Cloud PBX
Solutions

Cloud PBX Core

Documentation

2011-2023

THE SPECIFICATIONS AND INFORMATION REGARDING THE PRODUCTS IN THIS MANUAL ARE SUBJECT TO CHANGE WITHOUT NOTICE. ALL STATEMENTS, INFORMATION, AND RECOMMENDATIONS IN THIS MANUAL ARE BELIEVED TO BE ACCURATE BUT ARE PRESENTED WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED. USERS MUST TAKE FULL RESPONSIBILITY FOR THEIR APPLICATION OF ANY PRODUCTS.

THE SOFTWARE LICENSE AND LIMITED WARRANTY FOR THE ACCOMPANYING PRODUCT ARE SET FORTH IN THE INFORMATION PACKET THAT SHIPPED WITH THE PRODUCT AND ARE INCORPORATED HEREIN BY THIS REFERENCE. IF YOU ARE UNABLE TO LOCATE THE SOFTWARE LICENSE OR LIMITED WARRANTY, CONTACT THE SELLER REPRESENTATIVE FOR A COPY.

Contents

Cloud PBX Core

Document Purpose

This document is intended for engineers and architects of companies, including systems integrators and communication service providers. It provides information about the fundamental principles of the platform's operation and the necessary details for making decisions regarding the creation, development, and maintenance of systems and applications based on this platform.

Document Content

This document offers a high-level description of the platform's architecture.

In the "Typical Use Cases" section, you will find the main usage scenarios of the platform within hardware-software complexes and software solutions.

The "Architecture" section elaborates on the platform's components and their interaction principles at the network and protocol levels.

The "Performance and Scalability" section contains information about hardware requirements and the principles for scaling systems based on this platform.

The "Reliability" section lists the methods and principles that ensure the availability and fault tolerance of the platform's software components. It also includes requirements for hardware components and recommendations for system deployment.

In the "Compatibility" section, you will find a list of supported protocols, hardware platforms, operating systems, virtualization environments, and information about SORM support.

For a detailed description of software components, implementation details, and specific technical solutions, please refer to Document 2.

Definitions, Abbreviations, and Acronyms

- **Back End**: The core component of the platform responsible for data storage.

- **Domain**: A data model category corresponding to an organization. A domain contains information about an organization and consolidates user accounts. Domains can form a hierarchy.

- **Task**: A lightweight process for handling external requests to the system, such as a phone call or an HTTP request.

- **Installation**: A hardware-software complex that ensures the system's operation—a set of physical or virtual servers where the necessary components of the platform are deployed and services are configured.

- **Communication Core**: The system's core service, providing protocol support and a runtime environment for applications.

- **Component**: The fundamental unit of the platform. One or more components form services.

- **Media Processor**: A core platform component processing useful media payloads.

- **Operator**: A mobile or fixed-line operator.

- **Organization**: A structural unit that unites system users, such as a company, branch, department, etc.

- **Platform**: A software package that provides a set of software components for building call processing systems.

- **User**: A system user, which could be an individual, an employee of an organization, etc.

- **Application**: A software module designed for implementing the system's business logic—an aggregate of resources (media files, images) and executable code in the ECMAScript language containing task start points.

- **Service**: A logical unit of the system that performs a specific function.

- **System**: A hardware-software complex for call processing—an installation with deployed services and installed applications.

- **Account**: A data model category corresponding to a user. User information is stored in an account.

- **Front End**: A core platform component that handles incoming requests and ensures the execution of applications.

Typical Use Cases

Functionality

The platform is developed based on the concept of NGN and adheres to the architecture of a Multimedia IP System (IMS, 3GPP TS 24.229 specifications). It also follows the architecture of an Intelligent Network (IN, 3GPP TS 29.078 specifications).

The elements and functions provided by the platform include:

- SIP Application Servers (SIP AS)

- Call Session Control (P/I/S-CSCF)

- Multimedia Resource Handling (MRFC, MRFP)

- Home Subscriber Server (HSS) with a limited subset of interfaces

- Service Control Point (SCP)

- WebRTC Gateway

- STUN/ICE Server

- Web server supporting HTTP and WebSocket protocols.

Telecommunication System Organization

Installations created on this platform can be used as a compact IMS core or integrated into the operator's IMS network.

Telecommunication Operator Network Component

Integration with IM-SSF, SGW, MGCF, and MGW nodes from third-party manufacturers is required to connect an installation to traditional telephone networks.

Mobile Telecommunication Operator Network Component

Enterprise PBX

Business Logic Implementation

The business logic of systems built on this platform is realized through applications. The platform provides a runtime environment and mechanisms for deploying and managing application versions.

The cloud PBX application has been developed to create and use a PBX as part of a communication operator's IMS network. The application allows integrating a customer's installation into the communication operator's IMS network and provides a user interface for managing platform and PBX settings. The application's user interface supports branding.

Examples of other systems and components developed and operated on this platform include:

- Enterprise PBX for large organizations with complex branch structures.

- Voicemail system for a mobile operator serving 100 million subscribers.

- Call recording system for an operator.

- WebSocket/WebRTC gateway for an operator.

- Intelligent Session Border Controller (SBC) for an operator.

Application

The system business logic is implemented through applications — software modules. The application has the following attributes:

- Identifier — a unique name within the installation.

- Load address — URI or file path on the hard disk.

- Version.

The application comprises resources (media files, images) and executable code in ECMAScript (ECMA-262 6th edition standard). The executable code contains entry points for tasks, which are invoked when external requests are received by the system, such as phone calls or HTTP requests.

Performance and Scalability

Performance

The required performance of the installation is calculated based on the number of subscribers, their profiles, and traffic volume. The platform's logic maintains the performance level of the installation if the hardware requirements are met.

The communication core algorithms ensure:

- For BE nodes — even distribution of created accounts using a weighted consistent hashing algorithm.
- For FE nodes — even load distribution using a consistent hashing algorithm.
- For MP nodes — caching frequently used media objects, as well as uniform distribution of media streams and reduced delays during their transmission.

The media channel's endpoints are allocated on a single MP node to reduce media stream transmission delays. If, during a call session, the endpoints are on different MP nodes, an intra-cluster bus is used for media stream transmission.

The platform supports grouping MP nodes into zones. Placing MP node groups close to users helps reduce network delays if the application implements a specific group selection algorithm when establishing or terminating calls.

