УДК 621.397

В.І. МАГРО
Дніпровський національний університет імені Олеся Гончара
С.В. ПЛАКСІН
Інститут транспортних систем і технологій «Трансмаг» Національної академії наук України
В.О. СВЯТОШЕНКО
Національний технічний університет «Дніпровська політехніка»

ПОБУДОВА МОДЕЛІ ТЕСТУВАННЯ І МОНІТОРИНГУ В МІКРОСЕРВІСНІЙ ІНФРАСТРУКТУРІ

Розглянуто методику створення інфраструктури, яка використовує функції тестування і моніторингу медіасервісних систем. Дана методика дозволяє формувати критерії із оптимального вибору медіасервісних систем та їх побудови. Особливістю методики є розміщення досліджуваного сервісу в віртуальний контейнер, де проводиться дослідження його поведінки в залежності від навантаження. В якості платформи для управління контейнерами обраний Kubernetes. При цьому використовувались тільки проєкти з відкритим вихідним кодом. Це забезпечує доступність методики, а за необхідності дозволяє коригувати код або доповнювати його новими функціональними можливостями. Результати експерименту показали, що обрана платформа для «оркестрації» контейнерів (Kubernetes) добре масштабується для управління будь-якими існуючими медіасервісами, що мають відкритий вихідний код. Дослідження показали, що використання технології WebRTC дозволяє знизити навантаження на серверну частину медіасервісів. Використання автоматизації конфігурацій медіасервісної інфраструктури дозволяє встановлювати як окремі компоненти, так і всю структуру цілком. Автоматизація реалізована з використанням Ansible і bash скриптинга. Результат впровадження автоматизації установки і конфігурації компонентів системи забезпечують легку повторюваність інфраструктури. Необхідна для тестування інфраструктура розгорнута в своєму середовищі, що виключає такі недоліки як залежність від постачальника послуг, а також його контроль та впровадження обмежень. В екстрених випадках доступ до медіасервісу може бути організований локально, без доступу в Інтернет на глобальному або національному рівні. Блочність запропонованої інфраструктури робить її гнучкою у використанні. Таким чином, запропонована методика дозволяє оцінити можливість використання даної медіасервісної платформи при заданому навантаженні і конкретній конфігурації апаратної платформи. Запропонована методика дозволяє сформувати вимоги до необхідної апаратної платформи для функціонування конкретних медіасервісних платформ.

Ключові слова: контейнер, мікросервісна інфраструктура, оркестрація, потоковий сервіс, медіасервісна платформа, WebRTC, Kubernetes, Ansible, автоматизація тестування.

В.И. МАГРО
Днипровский национальный университет имени Олеся Гончара
С.В. ПЛАКСИН
Институт транспортных систем и технологий «Трансмаг» Национальной академии наук Украины
В.О. СВЯТОШЕНКО
Национальный технический университет «Днипровская политехника»

ПОСТРОЕНИЕ МОДЕЛИ ТЕСТИРОВАНИЯ И МОНИТОРИНГА В МИКРОСЕРВИСНОЙ ИНФРАСТРУКТУРЕ

Рассмотрена методика создания инфраструктуры, которая использует функции тестирования и мониторинга медиасервисных систем. Данная методика позволяет формировать критерии по оптимальному выбору медиасервисных систем и их построению. Особенностью методики является размещение исследуемого сервиса в виртуальный контейнер, где производится исследование его поведения в зависимости от нагрузки. В качестве платформы для управления контейнерами выбран Kubernetes. При этом использовались только проекты с открытым исходным кодом. Это обеспечивает доступность методики, а при необходимости позволяет корректировать код или дополнять его новыми функциональными возможностями. Результаты эксперимента показали, что выбранная платформа для «оркестрации» контейнеров (Kubernetes) хорошо масштабируется для управления любыми существующими медиасервисами, имеющими открытый исходный код. Исследования показали,

https://doi.org/10.32782/KNTU2618-0340/2021.4.2.1.16

что использование технологии WebRTC позволяет снизить нагрузку на серверную часть медиасервисов. Использование автоматизации конфигураций медиасервисной инфраструктуры позволяет устанавливать как отдельные компоненты, так и всю структуру целиком. Автоматизация реализована с использованием Ansible и bash скриптинга. Результат внедрения автоматизации установки и конфигурирования компонентов системы обеспечивают легкую повторяемость инфраструктуры. Необходимая для тестирования инфраструктура развернута в собственной среде, что исключает такие недостатки как зависимость от поставщика услуг, а также его контроль и внедрение ограничений. В экстренных случаях доступ к медиасервисам может быть организован локально, без доступа в Интернет на глобальном или национальном уровне. Блочность предлагаемой инфраструктуры делает ее гибкой по использованию. Таким образом, предлагаемая методика позволяет оценить возможность использования данной медиасервисной платформы при заданной нагрузке и конкретной конфигурации аппаратной платформы. Предлагаемая методика позволяет сформировать требования к необходимой аппаратной платформе для функционирования конкретных медиасервисных платформ.

