Oczekiwany rezultat:

- Skrócenie do maksymalnego dopuszczalnego czasu

TC-REGISTER-010 – Expires: 0

Kroki:

1. REGISTER z Expires: 0

Oczekiwany rezultat:

- serwer usuwa Contact natychmiast (odpowiednik wylogowania)

Nagłówek **Expires** określa, **jak długo rejestracja użytkownika ma być ważna na serwerze SIP**. Expires: 3600 oznacza 3600 sekund, czyli jedną godzinę.

Opisane powyżej przypadki testowe nie wyczerpują wszystkich możliwości. Można przeprowadzić test obciążeniowy lub przeciążeniowy, który

będzie rejestrował bardzo dużą liczbę użytkowników w krótkim czasie lub test, który nagle wyrejestruje dużą liczbę użytkowników, by następnie badać zachowanie systemu.

3.2 Wiadomości INVITE

Jak już zostało wspomniane wcześniej w rozdziale, INVITE służy do rozpoczęcia sesji komunikacyjnej pomiędzy użytkownikami. Testowanie samej wiadomości INVITE na ogół nie ma większego sensu, ponieważ byłoby oderwane od kontekstu. W celu zaprojektowania testów, wykorzystane zostaną inne rodzaje wiadomości opisane w rozdziale 2. Testowy framework musi być zaimplementowany w taki sposób, aby mógł symulować kody odpowiedzi oraz wymagane do testów typy wiadomości. Pisząc przypadki testowe przyjmowane jest założenie, że użytkownicy są zarejestrowani (o ile testy nie wymagają użytkownika bez rejestracji).

Jeżeli projektowane są testy wiadomości INVITE lub reINVITE, należy wiedzieć czym jest SDP. SDP (Session Description Protocol) to protokół służący do opisu parametrów sesji multimedialnej.

W kontekście VoIP (Voice over IP) SDP nie przesyła samego dźwięku ani wideo – zamiast tego informuje on, jak połączyć się do rozmowy: jakie formaty audio/wideo są obsługiwane, jakie porty sieciowe użyć, adresy IP, itp. Główną rolą SDP jest:

- Negocjacja kodeków multimedialnych do obsługi dźwięku oraz obrazu,
- Wybór transportu i portów – np. UDP port 5004 dla RTP (Real-time Transport Protocol).
- Określenie parametrów multimedialnych – np. częstotliwość próbkowania, liczba kanałów audio, liczba kodeków video,
- Ustalenie adresów IP dla przesyłania danych – pozwala aplikacjom VoIP wiedzieć, dokąd wysyłać dźwięk lub obraz.

Testy pozytywne INVITE

TC-INVITE-001 – Poprawne zestawienie połączenia

Kroki:

1. U1 wysyła INVITE do U2
2. U2 odpowiada 100 Trying
3. U2 odpowiada 180 Ringing
4. U2 odpowiada 200 OK
5. U1 wysyła ACK

Oczekiwany rezultat:

- 200 OK
- Połączenie zestawione

TC-INVITE-002 – Autoryzacja INVITE (poprawna)

Kroki:

1. U1 wysyła INVITE do U2 bez Authorization
2. U2 odpowiada 401 Unauthorized
3. U1 wysyła INVITE z Authorization

Oczekiwany rezultat:

- 200 OK
- Połączenie zestawione

TC-INVITE-003 – Zestawienie połączenia z różnymi nagłówkami Contact

Kroki:

1. U1 wysyła INVITE do U2 z nagłówkiem Contact (atrybut wiadomości INVITE)
2. U2 odpowiada 200 OK
3. U1 wysyła ACK

Oczekiwany rezultat:

- Połączenie poprawnie zestawione

- Kontakt zaktualizowany

TC-INVITE-004 – Zestawienie połączenia z obsługą re-INVITE w przyszłości

Kroki:

1. U1 wysyła INVITE do U2
2. U2 odpowiada 100 Trying, następnie 180 Ringing, 200 OK
3. U1 wysyła ACK

Oczekiwany rezultat:

- Połączenie zestawione
- Obsługa przyszłego re-INVITE jest możliwa

TC-INVITE-005 – Forking (wiele U2)

Kroki:

1. U1 wysyła INVITE do wielu użytkowników
2. Użytkownicy wysyłają 180 Ringing, następnie 200 OK
3. U1 wysyła ACK dla każdego 200 OK

Oczekiwany rezultat:

- Połączenie zestawione z pierwszą odpowiedzią 200 OK
- Obsługa pozostałych odpowiedzi bez błędów

TC-INVITE-006 – Poprawne zestawienie połączenia z długim SDP

Kroki:

1. U1 wysyła INVITE do U2 z SDP dużej długości
2. U2 odpowiada 200 OK
3. U1 wysyła ACK

Oczekiwany rezultat:

- Połączenie poprawnie zestawione
- SDP poprawnie przetworzone

TC-INVITE-007 – Obsługa odpowiedzi 183 Session Progress

Kroki:

1. U1 wysyła INVITE do U2
2. U2 odpowiada 183 Session Progress
3. U2 odpowiada 200 OK
4. U1 wysyła ACK

Oczekiwany rezultat:

- Połączenie zestawione
- 183 Session Progress poprawnie obsłużone

TC-INVITE-008 – Zestawienie połączenia z wymaganym P-Asserted-Identity

Kroki:

1. U1 wysyła INVITE do U2 z nagłówkiem P-Asserted-Identity
2. U2 odpowiada 200 OK
3. U1 wysyła ACK

Oczekiwany rezultat:

- Połączenie poprawnie zestawione
- P-Asserted-Identity prawidłowo odczytane

Testy negatywne INVITE

TC-INVITE-009 – INVITE z niepoprawnym URI

Kroki:

1. U1 wysyła INVITE do niepoprawnego URI U2
2. U2 odpowiada 404 Not Found

Oczekiwany rezultat:

- Połączenie nie zostaje zestawione
- U1 otrzymuje odpowiedź 404

TC-INVITE-010 – INVITE bez nagłówka To

Kroki:

1. U1 wysyła INVITE do U2 bez nagłówka To
2. U2 odpowiada 400 Bad Request

Oczekiwany rezultat:

- Połączenie nie zostaje zestawione
- Błąd 400 wskazuje brak wymaganego nagłówka

TC-INVITE-011 – INVITE z niepoprawnym nagłówkiem From

Kroki:

1. U1 wysyła INVITE do U2 z niepoprawnym formatem nagłówka From
2. U2 odpowiada 400 Bad Request

Oczekiwany rezultat:

- Połączenie nie zostaje zestawione
- Błąd 400 wskazuje niepoprawny nagłówek

TC-INVITE-012 – INVITE z nagłówkiem Content-Type nieobsługiwanym przez U2

Kroki:

1. U1 wysyła INVITE z Content-Type np. application/unknown
2. U2 odpowiada 415 Unsupported Media Type

Oczekiwany rezultat:

- Połączenie nie zostaje zestawione
- U2 odrzuca wiadomość z powodu nieobsługiwanego typu treści

TC-INVITE-013 – INVITE do zajętego użytkownika (U2)

Kroki:

1. U1 wysyła INVITE do U2, który jest już w trakcie połączenia
2. U2 odpowiada 486 Busy Here

Oczekiwany rezultat:

- Połączenie nie zostaje zestawione
- U1 otrzymuje odpowiedź 486

Testy brzegowe INVITE**TC-INVITE-014 – INVITE bez nagłówka Contact****Kroki:**

1. U1 wysyła INVITE do U2 bez nagłówka Contact
2. U2 odpowiada 484 Address Incomplete

Oczekiwany rezultat:

- Połączenie nie zostaje zestawione
- U1 otrzymuje odpowiedź wskazującą brak wymaganego nagłówka

TC-INVITE-015 – INVITE z nagłówkiem Max-Forwards = 0**Kroki:**

1. U1 wysyła INVITE do U2 z nagłówkiem Max-Forwards ustawionym na 0
2. U2 odpowiada 483 Too Many Hops

Oczekiwany rezultat:

- Połączenie nie zostaje zestawione
- Serwer zgłasza, że pakiet nie może zostać przesłany dalej

TC-INVITE-016 – INVITE z bardzo dużym nagłówkiem**Kroki:**

1. U1 wysyła INVITE z nagłówkiem zawierającym maksymalną dozwoloną długość (np. 64 KB)
2. U2 odpowiada 200 OK lub 400 Bad Request w zależności od obsługi

Oczekiwany rezultat:

- Sprawdzenie limitów serwera SIP
- Połączenie może zostać odrzucone, jeśli nagłówek przekracza dopuszczalny rozmiar

TC-INVITE-017 – INVITE z pustym nagłówkiem Call-ID

Kroki:

1. U1 wysyła INVITE z pustym Call-ID
2. U2 odpowiada 400 Bad Request

Oczekiwany rezultat:

- Połączenie nie zostaje zestawione
- Brak Call-ID powoduje odrzucenie wiadomości

Atrybut Contact określa adres, pod którym można bezpośrednio skontaktować się z nadawcą SIP (urządzeniem lub klientem). Natomiast atrybut Max-forwards zapobiega nieskończonemu krążeniu wiadomości SIP w sieci.

3.3 Wiadomość reINVITE

Zaproponowany zestaw testów jest częściowo oparty na SDP, analogicznie jak testy do INVITE. W tym celu należy znać takie atrybuty jak:

- a) **Session Timer (RFC 4028)** – mechanizm ograniczenia czasu życia sesji SIP: jeśli strony przestaną się „odzywać”, połączenie ma być świadomie zakończone (np. BYE), Odświeżanie wykonywane jest zwykle przez re-INVITE lub UPDATE z nagłówkami timera,
- b) **Session-Expires** – mówi po ilu sekundach od ostatniego udanego odświeżenia sesja ma zostać uznana za wygasłą, jeśli nie będzie kolejnego refresh,
- c) **Atrybut kierunku a** – opisuje kierunek przepływu mediów RTP dla danego strumienia np. m=video, może przyjmować takie wartości jak:
 - i. a=sendrecv, wysyła i odbiera RTP dla tego medium,
 - ii. a=sendonly, tylko wysyła RTP; nie oczekuje odbioru,
 - iii. a=recvonly, tylko odbiera RTP; nie wysyła,

- iv. a=inactive, brak mediów w żadną stronę (wstrzymanie połączenia).

