VoIP as a Service: Asterisk in OpenStack platform

Martin Kontsek
Faculty of Management Science and Informatics
University of Zilina
Zilina, Slovakia
martin.kontsek@fri.uniza.sk

Pavel Segec
Faculty of Management Science and Informatics
University of Zilina
Zilina, Slovakia
pavel.segec@fri.uniza.sk

Marek Moravcik
Faculty of Management Science and Informatics
University of Zilina
Zilina, Slovakia
marek.moravcik@fri.uniza.sk

Ivana Bridova
Faculty of Management Science and Informatics
University of Zilina
Zilina, Slovakia
ivana.bridova@fri.uniza.sk

2025 International Conference on Emerging eLearning Technologies and Applications (ICETA) | 979-8-3315-7426-0/25/$31.00 ©2025 IEEE | DOI: 10.1109/ICETA67772.2025.11280162

***Abstract*—This article explores the integration of Asterisk as a VoIP service within the OpenStack cloud platform. It highlights the advantages of cloud services, particularly for end users and IT departments. The implementation process of Asterisk, including installation, configuration, and the development of automation scripts, is detailed. The significance of Network Address Translation (NAT) functionality is emphasized, along with the development of user-friendly configuration scripts. Lastly, the article discusses challenges encountered with the ManageIQ integration, suggesting potential improvements for future deployments.**

***Index Terms*—Cloud Computing, OpenStack, Asterisk, VoIP, ManageIQ**

I. Introduction

The trend toward the utilization of cloud services is currently gaining momentum. This phenomenon is driven by various factors, including technological advancements as well as economic and organizational aspects. For end users, leveraging cloud services presents a straightforward and efficient means of accessing diverse applications and data storage without the need to invest in personal infrastructure. Financially, it often proves to be more advantageous, as companies can avoid the high costs associated with purchasing expensive servers and hardware, opting instead to use available cloud platforms for a monthly fee.

For administrators and IT departments, utilizing cloud services is equally beneficial. Managing the devices on which these services operate becomes simpler, as physical maintenance and server updates are no longer a concern. Cloud platforms often provide tools for monitoring and managing services, enhancing efficiency and offering better insights into who utilizes the service, when, and to what extent [1].

In the context of our specific task—implementing Asterisk as a cloud service for the staff of the Department of Information Networks—it is essential to consider multiple aspects. These include security, scalability, availability, and performance. The implementation of Asterisk in the cloud should be well thought out, taking into account user needs as well as the technical requirements of the system. It is crucial to ensure that communication through Asterisk is reliable and secure while also being sufficiently flexible to adapt to the changing needs of users. The selection of an appropriate cloud platform and the configuration of Asterisk are key steps in this implementation [2, 3].

In recent years there weren't many attempts to bring VoIP as a service (VoIPaaS) into private cloud platforms. In [4], researchers are describing their solution to deploy Asterisk VoIP into Oracle public cloud platform. In [5], there is a description of highly available container based VoIP service using Kubernetes. Article [6] presents custom VoIP application experimentaly verified in Docker container environment. Researchers in [7] present custom Unified Communication solution deployed in private OpenStack cloud platform.

This article introduces the reader to the VoIP cloud service, which was developed in the Department of InfoComm Networks, FRI, UNIZA. The motivation for this service creation was to simplify the deployment of the open-source VoIP server used across several courses by students and educators and also to support research by multiple members of the Faculty. The second section of the article presents Asterisk PBX as the base of the VoIPaaS. The third section overviews a private Cloud Computing platform maintained by the teachers and students in the Department of InfoComm Networks. Forth section discusses the design and deployment of the proposed VoIPaaS service.