Ключевые слова: контейнер, микросервисная архитектура, оркестрация, поточный сервис, медиасервисная платформа, WebRTC, Kubernetes, Ansible, автоматизация тестирования.

V.I. MAGRO
Oles Honchar Dnipro National University
S.V. PLAKSIN
Institute of Transport Systems and Technologies of the National Academy of Sciences of Ukraine
V.O. SVYATOSHENKO
National Technical University “Dnipro Polytechnic”

BUILDING A TEST AND MONITORING MODEL IN A MICROSERVICE INFRASTRUCTURE

The technique of creating an infrastructure is considered, which uses the functions of testing and monitoring media service systems and helps to form criteria for their selection or construction. The peculiarity of the technique lies in placing the service under investigation in a virtual container and examining its behavior depending on the load. Kubernetes was chosen as the platform for managing containers. Only open source projects were used as individual elements. This gives accessibility and, if necessary, to correct the code or supplement it with new functionality. The experiment results showed that the chosen container orchestration platform (Kubernetes) scales well to manage any existing open source media services. This gives accessibility and, if necessary, to correct the code or supplement it with new functionality. The experiment results showed that the chosen container orchestration platform (Kubernetes) scales well to manage any existing open source media services. Studies have also shown that the use of WebRTC technology can reduce the load on the server side of media services. Using the automation of configurations of the media service infrastructure allows you to install both individual components and its entirety. The automation was implemented using Ansible and bash scripting. The result of the implementation of automation of installation and configuration of system components is an easy repeatability of the infrastructure. The infrastructure required for testing was deployed in its own environment, which eliminates such disadvantages as dependence on a service provider, as well as its control and imposed restrictions. In case of emergency, access to media services can be organized locally, without access to the Internet at the global or national level. The blockiness of the proposed infrastructure makes it flexible in use. Thus, the proposed methodology makes it possible to assess the possibility of using this media service platform for a given load and a specific configuration of the hardware platform. The proposed methodology makes it possible to formulate requirements for the required hardware platform for the functioning of specific media service platforms.

Keywords: container, microservice architecture, orchestration, streaming service, media service platform, WebRTC, Kubernetes, Ansible, test automation.

Формулювання проблеми

Однією з важливих відмінних особливостей сучасних телекомунікаційних систем є необхідність створення якомога більш придатного середовища для розгортання тих чи інших додатків. Наприклад, в сучасних умовах карантину та самоізоляції, набувають більшої популярності медіаплатформи та програмне забезпечення для створення відеоконференцій. Важливою вимогою для ефективної

https://doi.org/10.32782/KNTU2618-0340/2021.4.2.1.16

роботи таких додатків є використання спеціально налаштованого для цього середовища.

Існуюче програмне забезпечення для проведення відеоконференцій на корпоративному рівні, як правило, потребує відповідного ліцензування. Для навчальних закладів та деяких підприємств така модель співпраці може бути невигідною. Більш того, гостро постає проблема конфіденційності даних, що передаються під час конференцій при використанні існуючих варіантів.

Ефективним методом в цьому випадку є розробка та впровадження власної платформи для розгортання системи відеоконференцій. Оптимальним варіантом впровадження цього методу є використання існуючих рішень з відкритим кодом. Об'єднання таких проектів та їх властивостей дозволяє досягти мети даного проекту, а саме створити високодоступне, масштабоване, надійне та гнучке середовище для розгортання систем відеоконференцій або додатків.

Аналіз останніх досліджень і публікацій

Тестування медіасерверної системи є складним завданням. Зазвичай для цього потрібна велика кількість тестових прикладів, значні ресурси і географічно розподілені сценарії використання [1]. Створення тестового середовища і досягнення певного рівня достовірності тестування це досить дороге задоволення. Щоб вирішити ці проблеми, тестові системи повинні бути економічними і масштабованими.