Scalability

The platform supports vertical and horizontal scalability.

Vertical scalability of the installation is achieved by increasing the hardware power of individual cluster nodes.

Horizontal scalability of the installation is close to linear and is achieved by adding new nodes to the cluster. Adding nodes does not require service interruption.

The platform architecture supports the creation of loosely coupled clusters with static user binding to clusters and inter-cluster replication support, ensuring potentially unlimited installation scalability.

Hardware Requirements

The type of node determines hardware requirements and scaling factors.

Back End

The performance of BE nodes primarily depends on the amount of RAM (Random Access Memory) and disk write speed (write IOPS). For a typical cloud PBX service, approximately 4 GB + 0.2 Mb per subscriber, 0.8 IOPS per 100 subscribers, and 2 Cores + 1 Core per 10 000 subscribers are required.

Typical BE Configuration for 100 000 Subscribers

BE nodes support double and triple replication. The number of BE nodes in the installation increases proportionally to the replication coefficient.

CPU	Intel Xeon E5-2620, 12 Cores
RAM	32 GB
Disk	SSD 1 TB, 800 IOPS

Front End

The performance of FE nodes primarily depends on the processor (CPU) characteristics and the amount of the Random Access Memory (RAM). For a typical cloud PBX service, around 5 GB + 50 GB per 1000 concurrent calls, and about 2 Cores + 1 Core per 2.5 CAPS are required.

Typical Configuration of One FE for a 30 CAPS / 1500 CC Stream

CPU	Intel Xeon E5-2620, 12 Cores
RAM	128 GB
Disk	SSD 256 GB

Backup is not required for FE nodes.

Media Processor

The performance of MP nodes primarily depends on the processor (CPU) characteristics and the network interface card (NIC I/O) performance. For a typical cloud PBX service with 50% of media streams encoded in mp3, approximately 2 Cores + 1 Core / 40 CC and 160 kBps/120 pps / 1CC are required.

Typical Configuration of One MP for a 880 CC Stream

CPU	2 x Intel Xeon E5-2620, 12 Cores
RAM	16 GB
Disk	SSD 256 GB
NIC	2 x 8 tx/rx queues 1 Gbps NIC, 150 MBps / 110 kPPS

Backup is not required for MP nodes.

Performance Level Support

Equipment load depends on the subscriber profile; therefore, monitoring key metrics and expanding the installation when threshold values are exceeded is essential.

Threshold Values for Equipment Load

- For BE nodes: RAM usage — 75%, disk load — 75%.
- For FE nodes: RAM usage — 75%, CPU usage — 75%.
- For MP nodes: CPU usage — 75%, appearance of losses on interfaces.

For uninterrupted installation operation, it is recommended to maintain a 50% power reserve for each of the following indicators:

- Number of subscribers.
- Call volume capacity.
- Number of concurrent calls.

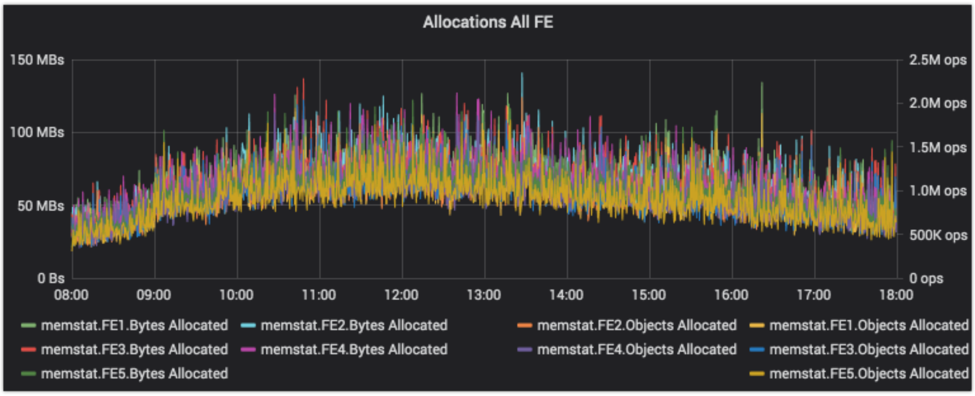
Scalability Evaluation for One Installation of Cloud PBX Core

Number of Concurrent Calls	BE	FE	MP
10000	4	8	14
8000	2	7	11
4000	2	4	6
1000	2	2	2
500	2	2	2
100	2	2	2

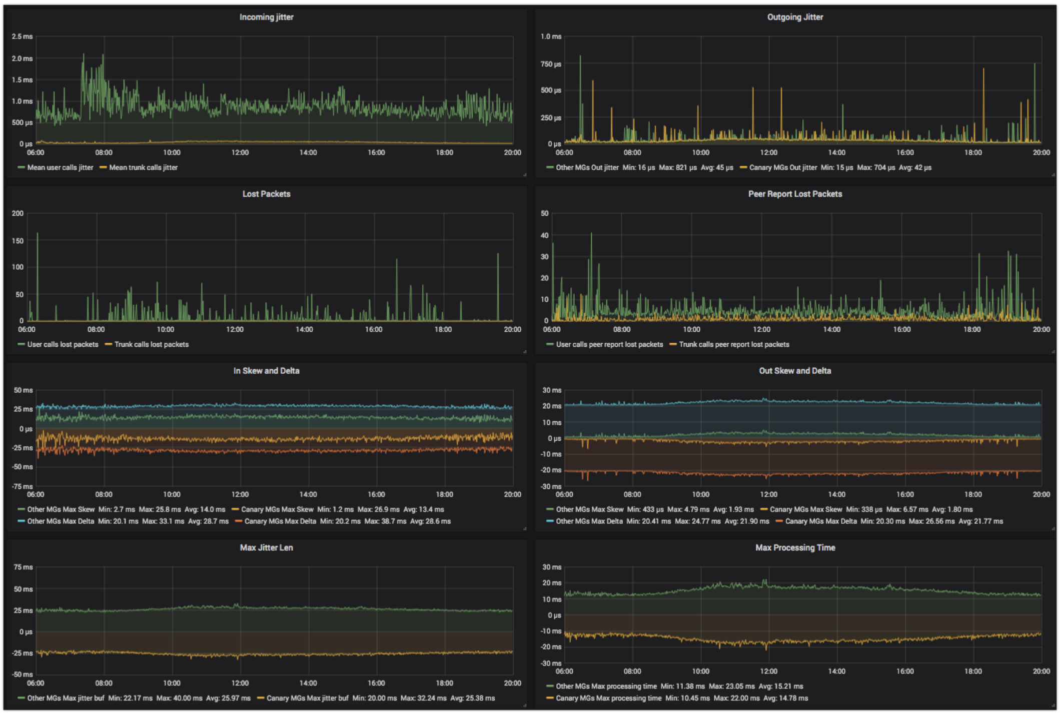
Monitoring and Statistics

All platform components support one or more of the following monitoring protocols:

- SNMPv1, SNMPv2, SNMPv3
- Golang expvars (JSON over HTTP)
- Cloud PBX Metrics Collector Protocol.