Niektóre kroki testów dotyczą wysłania odpowiedniego kodu odpowiedzi, który jest krokiem testu a nie oczekiwanym rezultatem, wobec tego wymagane będzie narzędzie, za pomocą którego krok ten jest możliwy do wykonania.

Przypadki pozytywne

TC-REINVITE-001 – Zmiana parametrów SDP (kodek)

Kroki:

1. U1 zestawia połączenie z U2 (INVITE → 200 OK → ACK).
2. U1 wysyła re-INVITE z innym kodekiem w SDP.
3. U2 odpowiada 200 OK z zaakceptowanym SDP.
4. U1 wysyła ACK.

Oczekiwany rezultat:

- Parametry sesji zostają zaktualizowane.
- Media przełączają się na nowy kodek.

TC-REINVITE-002 – Zmiana adresu media w SDP

Kroki:

1. U1 zestawia połączenie z U2 (INVITE → 200 OK → ACK).
2. U1 wysyła re-INVITE z innym adresem IP
3. U2 odpowiada 200 OK z odpowiednim SDP.
4. U1 wysyła ACK.

Oczekiwany rezultat:

- Ścieżka mediów jest zgodna z nowym SDP.

TC-REINVITE-003 – Hold (sendonly / inactive)

Kroki:

1. U1 zestawia aktywne połączenie z U2.
2. U1 wysyła re-INVITE z SDP w trybie hold.

3. U2 odpowiada 200 OK z dopasowanym SDP.
4. U1 wysyła ACK.

Oczekiwany rezultat:

- Stan hold jest spójny po obu stronach.
- Zachowanie zgodne z polityką SBC (np. hold music).

TC-REINVITE-004 – Wznowienie po hold (sendrecv)**Kroki:**

1. Połączenie jest w stanie hold.
2. U1 wysyła re-INVITE z a=sendrecv (lub równoważnie).
3. U2 odpowiada 200 OK z a=sendrecv.
4. U1 wysyła ACK.

Oczekiwany rezultat:

- Dwukierunkowy strumień mediów zostaje przywrócony.
- Brak niespójnego SDP po stronie SBC.

TC-REINVITE-005 – Session Timer (re-INVITE jako refresh)**Kroki:**

1. Połączenie z negocjacją Session Timer.
2. Przed timeoutem strona refresher wysyła re-INVITE.
3. Druga strona odpowiada 200 OK z aktualnym Session-Expires / Refresher.
4. U1 wysyła ACK.

Oczekiwany rezultat:

- Sesja nie jest rozłączana przez timer.
- Wartości zgodne z polityką SBC

TC-REINVITE-006 – Dodanie strumienia (np. wideo do audio)**Kroki:**

1. Połączenie audio-only.
2. U1 wysyła re-INVITE z dodatkową linią m=video.

3. U2 akceptuje lub precyzuje media w 200 OK.
4. U1 wysyła ACK.

Oczekiwany rezultat:

- Negocjacja zgodna z odpowiedzią.
- SBC zachowuje poprawną strukturę SDP

TC-REINVITE-007 – re-INVITE przez SBC**Kroki:**

1. Połączenie U1-SBC-U2.
2. U1 wysyła re-INVITE do SBC.
3. SBC negocjuje re-INVITE w stronę U2 z odpowiednim SDP.
4. 200 OK i ACK na obu nogach.

Oczekiwany rezultat:

- Media po zaktualizowanych adresach

Testy negatywne

TC-REINVITE-008 – Odrzucenie zmiany SDP (488 Not Acceptable Here)**Kroki:**

1. Aktywne połączenie.
2. U1 wysyła re-INVITE z nieakceptowalnym SDP.
3. U2 odpowiada 488 (ew. Warning).
4. U1 wysyła ACK na 488.

Oczekiwany rezultat:

- Poprzedni stan mediów bez zmian.
- Dialog nie jest niepotrzebnie zrywany przez SBC.

TC-REINVITE-009 – re-INVITE zabroniony lub wymagający uwierzytelnienia**Kroki:**

1. Połączenie po uwierzytelnionym INVITE.
2. U1 wysyła re-INVITE.

Oczekiwany rezultat:

- Zachowanie zgodne z dokumentacją / polityką.
- Brak niespójności sygnalizacja OK / media zablokowane bez przyczyny.
- SBC stosuje politykę: odrzucenie (np. 403) lub challenge (np. 407) zgodnie z konfiguracją.

TC-REINVITE-010 – re-INVITE w nieistniejącym lub błędnym dialogu (481 Call/Transaction Does Not Exist)**Kroki:**

1. U1 zestawia i zrywa połączenie (BYE).
2. U1 wysyła re-INVITE w ramach „martwego” lub niepasującego dialogu.

Oczekiwany rezultat:

- Brak otwarcia nowej sesji mediów ani „przyklejenia” re-INVITE do innego połączenia.
- SBC lub U2 odpowiada 481.

TC-REINVITE-011 – Drugi re-INVITE w trakcie pierwszego (491 Request Pending)**Kroki:**

1. Aktywne połączenie.
2. U1 wysyła re-INVITE (transakcja nie zakończona jeszcze brakiem finalnej odpowiedzi / ACK w zależności od scenariusza).
3. Zanim zamknie się pierwsza transakcja, U1 (lub druga strona) wysyła kolejny re-INVITE w tym samym dialogu.