II. Asterisk

Asterisk is an open-source software framework designed for building multi-protocol communication applications in a real time, primarily focused on voice over IP (VoIP) services. Initially developed by Mark Spencer in 1999 and now maintained by Sangoma Technologies, Asterisk has evolved into one of the most versatile and widely adopted platforms for creating

979-8-3315-7426-0/25/$31.00 ©2025 IEEE
Authorized licensed use limited to: University of Zilina. Downloaded on December 22,2025 at 10:38:11 UTC from IEEE Xplore. Restrictions apply.

telephony applications. It is used globally to support a range of services, including private branch exchange (PBX) systems, VoIP gateways, and conference servers. Asterisk's open-source nature provides users with the flexibility to develop tailored communication solutions, whether for small business environments or large-scale telephony infrastructures. As a robust and cost-effective alternative to proprietary telecommunication systems, Asterisk has gained widespread use in both enterprise and service provider markets. Asterisk is used by various states all around the world.

One of the primary strengths of Asterisk lies in its adaptability, as it supports a wide array of communication protocols, including the Session Initiation Protocol (SIP), H.323, and other legacy telephony systems. This flexibility allows seamless integration with both traditional and IP-based telephony networks, providing businesses with the opportunity to modernize their communication infrastructure without fully abandoning older technologies.

A. Key features

Asterisk provides a comprehensive set of features that to meet both basic and advanced communication needs. At its core, Asterisk functions as a fully-featured PBX (Private Branch Exchange) system, offering essential call management features such as call routing, voicemail, call transfer, and call waiting. These capabilities allow businesses to effectively manage internal and external communication, optimizing their workflow through automation and streamlined call handling processes. Additionally, Asterisk's PBX functionality can be enhanced by integrating it with Interactive Voice Response (IVR) systems, which enable the automation of incoming call handling through voice menus.

One of Asterisk's defining features is its role as a VoIP gateway, bridging traditional telephony systems with modern VoIP networks. By supporting multiple communication protocols, Asterisk allows organizations to transition from analog and digital telephony systems to VoIP, enabling them to leverage the cost efficiency and scalability of internet-based communications. The inclusion of industry-standard protocols such as SIP and H.323 ensures compatibility with a wide range of VoIP services and devices, while also allowing the integration of legacy systems via MGCP and SCCP.

Asterisk's dialplan configuration system further enhances its flexibility by allowing users to define custom call handling logic. The dialplan is highly configurable and can be scripted to manage call routing based on user-defined rules, which can be as simple or complex as required [8, 9].

B. Architecture

Asterisk's architecture is composed of a core system that manages key functionalities, such as reading configuration files and building the dialplan, which controls how calls are handled. The core interacts with modules, which add various capabilities, including channel drivers (e.g., SIP), voicemail, and conferencing features. These modules are dynamically loaded, allowing for customization and scalability. Channels facilitate communication between devices, and calls are managed by connecting these channels. Asterisk's architecture is modular, as shown in Fig. 1, enabling integration with other telephony systems and extensive customization [10].

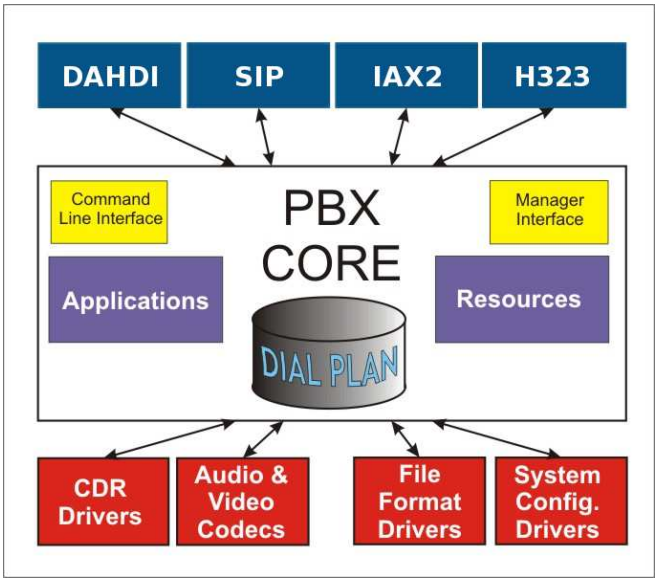


Fig. 1. Asterisk Architecture [11]