Example of displaying Golang Runtime Memory statistics



Example of displaying media stream statistics

Architecture

The platform consists of a set of components that provide services for creating call processing systems.

The main service, the Communication Core service, or Centrex, provides protocol support and an execution environment for applications.

Auxiliary services ensure the operation of the communication core and provide additional functions. The system requirements determine the set and composition of auxiliary services.

Communication Core

The Communication Core provides protocol support and a runtime environment for applications.

Data Model

The data model of the communication core describes three types of entities:

- Global settings
- Domain — corresponds to an organization
- Account — corresponds to a user.

Global Settings

Global settings contain:

- Installation settings
- Metadata about users and domains — information about which Back-End nodes store data for a specific user or domain.

Installation settings include:

- Settings for communication core components
- Rules for handling incoming requests — application, task, and parameters
- Application data — name and download address.

Storage of **metadata about users and domains** is implemented using a UserIndex object, which:

- Organizes namespaces, ensuring the uniqueness of domain and account names within the installation.
- Associates object names with Back-End nodes that store these objects, both directly and through a sequence of name references (forwarders or aliases).

Domain

A domain corresponds to an organization and stores:

- Domain data — a list of accounts and specification of an application identifier for processing requests to this domain and its accounts
- Domain settings — account groups, interface language, etc.

The system domains form a hierarchy, with the root node being the main or system domain.

Settings and data of child domains are accessible to parent domains for viewing and modification. Child domains can use some settings and data of parent domains. For example, if no application is specified for a child domain, the application explicitly defined for the nearest parent domain in the hierarchy will be used. If no application is specified for parent domains, the application defined for the main domain will be used.

This structure allows for managing domain visibility and organizing complex service provisioning schemes within a single system.

Account

An account corresponds to a user and stores:

- Account data (creation date and password)

- Account settings
- Temporary data — data that changes in real-time, such as dialog and registration states, call parameters, etc.

Data for domains and accounts are defined and managed by the communication core.

Settings for domains and accounts are defined and managed by the application.

Components

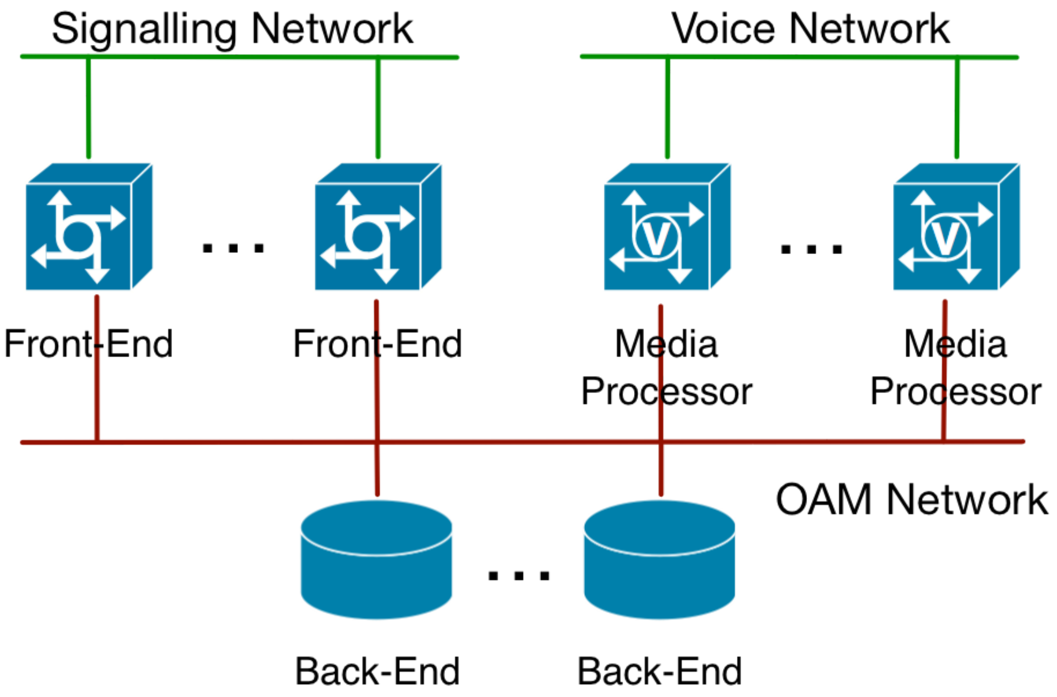
General Information

Components that make up the communication core service:

- Front End
- Back End
- Media Processor.

The Front End and Back End components are written in the Go (Golang) programming language, which ensures scalability and efficient utilization of computational resources for running parallel and concurrent applications.

The Media Processor component is written in the C++14 programming language. Its independent parallel pipeline architecture for media stream processing ensures maximum throughput and hardware utilization efficiency while minimizing latency.



The interconnection of nodes in the communication core of the Cloud PBX Solutions platform

Front End

The Front End component implements a mechanism for interacting with other core components and external systems. It also serves as an environment for running and executing applications.

The Front End service provides:

- Endpoints for receiving incoming requests from users and external systems using the following protocols:

  - SIP/UDP, SIP/TCP, SIPS
  - HTTP, HTTPS, WebSocket

Upon receiving an incoming request, the Front End initiates a task for processing it.

- Programmatic interfaces for applications to interact with external systems using protocols such as:

- SIP/UDP, SIP/TCP, SIPS
- HTTP, HTTPS
- RADIUS, DIAMETER
- LDAP
- SMTP.

- Programmatic interfaces for applications to work with other core components.