Oczekiwany rezultat:

- Brak uszkodzenia stanu dialogu ani „podwójnego” SDP.
- Odpowiedź zgodna z implementacją: np. **491 Request Pending** na drugi re-INVITE lub równoważna obsługa kolizji.

TC-REINVITE-012 – Session Timer odrzucony przez SBC (422 Session Interval Too Small)

Kroki:

1. Aktywne połączenie z negocjacją Session Timer.
2. U1 wysyła re-INVITE z Session-Expires **poniżej** wartości wymaganej przez politykę SBC
3. SBC odpowiada **422 Session Interval Too Small** (z nagłówkiem Min-SE zgodnym z konfiguracją).

Oczekiwany rezultat:

- Sesja mediów pozostaje w poprzednim stanie do czasu poprawnego re-INVITE z akceptowalnym Session-Expires.
- Wartość Min-SE w odpowiedzi jest zgodna z dokumentacją SBC.

Testy brzegowe

TC-REINVITE-013 – równoległe re-INVITE z obu stron

Kroki:

1. Aktywne połączenie.
2. U1 i U2 niemal jednocześnie wysyłają re-INVITE.
3. Rozstrzygnięcie kolizji zgodnie z RFC 3261
4. Dokończenie transakcji i ewentualne ponowienie drugiej strony.

Oczekiwany rezultat:

- Końcowy stan SDP spójny; brak zawieszenia sygnalizacji.
- SBC zachowuje spójność przestanych parametrów

TC-REINVITE-014 – Dwa re-INVITE z rzędu w bardzo krótkim czasie (stabilność stanu SDP)

Kroki:

1. Aktywne połączenie.

2. U1 wysyła re-INVITE ze zmianą SDP (np. kodek A); po pełnym zamknięciu transakcji (200 OK + ACK) wysyła **drugi** re-INVITE ze zmianą SDP (np. kodek B) w odstępie znacznie krótszym niż typowy interwał użytkownika (np. < 1 s), zgodnie z definicją w planie testów.
3. U2 odpowiada 200 OK na drugi re-INVITE.
4. U1 wysyła ACK do U2.

Oczekiwany rezultat:

- Końcowy stan mediów odpowiada **ostatniej** zaakceptowanej negocjacji.
- Brak „zatrzymania się” na pośrednim SDP po stronie SBC.

TC-REINVITE-015 – Duży / rozbudowany SDP w re-INVITE (wiele kodeków, atrybutów)**Kroki:**

1. Aktywne połączenie.
2. U1 wysyła re-INVITE z **rozbudowanym** SDP (wiele wpisów a=rtptime / a=fmtp / kandydujących linii m=, zgodnie z profilem testowym).
3. U2 odpowiada 200 OK z sensownym zawężeniem do obsługiwanych formatów.
4. U1 wysyła ACK do U2.

Oczekiwany rezultat:

- SBC przepuszcza lub poprawnie normalizuje komunikat bez obciążenia krytycznych części SDP (o ile mieści się w limitach produktu).
- Negocjacja kończy się spójnym RTP; brak losowych 4xx/5xx wyłącznie z powodu rozmiaru SDP, jeśli dokumentacja dopuszcza taki rozmiar.

TC-REINVITE-016 – re-INVITE z identycznym SDP**Kroki:**

1. Aktywne połączenie.
2. U1 wysyła re-INVITE z SDP identycznym jak po ostatniej negocjacji.

3. U2 odpowiada 200 OK.
4. U1 wysyła ACK do U2.

Oczekiwany rezultat:

- Sesja stabilna; brak zbędnego przerywania mediów.
- SBC nie wprowadza przypadkowych zmian SDP.

3.4 Wiadomości w dialogu

Wiadomości w dialogu to kolejne żądania i odpowiedzi SIP, które nie zaczynają nowej rozmowy od zera, a należą do istniejącej już sesji: np. ACK, BYE, re-INVITE, UPDATE, INFO, PRACK, REFER, a także MESSAGE czy SUBSCRIBE/NOTIFY, jeśli są prowadzone w ramach powiązanej relacji (tak samo Call-ID oraz na ogół te same tagi). Wspomniane wyżej wiadomości nie stanowią pierwszego kontaktu. Przypadki testowe będą zawierały takie atrybuty jak:

- a) **RAck** – nagłówek, który wiąże PRACK z konkretną odpowiedzią wstępną: zawiera m.in. numer odpowiedzi (status code),
- b) **RSeq** – numer sekwencyjny tej konkretnej odpowiedzi wstępnej. Każda kolejna wstępna odpowiedź w tej samej transakcji INVITE może mieć wyższy RSeq,
- c) **100rel** – opcja która jest niezbędna do wywołania wiadomości PRACK.

Ciekawy obiekt testów stanowi wiadomość SUBSCRIBE. Otóż w tej wiadomości, istnieje dodatkowy nagłówek **Event**, który odpowiada na pytanie jaki typ zdarzeń ma być zgłaszany w późniejszych NOTIFY. Najważniejsze typy eventów to:

- a) **dialog** – zmiany stanu dialogu,
- b) **message-summary** – skrzynka głosowa / MWI,
- c) **presence** – obecność,
- d) **reg** – rejestracja.