C. Difference between chan_sip and pjsip

In Asterisk, the chan_sip and PJSIP modules serve as SIP channel drivers, but they differ significantly in terms of architecture, performance, and features. chan_sip, the older and now deprecated module, was widely used for basic SIP functionality, but it faced limitations in terms of scalability, security, and complex configuration management. PJSIP, by contrast, is a modern, highly scalable SIP stack that offers enhanced performance, modularity, and better support for advanced features like NAT traversal, TLS, and SRTP. Unlike chan_sip, PJSIP also supports asynchronous operations, which improve system efficiency and responsiveness under high call volumes. Additionally, PJSIP's configuration is more flexible and allows for better integration with external systems, making it the preferred choice for contemporary Asterisk deployments. Due to its ongoing support and richer feature set, PJSIP is generally recommended for new Asterisk installations over chan_sip, which has been deprecated [12].

III. OpenStack

OpenStack is an open-source cloud computing platform designed to enable the development and management of public and private clouds. Initially developed by NASA and Rackspace, OpenStack is now maintained by the OpenStack Foundation. It provides a set of interrelated services that manage computing, storage, and networking resources through a web-based dashboard, command-line tools, and RESTful APIs. OpenStack is designed to be highly scalable, allowing businesses to build large, multi-tenant cloud infrastructures.

979-8-3315-7426-0/25/$31.00 ©2025 IEEE 428
Authorized licensed use limited to: University of Zilina. Downloaded on December 22,2025 at 10:38:11 UTC from IEEE Xplore. Restrictions apply.

A. Key features

One of OpenStack's primary features is its modular architecture, which is composed of various services that can be integrated based on an organization's needs. These services include Nova, which provides compute resources by managing virtual machines (VMs), and Neutron, responsible for managing networking, enabling complex network configurations such as software-defined networking (SDN). Another core service is Cinder, which handles block storage, enabling the creation and management of persistent storage volumes for instances. For object storage, OpenStack offers Swift, which provides scalable storage ideal for archiving and large-scale data storage.

OpenStack also emphasizes multi-tenancy, allowing different users or departments within an organization to share a cloud environment while maintaining isolated resources. This capability makes it ideal for both private and public clouds, where resource allocation and security are paramount. Furthermore, OpenStack is highly scalable and can be deployed across hundreds or even thousands of servers, making it suitable for large-scale enterprise environments. Its open-source nature and large community of contributors ensure that it stays at the forefront of cloud innovations while offering a cost-effective solution compared to proprietary cloud services. OpenStack's flexibility, wide-ranging support for virtualization technologies, and active community contribute to its growing adoption in both enterprise and service provider environments [13].

IV. Integration of Asterisk to OpenStack

In this section we are describing our solution for building VoIPaaS based on Asterisk for OpenStack. During the creation of our solution, we have researched multiple solutions done in the past by other research groups which influenced our work [14, 15, 16]

A. Installation and configuration of Asterisk

The initial phase of our project involved the manual installation of Asterisk on a Debian instance within the OpenStack environment. We began by deploying a new instance featuring the latest version of Debian. Our intention was to install Asterisk directly from the default package repositories. However, we encountered an obstacle, as the version of Asterisk available was outdated and not aligned with our requirements. To address this issue, we needed to configure the system to include access to older repositories by adding the necessary entries to the sources list, as shown in Fig. 2.

Upon successfully installing Asterisk, we developed a basic configuration that enabled calls to a specific number, accompanied by an audio playback feature. This initial setup facilitated our understanding of Asterisk's capabilities. We then enhanced the configuration to allow for calling between various users registered within the Asterisk exchange. These configurations served as valuable exercises for reinforcing our knowledge of Asterisk's functionalities. However, recognizing the importance of utilizing updated technology, we decided



Fig. 2. Debian sources list

to upgrade Asterisk to a newer version to incorporate the res_pjsip module, which is a modern alternative to the deprecated chan_sip module.