- Command-line interface (CLI) for basic system management.

- A set of functions and commands for accessing data from application code or CLI.

Back End

The Back End component implements a distributed "key-value" store with optimistic logging, storing data for domains and users.

The Back End service ensures storage and access to data in the following categories:

- Long-term information and metadata about domains and users.
- Real-time changing data, such as dialog and registration states, call parameters, etc.

The storage organization and access speed depend on the data category.

The Back End component interacts with other communication core components by:

- Providing domain and subscriber information to the Front End component and synchronization primitives for signaling tasks.
- Providing media files to the Media Processor component.

Media Processor

The Media Processor component implements mechanisms for handling useful media payload: receiving, sending, processing, and mixing media streams, as well as recording and playing media files.

The following protocols are supported for incoming traffic:

- RTP and RTCP
- WebRTC

The Media Processor component supports codecs:

- G.711 A-law and G.711 μ-law
- G.729
- WebRTC Opus
- AMR-NB/WB
- Speex
- PCM/WAV (mono/stereo 16 bit/8, 32, 48 kHz)
- MP3 (mono/stereo 16 bit/32 kHz).

The Media Processor service interacts with the Front End service by:

1. Allowing the allocation of media channel endpoints.
2. Providing the capability to process media streams, offering a programmatic interface to create media objects (transcoders, DTMF and voice activity detectors, recorders, mixers, etc.) and organize a pipeline from them.

Centrex

General Information

The Centrex service, or the Communication Core service, is provided by an elastic high-availability cluster.

Types of cluster nodes:

- Front End (centrex-frontend)
- Back End (centrex-backend)
- Media Processor (centrex-mediaprocessor).

Front-End and Media Processor nodes do not maintain their state.

Back-End nodes store the core data and configurations.

Cluster Node Interaction

Cluster nodes interact via the HTTP protocol and proprietary protocols:

- Cluster Discovery
- MP Discovery
- Inter-cluster RPC
- Inter-cluster RTP.

The **HTTP** protocol is used for file transfer:

- From a Back-End node to a Front-End or Media Processor node, followed by user delivery.
- From a Front-End or Media Processor node to a Back-End node for storage.

Back-End and Front-End nodes use the Cluster Discovery protocol to notify the inclusion and exclusion from the cluster mutually.

The **MP Discovery** protocol is used by Front-End and Media Processor nodes to mutually notify the inclusion and exclusion from the cluster and the current load.

Inter-cluster RPC, a TCP-based protocol, is used by nodes:

- Back Ends for data synchronization.
- Front Ends to query Back Ends for user information.
- Front Ends for exchanging messages between signaling tasks.
- Front Ends and Media Processors to allocate media channel endpoints.

Inter-cluster RTP, a UDP-based protocol, is used by media processor nodes to switch media streams.

Performance

Communication Core algorithms provide:

- For Back-End nodes, even distribution of created accounts using a weighted consistent hashing algorithm.
- For Front-End nodes, even load distribution using a consistent hashing algorithm.
- For Media Processor nodes, caching frequently used media objects, even distribution of media streams and transmission delays reduction.

Media channel endpoints are allocated to a single Media Processor to minimize transmission delays of media streams. If endpoints are on different nodes during a call session, an intra-cluster bus is used for media stream transmission.

The platform supports the grouping of Media Processors by zones. Placing groups of Media Processors near users reduces network delays if the application implements a specific node selection algorithm when establishing or ending a call.

Fault Tolerance

The Cluster Controller and Orchestrator provide cluster fault tolerance if the system configuration includes one.

The **Cluster Controller** is a Back-End node that is automatically selected and monitors the state of cluster nodes.

The **Orchestrator** also monitors the state of cluster nodes and manages the cluster's configuration, including introducing nodes into operation and removing them from the cluster, ensuring the necessary number of node instances.

In the event of a software failure of a Front-End or Media Processor node or a hardware failure of the server hosting such a node, the node is removed from the cluster, and the load is automatically redistributed among available nodes of the corresponding type.

A Back-End node unavailable for 1500 ms is automatically removed from the cluster. If this node is the cluster controller, a new controller is selected. The orchestrator initiates the launch of the Back-End node replica.

To ensure the fault tolerance of the Centrex service, it is necessary to establish a distributed storage system.

Auxiliary Services

The set and composition of solutions and components that ensure the operation of auxiliary services are determined by system requirements. For example, to operate a cloud PBX, the following are additionally required:

1. Call Recording Storage.
2. Call History — uses the queue service and a DBMS.
3. Pre-billing — uses a DBMS.

Some services require the Orchestrator for their operation.

Call History

Call History is a service for keeping call statistics and call history records.

The centrex-history and history-rest components provide the service. The centrex-history component is responsible for writing data to the database, while the history-rest component is responsible for reading data from the database.

To operate the Call History service, a queue service and a DBMS are required.

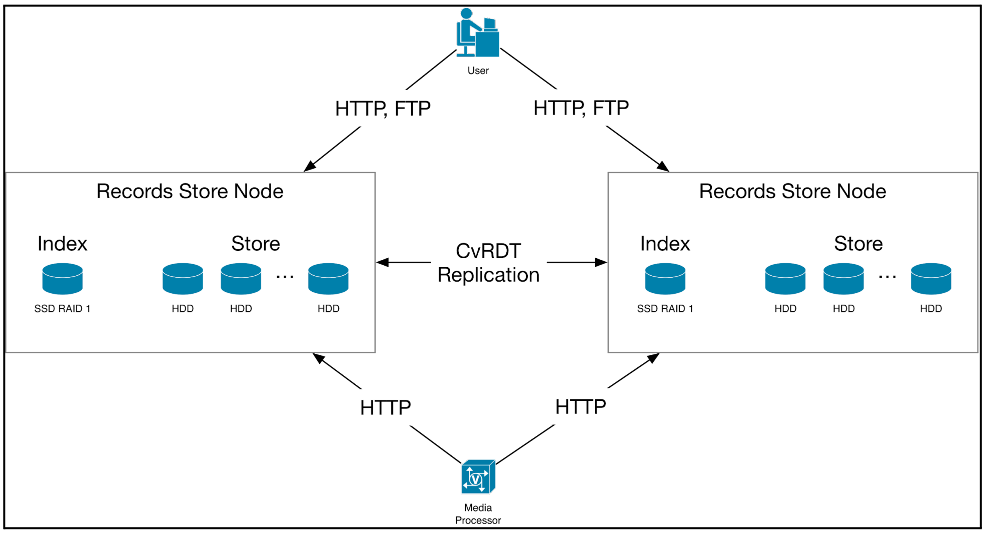
The built-in DBMS replication mechanisms ensure the service's fault tolerance.