Framework testowy musi mieć możliwość dodawania SUBSCRIBE oraz Event w parze.

Testy pozytywne

TC-DLG-001 – PRACK w dialogu

Kroki:

1. U1 wysyła INVITE; U2 zwraca odpowiedź wstępną z Require: 100rel i RSeq.
2. U1 wysyła PRACK z poprawnym RACK.
3. U2 odpowiada 200 OK PRACK.

Oczekiwany rezultat:

- PRACK i 200 OK PRACK są w tym samym dialogu co INVITE.
- Sekwencja RSeq/RACK jest poprawna.

TC-DLG-002 – REINVITE w istniejącym dialogu

Kroki:

1. U1 i U2 mają aktywną sesję po INVITE/ACK.
2. U1 wysyła RE-INVITE w tym samym dialogu (nowa lub zmieniona oferta SDP – np. hold / kodek, wg profilu).
3. U2 odpowiada 200 OK RE-INVITE z SDP (lub inna odpowiedź wg scenariusza).
4. U1 wysyła ACK, jeśli model tego wymaga.

Oczekiwany rezultat:

- Ten sam Call-ID; rosnący CSeq dla INVITE.
- Stan sesji odzwierciedla uzgodnioną zmianę.

TC-DLG-003 – UPDATE w istniejącym dialogu

Kroki:

1. U1 i U2 mają aktywną sesję.
2. U1 wysyła UPDATE (z SDP lub bez – wg profilu).

3. U2 odpowiada 200 OK UPDATE (lub kod błędu w teście negatywnym).

Oczekiwany rezultat:

- UPDATE w ramach istniejącego dialogu (nagłówki spójne).
- Odpowiedź zgodna z żądaniem i polityką SBC.

TC-DLG-004 – INFO w istniejącym dialogu

Kroki:

1. U1 i U2 mają aktywną sesję.
2. U1 wysyła INFO z treścią wg profilu (typ zawartości / DTMF itd.).
3. U2 odpowiada 200 OK INFO (lub odrzuca uzgodnionym kodem).

Oczekiwany rezultat:

- INFO nie przerywa dialogu głosowego.
- Odpowiedź i propagacja treści zgodne z wymaganiami.

TC-DLG-005 – OPTIONS w kontekście sesji

Kroki:

1. U1 i U2 mają ustanowioną sesję (lub przygotowany adres drugiej strony).
2. U1 wysyła OPTIONS do URI powiązanego z drugą stroną (wg klienta).
3. U2 lub sieć odpowiada 200 OK OPTIONS z Allow/Supported wg profilu.

Oczekiwany rezultat:

- 200 OK z oczekiwanym zestawem metod.
- Trwająca rozmowa nie jest bez powodu rozłączana (jeśli OPTIONS jest dodatkowe).

TC-DLG-006 – SUBSCRIBE (pakiet zdarzeń)

Kroki:

1. U1 i U2 mają aktywną sesję lub inne warunki wstępne pakietu Event.
2. U1 wysyła SUBSCRIBE z Event i Expires (np. dialog / call-info – wg profilu).
3. U2 odpowiada 200 OK SUBSCRIBE.
4. U2 wysyła NOTIFY ze Subscription-State.

Oczekiwany rezultat:

- Subskrypcja ustanowiona (lub odrzucona w teście negatywnym).
- NOTIFY spójne z pakietem i stanem rozmowy.

TC-DLG-007 – NOTIFY w ramach subskrypcji**Kroki:**

1. Istnieje aktywna subskrypcja po SUBSCRIBE.
2. Strona notyfikująca wysyła NOTIFY ze zmianą stanu.
3. Odbiorca odpowiada 200 OK NOTIFY.

Oczekiwany rezultat:

- NOTIFY: poprawne Event, Subscription-State i treść pakietu.

TC-DLG-008 – MESSAGE (pager) przy aktywnej rozmowie**Kroki:**

1. Opcjonalnie U1 i U2 mają równoległą sesję głosową.
2. U1 wysyła MESSAGE do U2 z treścią (Content-Type wg profilu).
3. U2 odpowiada 200 OK MESSAGE.

Oczekiwany rezultat:

- MESSAGE dostarczone; 200 OK.
- Sesja głosowa nie jest rozłączana przez samo MESSAGE

TC-DLG-009 – REFER: przekierowanie połączenia (blind transfer) z NOTIFY**Kroki:**

1. U1 i U2 mają aktywną rozmowę głosową po ACK.
2. U1 wysyła REFER z Refer-To wskazującym URI strony U3 (przekierowanie bez udziału U1 w nowej nodze – model blind zgodny z profilem).
3. U2 odpowiada 202 Accepted lub 200 OK zgodnie z produktem.
4. U2 realizuje zestawienie nogi do U3 zgodnie z polityką i wysyła do U1 NOTIFY z pakietem refer (np. trying, potem success).
5. U1 na każdy NOTIFY odpowiada 200 OK NOTIFY.