To facilitate this transition, we terminated the existing Debian instance and launched a new one, following the installation guidelines specified in the Asterisk documentation [17]. Once the installation was complete, we created a configuration nearly identical to our previous setup, ensuring it was compatible with the res_pjsip module. Given the lengthy nature of the testing phase, we opted to develop a script that could be embedded in the instance's metadata during its creation. This script would automate the installation of all prerequisites for Asterisk, streamline the installation process itself, and establish a fundamental test configuration.

B. NAT Functionality

The subsequent step in our project involved the integration and testing of Network Address Translation (NAT) functionality, which is critical for VoIP applications operating within private networks. To implement this, we utilized the Floating IP feature provided by OpenStack. We initiated the creation of a new private network, which enabled the addition of new instances while ensuring connectivity to the public internet via a router. By assigning the Floating IP to the external interface of the router, as shown in Fig. 3, we ensured that our VoIP exchange became accessible from the public network, thus facilitating communication with external devices beyond the confines of the private network.

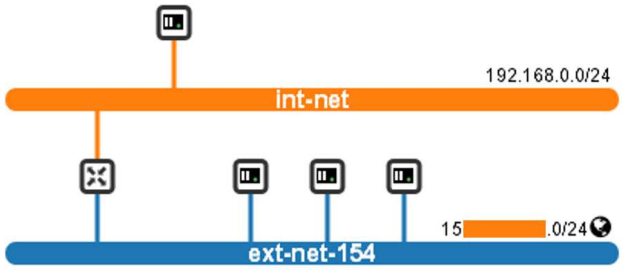


Fig. 3. OpenStack network with FloatingIP router

979-8-3315-7426-0/25/$31.00 ©2025 IEEE
Authorized licensed use limited to: University of Zilina. Downloaded on December 22,2025 at 10:38:11 UTC from IEEE Xplore. Restrictions apply.

C. Script for creating configuration

In our efforts to streamline the configuration process, we aimed to provide users with an accessible method for configuring the exchange without requiring in-depth knowledge of Asterisk module configuration files. To this end, we developed a user-friendly script designed to generate a basic configuration and assist users in creating account configurations for the exchange. The script requires minimal user input, specifically the names and passwords for the accounts to be created. Users can specify one or multiple accounts during the initial setup, and the script also allows for the generation of new configurations or the addition of additional accounts as needed.

D. Adding a service to the ManageIQ environment

Finally, we sought to further automate the entire process through ManageIQ, enabling users to deploy new instances with a single click. This would save the user from having to manually select instance parameters and copy the script to the ”User Data” section.

The ManageIQ platform is an open-source system designed for managing hybrid IT environments, which provides comprehensive tools for managing, monitoring, automating and optimizing infrastructure across virtualization, private and public clouds, containers, networks and storage. It was created as a community project under Red Hat [18, 19, 20].

We started by creating a so-called ”Customization Template”, which is used to insert Cloud-init data for the instance being created. In the script section, we inserted the script that we had been using so far and the user had to manually copy it into the given field in the OpenStack environment. Next, we started creating a specific service. We created a new item under the student test catalog item and filled in the necessary fields, such as Name, Service catalog selection and Dialog for user interaction, which is shown in Fig. 4.

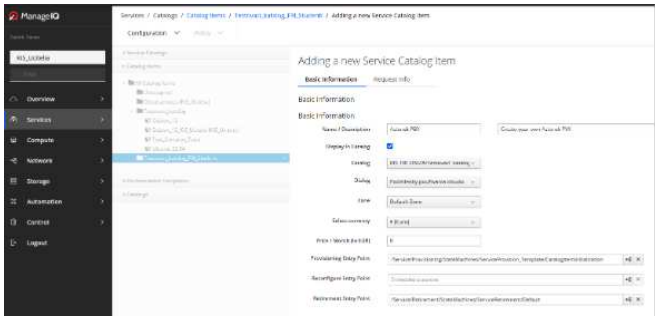


Fig. 4. ManageIQ catalog creation

We continued the configuration on the next tab, which contains several subtabs. There are also some mandatory fields in these subtabs that need to be filled in. This is where we encountered the first problems, where, for example, in the “Environment” subtab, we could not select any value for the “Cloud tenant” attribute. This is a mandatory attribute and unfortunately, even when we tried to create a new one in the ManageIQ environment, it did not appear in the drop-down menu. The same problem occurred on the ”Properties” subtab, where we could not select a specific instance type. Next, in ”Volumes”, we set the disk size to 20GB and checked the option to delete it when deleting the instance.