Call Recording Storage

The Call Recording Storage is a service designed for storing and retrieving call records.

The centrex-records-store component provides this service. The component implements mechanisms for receiving, storing, and delivering call records and supports HTTP and FTP protocols.

The server Cloud PBX Core Recording Storage, written in the Go programming language, is used as the storage.



The service represents a distributed, eventually strongly consistent storage with a configurable replication factor. The storage employs its synchronization protocol based on conflict-free replicated data types (CvRDT), making it resilient to network partitioning and allowing replicas to be placed in different data centers. The storage utilizes a consistent hashing algorithm to distribute records across nodes and can scale up to petabytes of records.

Logging System

The logging system is a service for maintaining a unified log of requests to centrex-frontend nodes (part of the Communication Core service Centrex with the ability to view and filter events.

The components logs-agent and centrex-logs provide this service. The logs-agent component is designed to collect data on the operation of services and send them to the storage, while the centrex-logs component implements the storage and provides a mechanism for log filtering.

Metrics Collection

The Metrics Collection service is designed to receive, store, and provide metrics for the status of installation servers/system services.

The components centrex-metrics-collector and centrex-metrics-processor provide this service. The centrex-metrics-collector component ensures metric collection, while the centrex-metrics-processor component offers a mechanism for processing the received data.

A queue service and a DBMS are required to operate the service.

Pre-Billing

Pre-Billing is a service designed to facilitate interactions with operator operational and business systems.

The bossie component provides this service.

To operate the service, a DBMS is required.

Integration With CRM Systems

This service is designed for integrating with external CRM systems.

Call Authorization

The Call Authorization service is designed to block unauthorized calls and limit services.

Installation

General Information

Platform installation is a set of servers, both physical and/or virtual machines, on which the components of the platform are deployed, and services are configured.

The installation is designed based on system requirements. During the planning phase, the following aspects are determined:

1. The set and composition of services. #. Type and quantity of platform components. #. Components that ensure the operation of services.
2. Server configuration for deploying components.

Components can be deployed on either physical servers or in a virtualized environment. A physical or virtual machine, or a container where a component is deployed and running, is called an **installation server**.

Server requirements are detailed in the Resources section, and the installation deployment sequence is described in the Deployment section.

Installation Scheme

Within this document, installation servers are named following the primary component deployed on the server:

- FE — a server on which the Front End component is deployed.
- BE — a server on which the Back End component is deployed.
- MP — a server on which the Media Processor component is deployed.

Compatibility

Supported Hardware Platforms

- Intel X86-64
- IBM z/Series (Requires the presence of Intel X86-64 nodes for processing media streams).

Supported Operating Systems

- Debian GNU/Linux 9 and above
- Red Hat Enterprise Linux 8.0 and above
- Fedora Linux 28 and above
- IBM z/LinuxONE.

Supported Virtualization Environments

- Linux KVM
- VMware ESXi
- IBM z/VM.

The Deployment section provides recommendations for configuring the virtualization environment.

Deployment

Deployment Sequence

1. Server Preparation. Deployment on bare metal. In virtualization systems. In containers.
2. Server Configuration. OS installation and configuration. Software installation. Network connectivity setup.
3. Deployment of platform services (components).
4. Routing configuration.

Distribution of Components Across Servers

Communications Core

Recommendations for deploying the Centrex service: placing each communications core component on a separate server is recommended.

Auxiliary Services

Recommendations for auxiliary services:

1. Call History: It is recommended to place Centrex History components on separate servers. It is recommended to place the DBMS on separate servers.
2. Centrex Record Store is recommended to be placed on separate servers.

Installation Network Connectivity

Connecting FE, BE, and MP servers through independent networks is recommended.

- The OAM Network (Operations, Administrations, and Maintenance Network) is intended for internal communication between FE, BE, and MP servers.
- Public IP Network (border zone) connects external subscribers.
- SIG and Media networks are intended for connecting gateways and other operator services.

The OAM, SIG, Media, and Public networks should be physically separate or virtual local area networks (VLANs). It is assumed that there is no routing between these networks.

FE servers can be located in the operator's Public IP DMZ, as Front-End nodes don't store client data in long-term memory and fully implement the functionality of a session border controller (SBC), including:

- Limiting incoming traffic capacity (call admission control) both in general and for individual directions and clients.
- Limiting the number of sessions both in general and for individual directions and clients.
- Limiting traffic from unauthorized and unknown sources.
- Access control lists (ACLs) for individual clients.
- Ensuring the correctness of the SIP protocol and protection against improperly formed, including malicious, packets.
- Mechanisms to notify potential malicious activity (username and password guessing, unexplained traffic growth).
- An API for integration with systems that block malicious traffic to protect against attacks, including distributed attacks aimed at blocking a service (DoS and DDoS).