Oczekiwany rezultat:

- Połączenie zostaje przekierowane zgodnie z Refer-To; U1 otrzymuje informację o stanie w NOTIFY.
- Zakończenie rozmowy pierwotnej U1-U2 zgodne z wymaganiami

Testy negatywne**TC-DLG-010 – UPDATE bez odpowiedzi w wymaganym czasie****Kroki:**

1. U1 oraz U2 prowadzą aktywną rozmowę po ACK.
2. U1 wysyła UPDATE; U2 nie wysyła ostatecznej odpowiedzi w czasie określonym w teście (symulacja braku odpowiedzi).

Oczekiwany rezultat:

- Zachowanie zgodne z polityką produktu (retransmisje, timeout, ewentualne rozłączenie).
- Wynik końcowy sesji zgodny z wymaganiami

TC-DLG-011 – INFO odrzucone, potem INFO zaakceptowane**Kroki:**

1. U1 oraz U2 prowadzą aktywną rozmowę.
2. U1 wysyła INFO z nieobsługiwany Content-Type; U2 odpowiada kodem błędu z profilu.

Oczekiwany rezultat:

- Pierwsze INFO kończy się błędem zgodnym z profilem.

TC-DLG-012 – REFER odrzucony (403)**Kroki:**

1. U1 i U2 mają aktywną rozmowę.
2. U1 wysyła REFER; U2 odpowiada 403 Forbidden zgodnie z polityką zabraniającą transferu.

Oczekiwany rezultat:

- Transfer nie jest realizowany.
- Rozmowa pierwotna może trwać dalej, jeśli scenariusz tak zakłada.

TC-DLG-013 – SUBSCRIBE z nieobsługiwanym Event, potem SUBSCRIBE z obsługiwanym atrybutem Event**Kroki:**

1. U1 i U2 mają aktywną rozmowę.
2. U1 wysyła SUBSCRIBE z atrybutem Event nieobsługiwanym przez U2.
3. U2 odpowiada 489 Bad Event lub 403.

Oczekiwany rezultat:

- Subskrypcja jest odrzucona zgodnie z kodem.

TC-DLG-014 – MESSAGE z błędem, potem poprawne MESSAGE**Kroki:**

1. U1 i U2 mają opcjonalnie równoległą rozmowę głosową, jeśli scenariusz to przewiduje.
2. U1 wysyła MESSAGE na cel błędny lub niedostępny; sieć lub U2 zwraca 404 lub 480 zgodnie z profilem.

Oczekiwany rezultat:

- Pierwsze MESSAGE kończy się uzgodnionym błędem.

Testy brzegowe**TC-DLG-015 – Jednoczesna próba UPDATE i REINVITE z obu stron****Kroki:**

1. U1 i U2 mają aktywną rozmowę.
2. W krótkim odstępie czasu U1 wysyła UPDATE, a U2 wysyła RE-INVITE
3. Strony wymieniają odpowiedzi zgodnie z zachowaniem endpointów

Oczekiwany rezultat:

- Brak zakleszczenia (deadlocka); zachowanie zgodne z RFC i profilem produktu.
- Po ustabilizowaniu stan SDP jest spójny.

TC-DLG-016 – NOTIFY z Subscription-State terminated tuż po SUBSCRIBE**Kroki:**

1. U1 i U2 mają aktywną rozmowę.
2. U1 wysyła SUBSCRIBE z krótkim Expires; U2 odpowiada 200 OK SUBSCRIBE.
3. U2 natychmiast wysyła NOTIFY z Subscription-State: terminated; U1 odpowiada 200 OK NOTIFY.

Oczekiwany rezultat:

- Subskrypcja kończy się zgodnie z NOTIFY; brak niespójnych stanów.
- Dialog głosowy nie jest naruszony.

TC-DLG-017 – OPTIONS z rozbudowanymi nagłówkami przy granicy MTU**Kroki:**

1. U1 i U2 mają aktywną rozmowę.
2. U1 wysyła OPTIONS z bardzo rozbudowanymi nagłówkami zbliżonymi do limitu MTU dla UDP.
3. U2 lub sieć odpowiada zgodnie z profilem testu (200 OK OPTIONS lub inna zdefiniowana reakcja).

Oczekiwany rezultat:

- Dialog głosowy pozostaje stabilny przy oczekiwaniu pozytywnym.

TC-DLG-018 – PRACK z błędnym RACK, potem poprawny PRACK**Kroki:**

1. U1 inicjuje zestawienie sesji (INVITE); U2 odpowiada zgodnie z modelem (np. 100 Trying).
2. U2 wysyła odpowiedź wstępną wymagającą potwierdzenia (np. 183 Session Progress lub 180 Ringing) z nagłówkiem Require:

100rel i RSeq ustawionym na wartość N (zapisać N oraz CSeq związany z transakcją INVITE dla późniejszej weryfikacji).

3. U1 celowo wysyła PRACK, w którym nagłówek RACK niepoprawny
4. U2 odpowiada odrzuceniem, typowo 481 Call Leg Does Not Exist lub 400 Bad Request
5. U1 wysyła drugi PRACK z poprawnym Rack.

Oczekiwany rezultat:

- PRACK z błędnym RACK jest odrzucony odpowiedzią niebędącą 2xx (kod zgodny z implementacją, zapisany w teście).
- PRACK z poprawnym RACK otrzymuje 200 OK PRACK.