У ряді робіт обговорюються заходи в області масштабованої потокової передачі в реальному часі в мережі [2]. Пропонується набір випробувань, спрямованих на виявлення вузьких місць медіасерверних платформ. Відзначається, що для будь-якого медіа-сервера WebRTC в процесі тестування необхідно встановити можливості використання масштабованої бібліотеки при роботі з завданнями і діями протоколу ICE. В процесі тестування необхідно розглядати кілька сценаріїв.

Останнім часом Інтернет-спілкування в реальному часі (WebRTC) дуже швидко розвивається. Цей стандарт забезпечує незалежність від платформи аудіо/відео. Це комунікаційна платформа для веб-збірки прямо в браузер. Вона забезпечує повну реалізацію складного стека технологій, який включає в себе різні елементи, такі як: доставка контенту, обробка аудіо/відео, транспорт мультимедіа та контроль якості взаємодії, як для P2P, так і зв'язок з ретрансляцією через хмару [3]. При цьому необхідно проводити такі дослідження:

- вплив навантаження на серверну частину хмари;
- зниження навантаження для модулів вибіркового пересилання хмарного мультимедійного контенту;
- розробка схеми балансування навантаження для справедливого розподілу при тривалих сеансах зв'язку по декількох серверах;
- передбачуваність навантаження на сервер.

Медіа-сервери з підтримкою WebRTC (WebRTC SFU) широко використовуються для відеоконференцзв'язку та трансляції. При цьому необхідні дослідження п'яти основних SFU (Selective Forwarding Unit) WebRTC з відкритим вихідним кодом, що використовуються для відеоконференцзв'язку, під навантаженням [4]. Медіа-сервер WebRTC має набір API-інтерфейсів, спрямованих на спрощення розробки вдосконалених WebRTC додатків. Серед даних інтерфейсів API Kurento забезпечує високий рівень інфраструктурного тестування для оцінки сервісів WebRTC з точки зору функціональності, продуктивності і якості обслуговування [5, 6]. Під час налаштування системи особлива увага приділяється питанням масштабування і забезпечення якості (QoS, QoE) [7]. В роботі [8] розглядається простий підхід для не

https://doi.org/10.32782/KNTU2618-0340/2021.4.2.1.16

функціонального тестування сервісів WebRTC, що використовують відкриту платформу ElasTest для наскрізного тестування великих складних систем.

WebRTC – це загальний термін для декількох нових технологій, призначених для обміну мультимедійними даними в Інтернеті в режимі реального часу. Як і у випадку з іншими службами, пов'язаними з медіа, якість сприйняття зв'язку WebRTC можна виміряти за допомогою індикаторів якості сприйняття (QoE). Методи оцінки QoE можна розділити на суб'єктивні (оцінок) або об'єктивні (моделі, розраховані як функція різних параметрів). В роботі [9] основна увага приділяється методу VMAF (Video Multi-method Assessment Fusion), який являє собою нову повнофункціональну об'єктивну модель оцінки якості відео, розроблену Netix. VMAF зазвичай використовується для оцінки сервісів потокового відео. У даній роботі пропонується використання VMAF для оцінки WebRTC. Для цього використовуються добре відомі технології з відкритим вихідним кодом, такі як JUnit, Selenium, Docker і FFmpeg.

В роботі [10] пропонується концепція «оркестровки тестів» медіасерверних платформ, яка сприймається як новий спосіб вибору, порядку і виконання паралельно груп тестів. Оркестровку тестів можна вважати процесом, в якому організовані різні тестові приклади, зібрані і виконані по топології, яка визначає одночасне стрес-тестування платформ.

Мета дослідження

Метою роботи є створення інфраструктури що використовує сучасні приватні хмарні технологій та мінімальні недоліки в порівнянні з вже існуючими технічними реалізаціями.

В даній роботі розглянуто дві задачі: розробка методики тестування і моніторингу медіасервісної системи та формування критеріїв вибору ефективності медіасервісних систем.

Викладення основного матеріалу дослідження

Програмним забезпеченням на якому базується увесь принцип роботи розробленої системи є Kubernetes.