Interaction Between Installation Servers

Connected Servers	Cluster Node Protocol	Protocol Purpose	Network
BE, FE <-> BE, FE	Discovery	Notification of cluster topology changes	OAM
BE <-> BE	Inter-cluster	Data synchronization	OAM
FE -> BE	Inter-cluster	Intra-cluster RPC, client data retrieval, global process synchronization	OAM
FE <-> FE	Inter-cluster	Intra-cluster RPC, message exchange between signaling tasks	OAM
FE, MG <-> FE, MG	MG Discovery	Notification of MP composition changes, notification of MP loading	OAM
FE, MG <-> FE, MG	Inter-cluster	MP RPC (creation of RTP endpoints, statistics, DTMF)	OAM
MG <-> MG	Inter-cluster RTP	Media stream switching between different MP	OAM
BE -> FE	HTTP	Transfer of static files for business applications or user files for subsequent user delivery	OAM
FE -> BE	HTTP	User file storage	OAM
FE -> GIT	HTTP/GIT	Deployment of business applications	OAM
BE -> MG	HTTP	Transfer of user media files and media files of business applications for subsequent delivery in a voice channel	OAM
MG -> BE	HTTP	Storage of user media files, call recording (in the absence of an external record store)	OAM
MG -> RS	HTTP	Saving call records in the call recording repository	OAM
SLB <-> SLB	VRRPv2	Selection and assignment of a virtual cluster IP	SIG
SLB <-> SLB	IPVS/Sync	Synchronization of the master/slave balancer state	SIG
SLB -> FE	HTTP	Distribution of incoming client HTTP requests	SIG
SLB -> FE	SIP, SIP/TCP	Distribution of incoming client SIP/UDP packets and SIP/TCP connections	SIG
SLB -> FE	HTTP	FE health check	SIG
SLB -> RS	HTTP	Distribution of incoming client requests to the call recording store, health check of call recording store nodes	SIG
FE <-> Telegram GW	HTTP	Proxies requests to ensure the functionality of missed call notifications through Telegram	SIG
FE <-> CRM	HTTP/HTTPS	Services for integration with external CRM systems	OAM
FE, MG -> MON	SNMP	Transmission of current operational status indicators of nodes	OAM
*** -> MON**	SNMP	Each node includes standard snmpd for transmitting current operational status indicators of the operating system	OAM
FE <-> Hist	HTTP	Storage of call event records, query for statistics and call history	OAM
FE <-> KAFKA	TCP	Transmission of application statistics data.	OAM
KAFKA -> Metric		Storage and aggregation of application metrics and client statistical data	OAM
FE <-> BOSSGW	HTTP	Requests for information from the operator's B/OSS, receipt of provisioning notifications for B/OSS events	OAM

FE, BE, MG -> Log	TCP	Storage of system log records	OAM
TS -> *	SSH	Technological access from the terminal server to all platform nodes. Node management through Ansible	OAM

Geo-Distributed Operation Mode

To enhance availability, deploying a geographically distributed installation is recommended. The platform supports two options for geographically distributed installations:

- distributed installation
- redundant installation.

Distributed Installation

A distributed installation involves placing individual servers in two data processing centers. This mode of operation is recommended when reliable low-latency communication channels are available, such as within the same city.

Redundant Installation

This deployment option provides full installation replication or mutual replication of two installations in a hot standby mode. This mode of operation is recommended for setting up a geographically distributed installation in two different regions.

Reliability

The reliability of systems built on the platform depends on compliance with hardware and configuration requirements.

To enhance availability and fault tolerance, deploying a geographically distributed installation is recommended.

Fault Tolerance and Availability

It is essential to ensure fault tolerance at both the software and hardware levels to deploy an installation based on the platform without a single point of failure.

When configuration and hardware requirements are met, the installation's availability can reach 99.9%.

Software Fault Tolerance

At the software level, installation fault tolerance is achieved through the platform's logic and cluster configuration:

- For communication core nodes, using the N + 1 topology is necessary.
- For nodes implementing auxiliary services, use either the N + 1 topology or the 1 + 1 hot standby model.

Fault Tolerance for Core Nodes

All communication core nodes are active and participate in traffic processing.

The cluster controller monitors the state of core nodes.

In case of hardware or software failure of FE and MP nodes, the load is automatically redistributed to the remaining nodes. The Platform Core section provides a detailed description of the node selection algorithm.

If a BE node is unavailable for 1000 ms, it is automatically removed from the cluster. The replication factor specified during system deployment determines the maximum number of simultaneous failures on BE nodes. The Platform Core section provides detailed information on data storage organization and BE node operation.

Fault Tolerance for Nodes Implementing Auxiliary Services

For auxiliary nodes such as IN-SCP, CH, and Call Record Store, load balancing mechanisms are used between equivalent nodes.

Hot standby is utilized for auxiliary nodes like LB, B/OSS Gateway, and CRM-I.

Hardware Fault Tolerance

The following steps are necessary to ensure hardware fault tolerance:

- Ensure compliance with the hardware requirements.
- Organize the redundancy of installation hardware, including using RAID arrays and duplicate network cards with NIC Teaming support.

Handling Failures

In case of software or hardware failure, load redistribution within the cluster or switching to a standby node is performed automatically.

The platform supports automatic recovery after the failure of individual nodes or the entire system in a catastrophic event (e.g., data center power failure).

Resilience Against Exceeding Assignment Metrics

All platform components include built-in mechanisms for protection against exceeding assigned traffic metrics. Components may selectively refuse service when assignment metrics are exceeded, maintaining traffic at the highest possible level without performance degradation.

Compatibility

Technical Specifications

Supported Protocols

IPv4	Internet Protocol version 4
IPv6	Internet Protocol version 6
SIP	Session Initiation Protocol (IETF RFC 3261)
SCTP	Stream Control Transmission Protocol (IETF RFC 4960)
M3UA	Signaling System 7 Message Transfer Part 3 User Adaptation Layer (IETF RFC 4666)
CAP	CAMEL Application Part (3GPP TS 23.078 Release 9)
RTP/RTCP	Transport Protocol for Real-Time Applications (IETF RFC 3550)
STUN	Session Traversal Utilities for NAT (IETF RFC 5389)
ICE	Interactive Connectivity Establishment (IETF RFC 8445)
SDP	Session Description Protocol (IETF RFC 4566)
DTMF	RTP Payload for DTMF (IETF RFC 2833, 4733)
TLS	Transport Layer Security Protocol v. 1.3 (IETF RFC 8446)
SRTP	Secure Real-time Transport Protocol (IETF RFC 3711)
DTLS	Datagram Transport Layer Security v. 1.2 (IETF RFC 6347)
RADIUS	Remote Authentication Dial In User Service (IETF RFC 2865)
DIAMETER	Diameter Base Protocol (IETF RFC 6733)
LDAP	Lightweight Directory Access Protocol (IETF RFC 4511)
SNMPv1	Simple Network Management Protocol Version 1 (IETF RFC 1157)

The complete list of supported standards and recommendations is available in the Appendix.

Supported Hardware Platforms

- Intel X86-64
- IBM z/Series (requires the presence of Intel X86-64 nodes for media stream processing).

Supported Operating Systems

- Debian GNU/Linux 9 and above
- Red Hat Enterprise Linux 8.0 and above
- Fedora Linux 28 and above
- IBM z/LinuxONE.

Supported Virtualization Environments

- Linux KVM
- VMware ESXi
- IBM z/VM.