Based on our further testing, the implementation of ManageIQ in our OpenStack environment did not incorporate all the desired features, limiting the automation capabilities we had required for VoIPaaS automated deployment in our environment. We are, however, planning to further test ManageIQ integration in further research.

E. Steps for user to use VoIPaaS in OpenStack cloud platform

After logging into the OpenStack environment, click the ”Launch Instance” button in the ”Project/Compute/Instances” section. A dialog box will then appear in which you need to fill in the mandatory parameters marked with an asterisk. These parameters include the instance name, disk size (20GB), a button to delete the disk along with the instance (YES), and select the image by clicking the arrow next to its name, as shown in Fig. 5.

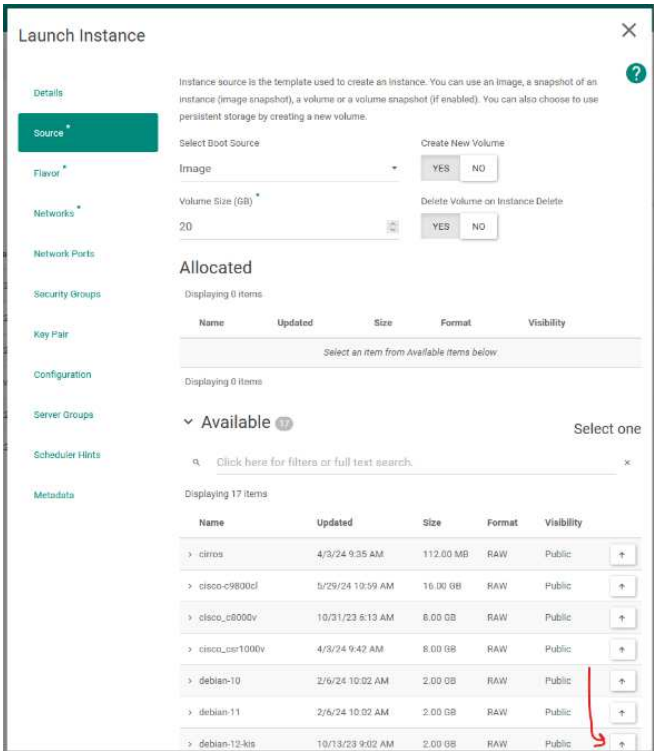


Fig. 5. OpenStack instance image selection

In the same way, we select “Flavor” – in our case, option 2c2r20d. We continue to select the network, where we select the int-net option, and the last part is to upload the script in the “Configuration” tab, where we select the given file or copy the script directly below, as shown in Fig. 6. Finally, we just click on the “Launch Instance” button.

After creating an instance, we can assign it a FloatingIP in the ”Project/Network/Floating IPs” section. Here, for the given

979-8-3315-7426-0/25/$31.00 ©2025 IEEE 430 Authorized licensed use limited to: University of Zilina. Downloaded on December 22,2025 at 10:38:11 UTC from IEEE Xplore. Restrictions apply.