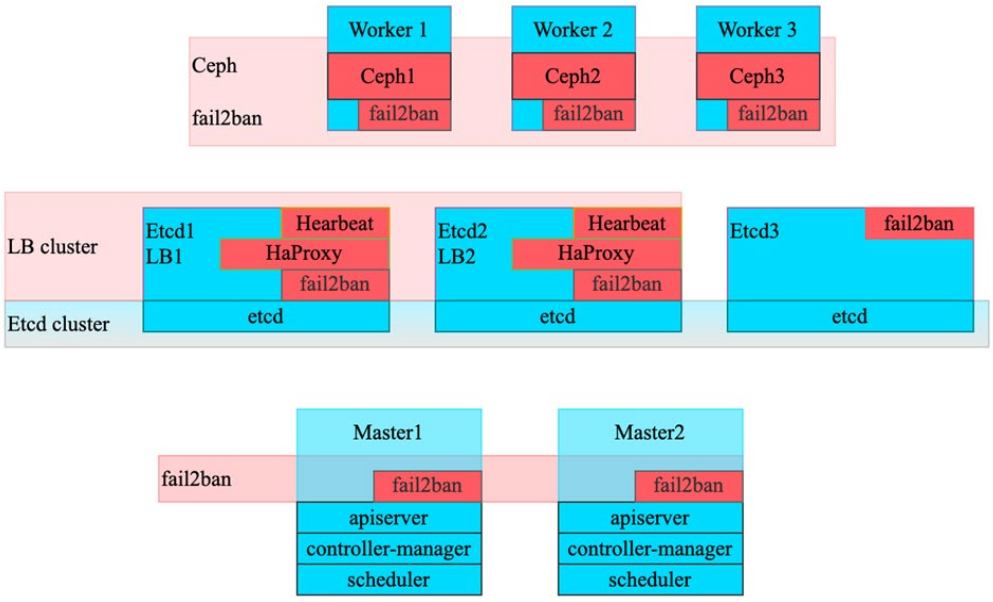


Рис. 1. Схема компонентів мультисервісної системи

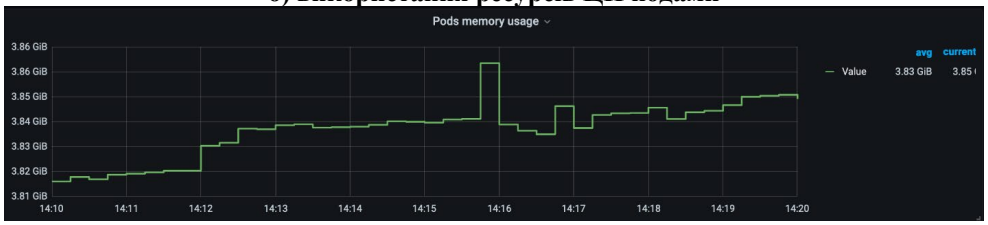
https://doi.org/10.32782/KNTU2618-0340/2021.4.2.1.16

На рис. 1 наведена схема компонентів системи що пропонується. Вона включає як стандартні компоненти (які, як правило, реалізовані у всіх системах що побудовані на базі Kubernetes), так і елементи, котрі були впроваджені в процесі розробки інфраструктури з метою підвищення якостей, властивостей та переваг розробленої мультисервісної системи, а також зменшення її недоліків в порівнянні з існуючими до неї аналогами. HaProxy – балансувальник вхідного навантаження, що надходить та розподіляється між Etcd вузлами; Heartbeat – інструмент, що підвищує стійкість системи та призначений для виключення наслідків виходу з ладу одного с вузлів управління; Fail2ban – програмне забезпечення, яке встановлене на усіх вузлах розробленої системи з метою підвищення їх безпеки; Ceph – розподілене сховище об’єктів та реплікації даних, що зберігаються.

Розроблена система надає ряд переваг, таких як надійне сховище, високодоступність, балансування навантаження, безпека та моніторинг. Система надає придатне середовище не тільки для розгортання тих чи інших додатків, але і для моніторингу та контролю споживання ресурсів системою та розгорнутими в ній додатками. Більш того, дана система надає можливість збору та аналізу вихідних параметрів цих додатків, наприклад за допомогою візуального аналізу графіків.