The Scaling Principles section provides recommendations for configuring the virtualization environment.

Fault Tolerance and Availability

The communication core of the Cloud PBX Core platform is a dynamically scalable high-availability cluster without a single point of failure. All core nodes are active and participate in traffic processing. In case of failure of any node, the load is automatically redistributed among the remaining nodes.

All auxiliary nodes of the platform also use one of the following redundancy mechanisms to ensure high availability:

- Load balancing among equivalent nodes (call record storage service, IN-SCP, metric collection service, and others).
- Hot standby (B/OSS integration services, CRM integration services, SLB, and others).

Thanks to the full redundancy of all nodes, the Cloud PBX Core platform can guarantee availability of no less than 99.9%. Depending on individual installations' architecture and specific hardware solutions, installation availability can reach 99.99%. Recommendations for designing highly available systems based on the platform are provided in the Redundancy Principles section.

Voice Quality

The architecture of the Cloud PBX Core platform allows for maximum voice quality (MOS >= 4). The platform's speech processing nodes collect an extensive set of metrics for each media stream, enabling control of both instantaneous quality characteristics and changes in voice transmission quality over time. In addition, the platform includes an external voice quality control and assessment tool used for monitoring quality from external networks.

Performance

The Cloud PBX Core platform does not impose fundamental architectural limitations on installation capacity. The number of nodes within an installation depends on the target capacity and load profile. The Scaling Principles section provides recommendations for planning installation capacity.

Lawful Interception (LI)

The Cloud PBX Core platform is fully compatible with all certified technical means for the Lawful interception (LI). This includes subsystems for passive traffic interception directly from OAM, SIG, and Media IP networks, supporting Ethernet, IPv4, IPv6, RADIUS, DIAMETER, SIP, SDP, and RTP protocols.

The platform also allows for the complete adaptation of recorded CDRs, considering an operator's service nomenclature.

The platform includes tools for copying transcoded and, if necessary, decrypted RTP traffic into GRE or ERSPAN channels and tools for generating synthetic SIP signaling traffic reflecting call status, with subsequent forwarding to GRE or ERSPAN channels. Copying can be performed for all calls or specified calls only. Thus, the platform ensures compatibility with passive interception systems, even for SRTP and DTLS-SRTP (WebRTC) calls, including calls with signaling through WebSocket.

Appendices

Abbreviations and Terminology Glossary

GPP — 3rd Generation Partnership Project: A consortium developing standards for mobile networks.

API — Application Programming Interface.

CAPS — Call Attempts per Second.

CC — Collision Count.

FTP — File Transfer Protocol.

HSS — Home Subscriber Server.

IMS — IP Multimedia Subsystem.

IPVS — IP Virtual Server: A Linux kernel module that allows load balancing at the IP address level.

MGCF — Media Gateway Controller Function.

MGW — Media Gateway.

MRF — Multimedia Resource Function.

MRFC — Multimedia Resource Function Controller.

MRFP — Multimedia Resource Function Processor.

NIC — Network Interface Card.

P/I/S-CSCF — Proxy, Interrogating, and Serving Call Session Control Function.

SCP — Secure Copy: A file copying protocol, part of the SSH package.

SGW — Signalling Gateway Interface.

:SIP AS — SIP Application Server.

:SSH — Secure Shell: A network protocol for securing remote UNIX systems connections.

VRRP — Virtual Router Redundancy Protocol.

PSTN — Public Switched Telephone Network.

Supported Standards and Recommendations

Supported 3GPP and GSMA Recommendations

3GPP TS 23.078	Mobile network Enhanced Logic (CAMEL) Application Part (CAP)
3GPP TS 24.228	Signalling flows for the IP multimedia call control based on Session Initiation Protocol (SIP) and Session Description Protocol (SDP)
3GPP TS 24.229	Internet Protocol (IP) multimedia call control protocol based on Session Initiation Protocol (SIP) and Session Description Protocol (SDP)
3GPP TS 24.237	IP Multimedia Subsystem (IMS) Service Continuity
3GPP TS 24.292	IP Multimedia Subsystem (IMS) centralized services
3GPP TS 24.604	Communication Diversion (CDIV) using IP Multimedia (IM) Core Network (CN) subsystem; Protocol specification
3GPP TS 24.615	Communication Waiting (CW) using IP Multimedia (IM) Core Network (CN) subsystem; Protocol specification ITU-T Q.1912.5 Interworking between Session Initiation Protocol (SIP) and Bearer
GSMA PRD IR.92	IMS Profile for Voice and SMS Version 1.0
GSMA PRD IR.94	IMS Profile for Conversational Video Services Version 4.0

Supported IETF Recommendations

RFC 3261	SIP: Session Initiation Protocol, June 2002
RFC 2327	SDP Session Description Protocol, April 1998
RFC 2543	SIP Session Initiation Protocol, March 1999
RFC 3261	SIP Session Initiation Protocol, June 2002
RFC 3262	Reliability of Provisional Responses in Session Initiation Protocol, June 2002
RFC 3311	The Session Initiation Protocol (SIP) UPDATE Method, September 2002
RFC 3312	Integration of Resource Management and Session Initiation Protocol (SIP), October 2002
RFC 3323	A Privacy Mechanism for the Session Initiation Protocol (SIP), November 2002
RFC 3325	Private Extensions to the Session Initiation Protocol (SIP) for Asserted Identity within Trusted Networks, November 2002
RFC 3326	The Reason Header Field for the Session Initiation Protocol (SIP)
RFC 3398	Integrated Services Digital Network (ISDN) User Part (ISUP) to Session Initiation Protocol (SIP) Mapping, December 2002
RFC 3420	Internet Media Type Message/sipfrag Session Initiation Protocol (SIP) Mapping, December 2002
RFC 3515	The Session Initiation Protocol (SIP) Refer Method
RFC 3892	The Session Initiation Protocol (SIP) Referred-By Mechanism
RFC 4028	Session Timers in the Session Initiation Protocol (SIP), April 2005
RFC 4412	Communications Resource Priority for the Session Initiation Protocol (SIP)
RFC 4458	Session Initiation Protocol (SIP) URIs for Applications such as Voicemail and Interactive Voice Response (IVR)
RFC 4488	Suppression of Session Initiation Protocol (SIP) REFER Method Implicit Subscription
RFC 4538	Request Authorization through Dialog Identification in the Session