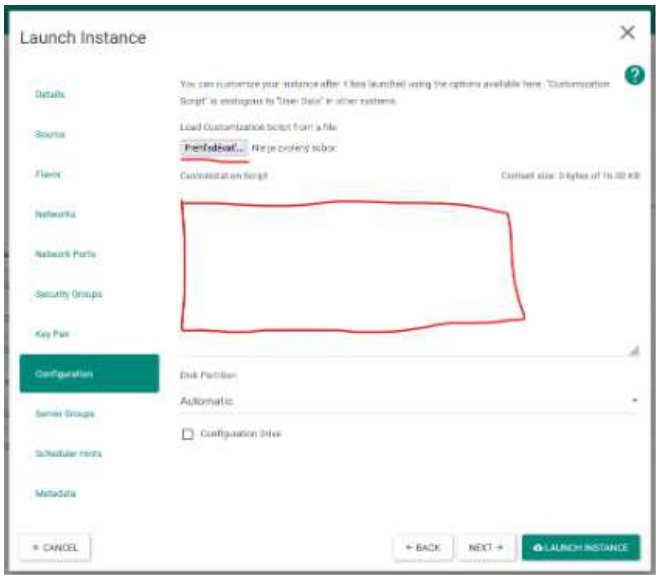


Fig. 6. OpenStack instance customization

IP address, click the "Associate" button and select the desired instance in the "Select a port" field. We can also create new FloatingIPs in this section of the OpenStack interface.

After a successful boot, users need to execute the prepared script, which is stored in $/etc/asterisk/asterisk_script.sh$. Here, we can create a new PBX configuration or add a new user if one already exists, as shown in Fig. 7. We have also considered executing the script after the first boot; however, it proved impractical because we were unable to set defaults for the desired variables. Further modifications could be added to allow the user to edit the configuration via the Web GUI.

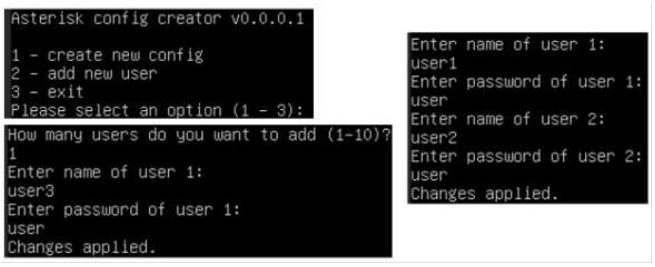


Fig. 7. Script menu for VoIPaaS configuration customization

V. Conclusion

In this article, we presented an overview of Asterisk and OpenStack, highlighting their key features and architectures. We discussed our efforts to implement Asterisk as a VoIP solution offered as a service. Initially, we attempted to create a modified Debian image with a preinstalled version of Asterisk, but encountered issues, including problems with unauthorized packages and installation errors related to library dependencies. Consequently, we opted for a simpler approach, utilizing a clean Debian image and automation scripts for installation and configuration.

The script we developed checks prerequisites and generates a basic configuration for Asterisk. Users can easily add one or more accounts by inputting the relevant names and passwords, and this script can be executed at any time.

Finally, we aimed to enhance the process further by integrating ManageIQ for automated instance deployment. Unfortunately, due to improper installation within our OpenStack environment, we were unable to deploy or test ManageIQ. However, we believe that, once correctly installed, it should function as intended.

VI. Acknowledgment

Funded by the EU NextGenerationEU through the Recovery and Resilience Plan for Slovakia under the project No. 09I05-03-V02-00010.

References

[1] Zakaria Benomar et al. "Enabling Secure RESTful Web Services in IoT using OpenStack". In: *2020 IEEE 17th International Conference on Mobile Ad Hoc and Sensor Systems (MASS)*. 2020, pp. 410–417. DOI: 10.1109/MASS50613.2020.00057.

[2] Benjamin Koné and Ahmed Dooguy Kora. "Management and Orchestration for Network Function Virtualization in a VoIP Testbed: A Multi-domain Case". In: *2021 44th International Conference on Telecommunications and Signal Processing (TSP)*. 2021, pp. 372–376. DOI: 10.1109/TSP52935.2021.9522622.

[3] Benjamin Koné et al. "Use Case of a Management and Orchestration for Network Functions Virtualization in a VoIP Testbed". In: *2021 IEEE International Black Sea Conference on Communications and Networking (BlackSeaCom)*. 2021, pp. 1–5. DOI: 10.1109/BlackSeaCom52164.2021.9527877.

[4] Akanksha et al. "Cloud-Hosted IP-PBX System with Asterisk and Oracle Cloud: A Scalable and Cost-Effective Solution for Business Communications". In: *2024 2nd International Conference on Disruptive Technologies (ICDT)*. 2024, pp. 837–841. DOI: 10.1109/ICDT61202.2024.10489415.