3.5 Testowanie kombinacji wiadomości SIP

W rozdziale tym zaprezentowane zostaną przykłady testów, które ilustrują różnorodne kombinacje wiadomości SIP w ramach pojedynczego dialogu. Pokazują one propozycje weryfikacji spójności sygnalizacji oraz współdziałania poszczególnych mechanizmów. Przedstawione scenariusze nie wyczerpują wszystkich możliwych przypadków. W rozdziale tym, nie zostanie dokonany podział na testy pozytywne, negatywne czy brzegowe, ponieważ strategia ta została już zaprezentowana.

TC-MIX-001- SUBSCRIBE, OPTIONS, MESSAGE, UPDATE, 2 razy NOTIFY

Kroki:

1. U1 i U2 prowadzą aktywną rozmowę po ACK
2. U1 wysyła SUBSCRIBE z nagłówkiem Event ustawionym na "abc".
3. U2 odpowiada 200 OK SUBSCRIBE.
4. U2 wysyła pierwszy raz NOTIFY z nagłówkiem Event ustawionym na "abc".
5. U1 odpowiada 200 OK NOTIFY.
6. U1 wysyła OPTIONS.
7. U2 odpowiada 200 OK OPTIONS.

8. U1 wysyła MESSAGE.
9. U2 odpowiada 200 OK MESSAGE.
10. U1 wysyła UPDATE.
11. U2 odpowiada 200 OK UPDATE.
12. U2 wysyła drugi raz NOTIFY z tym samym nagłówkiem Event co w kroku 2.
13. U1 odpowiada 200 OK NOTIFY.

Oczekiwany rezultat:

- SUBSCRIBE oraz wiadomości NOTIFY są spójne z profilem pakietu.
- OPTIONS, MESSAGE, UPDATE kończą się 2xx.
- dialog głosowy pozostaje stabilny.
- po kroku 12 zmienia się nagłówek Subscription-State (np. nowa wartość Expires).

TC-MIX-002 – PRACK (dwie odpowiedzi wstępne), ACK, INFO, MESSAGE, OPTIONS, UPDATE**Kroki:**

1. U1 wysyła INVITE z SDP i wsparciem 100rel.
2. U2 odpowiada 183 Session Progress z Require: 100rel.
3. U1 wysyła PRACK z poprawnym nagłówkiem RACK.
4. U2 odpowiada 200 OK PRACK.
5. U2 odpowiada 180 Ringing z Require: 100rel i kolejnym numerem RSeq.
6. U1 wysyła PRACK z poprawnym nagłówkiem RACK.
7. U2 odpowiada 200 OK PRACK.
8. U2 odpowiada 200 OK INVITE z SDP.
9. U1 wysyła ACK.
10. U1 wysyła INFO z treścią zgodną z profilem.
11. U2 odpowiada 200 OK INFO.
12. U1 wysyła MESSAGE.
13. U2 odpowiada 200 OK MESSAGE.

14. U1 wysyła OPTIONS.
15. U2 odpowiada 200 OK OPTIONS.
16. U1 wysyła UPDATE.
17. U2 odpowiada 200 OK UPDATE.

Oczekiwany rezultat:

- Pierwszy PRACK jest poprawnie sparowany z pierwszym RSeq.
- Drugi PRACK jest poprawnie sparowany z drugim RSeq.
- Sesja zostaje zestawiona (200 OK INVITE).
- ACK zostaje wysłany po 200 OK INVITE.
- INFO kończy się odpowiedzią 2xx.
- MESSAGE kończy się odpowiedzią 2xx.
- OPTIONS kończy się odpowiedzią 2xx.
- UPDATE kończy się odpowiedzią 2xx.
- Dialog głosowy pozostaje stabilny.

TC-MIX-003 – INFO, MESSAGE, UPDATE, RE-INVITE, OPTIONS**Kroki:**

1. U1 oraz U2 prowadzą aktywną rozmowę po ACK.
2. U1 wysyła INFO z treścią zgodną z profilem.
3. U2 odpowiada 200 OK INFO.
4. U1 wysyła MESSAGE.
5. U2 odpowiada 200 OK MESSAGE.
6. U1 wysyła UPDATE.
7. U2 odpowiada 200 OK UPDATE.
8. U1 wysyła RE-INVITE z SDP.
9. U2 odpowiada 200 OK na RE-INVITE z SDP.
10. U1 wysyła ACK.
11. U1 wysyła OPTIONS.
12. U2 odpowiada 200 OK OPTIONS.

Oczekiwany rezultat:

- INFO kończy się odpowiedzią 2xx.

- MESSAGE kończy się odpowiedzią 2xx.
- UPDATE kończy się odpowiedzią 2xx.
- RE-INVITE kończy się odpowiedzią 2xx.
- ACK zostaje wysłany po 200 OK na RE-INVITE.
- OPTIONS kończy się odpowiedzią 2xx.
- Stan SDP jest zgodny z ostatnią uzgodnioną wymianą po RE-INVITE.
- Dialog głosowy pozostaje stabilny.