	Initiation Protocol (SIP)
RFC 4575	A Session Initiation Protocol (SIP) Event Package for Conference State
RFC 4975	The Message Session Relay Protocol (MSRP)
RFC 5009	Private Header (P-Header) Extension to the Session Initiation Protocol (SIP) for Authorization of Early Media
RFC 5626	Managing Client-Initiated Connections in the Session Initiation Protocol (SIP)
RFC 5806	Diversion Indication in SIP
RFC 6050	A Session Initiation Protocol (SIP) Extension for the Identification of Services
RFC 6086	Session Initiation Protocol (SIP) INFO Method and Package Framework
RFC 6665	SIP-Specific Event Notification
RFC 6809	Mechanism to Indicate Support of Features and Capabilities in the Session Initiation Protocol (SIP)
RFC 7044	An Extension to the Session Initiation Protocol (SIP) for Request History Information
RFC 7315	Private Header (P-Header) Extensions to the Session Initiation Protocol (SIP) for the 3GPP
RFC 6585	Additional HTTP Status Codes M. Nottingham, R. Fielding. April 2012.
RFC 6585	Additional HTTP Status Codes M. Nottingham, R. Fielding. April 2012.
RFC 5785	Defining Well-Known Uniform Resource Identifiers (URIs) M. Nottingham, E. Hammer-Lahav. April 2010.
RFC 2818	HTTP Over TLS E. Rescorla. May 2000.
RFC 2817	Upgrading to TLS Within HTTP/1.1 R. Khare, S. Lawrence. May 2000.
RFC 2617	HTTP Authentication: Basic and Digest Access Authentication J. Franks, P. Hallam-Baker, J. Hostetler, S. Lawrence, P. Leach, A. Luotonen, L. Stewart. June 1999.
RFC 2616	Hypertext Transfer Protocol -- HTTP/1.1. R. Fielding, J. Gettys, J. Mogul, H. Frystyk, T. Berners-Lee. January 1997.
RFC 2388	Returning Values from Forms: multipart/form-data L. Masinter. August 1998.
RFC 2145	Use and Interpretation of HTTP Version Numbers J. C. Mogul, R. Fielding, J. Gettys, H. Frystyk. May 1997.
RFC 2109	HTTP State Management Mechanism D. Kristol, L. Montulli. February 1997.
RFC 4590	RADIUS Extension for Digest Authentication. B. Sterman, D. Sadolevsky, D. Schwartz, D. Williams, W. Beck. July 2006.
RFC 3748	PPP Extensible Authentication Protocol (EAP). B. Aboba, L. Blunk, J. Vollbrecht, J. Carlson, H. Levkowetz. June 2004.
RFC 3579	RADIUS (Remote Authentication Dial In User Service). Support For Extensible Authentication Protocol (EAP). B. Aboba, P. Calhoun. September 2003.
RFC 3079	Deriving Keys for use with Microsoft Point-to-Point Encryption (MPPE). G. Zorn. March 2001.
RFC 2869	RADIUS Extensions. C. Rigney, W. Willats, P. Calhoun. June 2000.
RFC 2868	RADIUS Attributes for Tunnel Protocol Support. G. Zorn, D. Leifer, A. Rubens, J. Shriver, M. Holdrege, I. Goyret. June 2000.
RFC 2866	RADIUS Accounting. C. Rigney. June 2000.
RFC 2865	Remote Authentication Dial-In User Service (RADIUS). C. Rigney, S. Willens, A. Rubens, W. Simpson. June 2000.
RFC 2759	Microsoft PPP CHAP Extensions, Version 2. G. Zorn. January 2000.
RFC 2548	Microsoft Vendor-specific RADIUS Attributes. G. Zorn. March 1999.
RFC 1994	PPP Challenge Handshake Authentication Protocol (CHAP) W. Simpson.

	August 1996.
RFC 5389	Session Traversal Utilities for NAT (STUN). J. Rosenberg, R. Mahy, P. Matthews, D. Wing. October 2008.
RFC 3489	STUN - Simple Traversal of User Datagram Protocol (UDP) Through Network Address Translators (NATs). J. Rosenberg, J. Weinberger, C. Huitema, R. Mahy. March 2003.
RFC 2578	Structure of Management Information Version 2 (SMIv2). K. McCloghrie, D. Perkins, J. Schoenwaelder. April 1999.
RFC 1907	Management Information Base for Version 2 of the Simple Network Management Protocol (SNMPv2). SNMPv2 Working Group & others. January 1996.
RFC 1906	Transport Mappings for Version 2 of the Simple Network Management Protocol (SNMPv2). SNMPv2 Working Group & others. January 1996.
RFC 1905	Protocol Operations for Version 2 of the Simple Network Management Protocol (SNMPv2). SNMPv2 Working Group & others. January 1996.
RFC 1904	Conformance Statements for Version 2 of the Simple Network Management Protocol (SNMPv2). SNMPv2 Working Group & others. January 1996.
RFC 1903	Textual Conventions for Version 2 of the Simple Network Management Protocol (SNMPv2). SNMPv2 Working Group & others. January 1996.
RFC 1902	Structure of Management Information for Version 2 of the Simple Network Management Protocol (SNMPv2). SNMPv2 Working Group & others. January 1996.
RFC 1901	Introduction to Community-based SNMPv2. SNMPv2 Working Group & others. January 1996.
RFC 1212	Concise MIB Definitions. Rose, M., and K. McCloghrie. March 1991.
RFC 1157	A Simple Network Management Protocol (SNMP). J. Case, M. Fedor, M. Schoffstall, J. Davin. May 1990.

Supported SMPP Version

SMPP 3.4	Short Message Peer to Peer Protocol Specification v3.4. SMPP Developers Forum. October 1999.

Applied Cryptographic Standards and Libraries

TLS 1.2	The Transport Layer Security (TLS) Protocol Version 1.2. IETF RFC 5247, August 2008
TLS 1.3	The Transport Layer Security (TLS) Protocol Version 1.3. IETF RFC 8446, August 2018
DTLS 1.2	Datagram Transport Layer Security Version 1.2. IETF RFC 6347, January 2012
OpenSSL	OpenSSL v.1.1.1c
libSRTP	libSRTP v.2.3
Go Crypto	Go v.1.13