[5] Cai-Sian Jhuang and Quincy Wu. "Container-Based VoIP Failover Design on Kubernetes Overlay Networks". In: *2022 International Conference on Computational Science and Computational Intelligence (CSCI)*. 2022, pp. 1317–1322. DOI: 10.1109/CSCI58124.2022.00237.

[6] ByungRae Cha, Sun Park, and JongWon Kim. "Experimental Verification of Docker-Based Distributed Cloud Applications for Secured Mobile VoIP". In: *2016 10th International Conference on Complex, Intelligent, and Software Intensive Systems (CISIS)*. 2016, pp. 585–589. DOI: 10.1109/CISIS.2016.100.

979-8-3315-7426-0/25/$31.00 ©2025 IEEE 431
Authorized licensed use limited to: University of Zilina. Downloaded on December 22,2025 at 10:38:11 UTC from IEEE Xplore. Restrictions apply.

[7] Aklilu Daniel Tesfamicael, Vicky Liu, and William Caelli. “Design and Implementation of Unified Communications as a Service Based on the Open Stack Cloud Environment”. In: *2015 IEEE International Conference on Computational Intelligence Communication Technology*. 2015, pp. 117–122. DOI: 10.1109/CICT.2015.133.

[8] *Getting Started with Asterisk*. URL: https://www.asterisk.org/get-started/.

[9] *Home - This is the home of the official documentation for The Asterisk Project*. URL: https://docs.asterisk.org/.

[10] *Asterisk Architecture*. URL: https://docs.asterisk.org/Fundamentals/Asterisk-Architecture/Asterisk-Architecture-The-Big-Picture/.

[11] *Asterisk docs - a big picture of Asterisk Architecture*. URL: https://docs.asterisk.org/Fundamentals/Asterisk-Architecture/bigpicture.png.

[12] *Asterisk docs - Migrating from $chan_siptores_pjsip$*. URL: https://docs.asterisk.org/Configuration/Channel-Drivers/SIP/Configuring-res_pjsip/Migrating-from-chan_sip-to-res_pjsip/.

[13] *What is Openstack*. URL: https://www.openstack.org/software/.

[14] Victor González Chamorro, Carlos Nuñez Castillo, and Fabio Lopez-Pires. “An Elastic VoIP Solution Based on OpenStack”. In: *2016 International Conference on Information Systems Engineering (ICISE)*. 2016, pp. 43–47. DOI: 10.1109/ICISE.2016.8.

[15] Aklilu Daniel Tesfamicael, Vicky Liu, and William Caelli. “Design and Implementation of Unified Communications as a Service Based on the Open Stack Cloud Environment”. In: *2015 IEEE International Conference on Computational Intelligence Communication Technology*. 2015, pp. 117–122. DOI: 10.1109/CICT.2015.133.

[16] Santiago J. Arévalo-Cordero et al. “Data traffic management in a hybrid cloud composed of Openstack and Azure”. In: *2019 IEEE Colombian Conference on Communications and Computing (COLCOM)*. 2019, pp. 1–6. DOI: 10.1109/ColComCon.2019.8809158.

[17] *Installing Asterisk From Source*. URL: https://docs.asterisk.org/Getting-Started/Installing-Asterisk/Installing-Asterisk-From-Source/What-to-Download/.

[18] *ManageIQ Anand Has Been Released*. URL: https://www.redhat.com/fr/blog/manageiq-anand-has-been-released.

[19] *ManageIQ Get Started*. URL: https://www.manageiq.org/docs/get-started/index.html.

[20] Luis Tomas Bolivar et al. “On the Deployment of an Open-Source, 5G-Aware Evaluation Testbed”. In: *2018 6th IEEE International Conference on Mobile Cloud Computing, Services, and Engineering (MobileCloud)*. 2018, pp. 51–58. DOI: 10.1109/MobileCloud.2018.00016.

979-8-3315-7426-0/25/$31.00 ©2025 IEEE University of Zilina. Downloaded on December 22,2025 at 10:38:11 UTC from IEEE Xplore. Restrictions apply.