TC-MIX-004 – PRACK, ACK, SUBSCRIBE, NOTIFY, MESSAGE, REFER, NOTIFY *2

Kroki:

1. U1 wysyła INVITE z 100rel (wg profilu).
2. U2 odpowiada 183 Session Progress z Require: 100rel i RSeq.
3. U1 wysyła PRACK z poprawnym nagłówkiem RACK.
4. U2 odpowiada 200 OK PRACK.
5. U2 odpowiada 200 OK INVITE z SDP.
6. U1 wysyła ACK.
7. U1 wysyła SUBSCRIBE z nagłówkiem Event ustawionym na „abc”.
8. U2 odpowiada 200 OK SUBSCRIBE.
9. U2 wysyła NOTIFY z nagłówkiem Event ustawionym na „abc”.
10. U1 odpowiada 200 OK NOTIFY.
11. U1 wysyła MESSAGE.
12. U2 odpowiada 200 OK MESSAGE.
13. U1 wysyła REFER z Refer-To wskazującym URI strony U3.
14. U2 odpowiada 202 Accepted (lub 200 OK – wg produktu).
15. U2 wysyła NOTIFY z Event: refer.
16. U1 odpowiada 200 OK NOTIFY.
17. U2 wysyła drugi NOTIFY z Event: refer ze zaktualizowanym stanem transferu.
18. U1 odpowiada 200 OK NOTIFY.

Oczekiwany rezultat:

- PRACK jest poprawnie sparowany z RSeq z kroku 2.

- Sesja zostaje zestawiona (200 OK INVITE).
- ACK zostaje wysłany po 200 OK INVITE.
- SUBSCRIBE jest zgodny z profilem pakietu „abc”.
- NOTIFY z Event: „abc” jest zgodny z profilem pakietu.
- MESSAGE kończy się odpowiedzią 2xx.
- REFER otrzymuje odpowiedź zgodną z profilem (202 lub 200).
- Pierwszy NOTIFY (refer) jest spójny z profilem pakietu refer.
- Drugi NOTIFY (refer) jest spójny z profilem pakietu refer.
- Dialog głosowy pozostaje stabilny do momentu zakończenia.

4. Podsumowanie

W niniejszej publikacji przedstawiona została koncepcja testowania komunikacji w protokole SIP, który jest bardzo ważnym składnikiem współczesnych systemów VoIP oraz architektur takich jak IMS Core. Przedstawione zostały podstawowe zagadnienia związane z działaniem SIP, jego rolą w zestawianiu oraz obsłudze połączeń, a także miejsce protokołu w rozwiązaniach typu SBC. Kluczowe było tu spojrzenie na SIP nie tylko jako na zbiór wiadomości, ale jako na mechanizm sterujący całym przebiegiem sesji – od rejestracji użytkownika, przez zestawienie połączenia, aż po jego modyfikację i zakończenie.

Głównym celem artykułu było zaproponowanie podejścia do testowania opartego na analizie sygnalizacji oraz logicznych przepływów wiadomości SIP. Podział testów na scenariusze pozytywne, negatywne i brzegowe pozwala w uporządkowany sposób pokryć zarówno typowe przypadki użycia, jak i sytuacje błędne oraz graniczne, które w praktyce często decydują o stabilności systemu. Zaproponowane zostały podejścia, jakie można zastosować dla kluczowych wiadomości SIP, takich jak REGISTER, INVITE czy re-INVITE, a także dla komunikacji w ramach istniejącego dialogu. Istotne było również uwzględnienie mechanizmów takich jak SDP, które bezpośrednio wpływają na przebieg oraz poprawność sesji.

Przyjęta koncepcja zakłada skupienie się na warstwie sygnalizacyjnej, co pozwala testować zachowanie systemu bez konieczności pełnej obsługi mediów. Dzięki temu, możliwe jest uproszczenie środowiska testowego, przy jednoczesnym zachowaniu wysokiej wartości testów. Ostatecznie zaproponowane podejście może być rozwijane w zależności od potrzeb konkretnego rozwiązania oraz poziomu jego złożoności.

Autor:

Marek Żukowicz jest absolwentem matematyki na Uniwersytecie Rzeszowskim. Obecnie pracuje jako SBC Senior Software Test Engineer w Nokii. Jego zainteresowania skupiają się wokół testowania oraz jakości oprogramowania, procesów i koncepcji związanych z wytwarzaniem oprogramowania, zastosowania algorytmów ewolucyjnych oraz zastosowania matematyki w procesie testowania. Interesuje się również muzyką.

Recenzenci:

Marcin Misiorny

Bibliografia:

- [1] https://pl.wikipedia.org/wiki/Session_Initiation_Protocol
- [2] https://en.wikipedia.org/wiki/Session_border_controller
- [3] <https://datatracker.ietf.org/>
- [4] <https://sipp.readthedocs.io/en/v3.6.1/transport.html>
- [5] https://en.wikipedia.org/wiki/IP_Multimedia_Subsystem
- [6] <https://www.3cx.pl/voip-sip/sip-methods/>
- [7] https://www.voipmonitor.org/doc/Understanding_the_SIP_Protocol
- [8] <https://datatracker.ietf.org/doc/html/rfc3261>
- [9] <https://datatracker.ietf.org/doc/html/rfc3515>
- [10] <https://datatracker.ietf.org/doc/rfc3263/>
- [11] <https://datatracker.ietf.org/doc/html/rfc3311>
- [12] <https://datatracker.ietf.org/doc/html/rfc3262>