а) використання ресурсів системою

б) використання ресурсів ЦП подами



в) використання ресурсів фізичної пам’яті подами

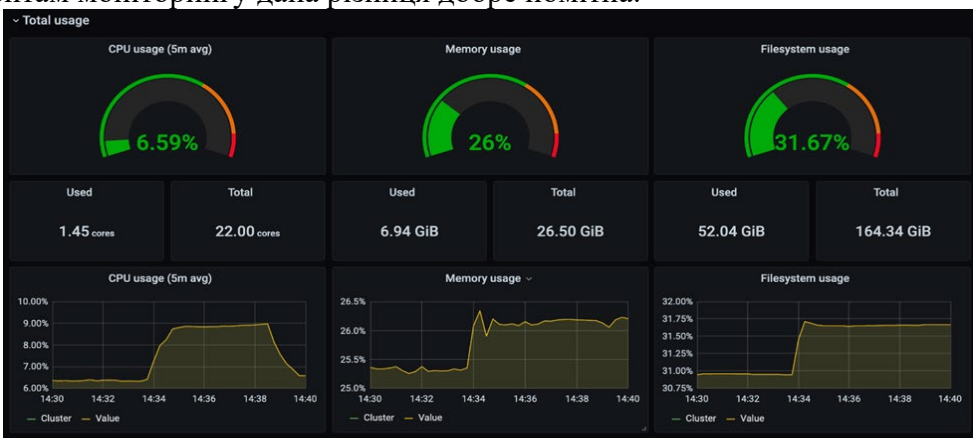
Рис. 2. Панелі моніторингу у стані «спокою»

На рис. 2 зображені панелі для моніторингу використання ресурсів. В цьому випадку, система знаходиться у стані “спокою”, іншими словами, ресурси використовуються тільки на підтримку роботи її компонентів.

https://doi.org/10.32782/KNTU2618-0340/2021.4.2.1.16

Для тесту роботи системи, в ній була розгорнута медіаплатформа Kurento. Ця платформа реалізує технологію WebRTC.

На рис. 3 видно як змінилися показники вихідних параметрів. Судячи з графіків, після розгортання Kurento додатку підвищився рівень завантаженості центрального процесора, а також оперативної пам'яті, що використовується. Завдяки впровадженим інструментам моніторингу дана різниця добре помітна.



а) використання ресурсів системою у стані «навантаження»

б) використання ресурсів ЦП подами у стані «навантаження»

в) використання ресурсів фізичної пам’яті подами у стані «навантаження»

Рис. 3. Панелі моніторингу у стані «навантаження»

Висновки

Запропоновано методику побудови мультисервісної системи для тестування та моніторингу розгорнутих в ній мережевих сервісів. Одним з прикладів такого програмного забезпечення є медіасервісні платформи. У якості тестової реалізована платформа Kurento. Вона розгорнута, протестовані її потреби в ресурсах. Використання мікросервісної інфраструктури, індивідуального середовища з принципами хмарних технологій для її розгортання, а також «найкращих практик» дозволило розробити високодоступну, масштабовану, надійну та масштабовану систему, яка придатна для тестування медіаплатформ. В роботі розроблено таке:

1. Реалізована політика безпеки, що спрямована на запобігання випадковому перезапису, видаленню та змінам. Для впровадження такої політики використовується RBAC (Role Based Access Control).
2. Побудовано швидкий доступ до архітектури. Це досягається за рахунок відсутності єдиної точки збірки у всій архітектурі, а також за рахунок того що розроблена система містить надлишкові елементи. Завдяки цьому досягається висока стійкість системи у випадку збою.

https://doi.org/10.32782/KNTU2618-0340/2021.4.2.1.16

3. Для ефективності роботи системи використані останні стабільні версії програмного забезпечення і конфігурацій компонентів архітектури. Це, в свою чергу, підвищує безпеку системи.
4. Впровадження автоматизації надає можливість управління конфігурацією, оркестровки, централізованої установки додатків та паралельного виконання типових завдань на групі систем. В даному випадку, автоматизація реалізована на основі Ansible і bash скріптингу. Результатом впровадження автоматизації розгортки компонентів системи є легко відновлювальна інфраструктура.
5. Необхідна інфраструктура розгорнута в індивідуальному середовищі, що виключає такі недоліки як прив'язаність до постачальника послуг, а також його контроль та впровадження обмежень. В екстреному випадку, доступ до мережі може бути організовано локально, без доступу через Інтернет.

Таким чином, розроблена система, яка представляє собою набір окремих блоків, кожен з яких включає в себе ті чи інші функціональні особливості та інструменти для виконання заявлених завдань.

Потребує наступних досліджень автоматичне формування віртуальних користувачів та розгортання необхідної їх кількості з метою тестування здатності навантаження статичних конфігурацій обладнання для медіа сервісів (або формування необхідних конфігурацій обладнання для заданого навантаження).

Список використаної літератури

1. BaiX., Li M., Huang X., Tsai WT., Gao J. Vee@Cloud: The virtual test lab on the cloud. *IEEE 8th international workshop on automation of software test (AST)*. 2013. Pp. 15-18. DOI: 10.1109/IWAST.2013.6595785
2. Amirante A., Castaidi T., Miniero L., Romano S.P., Toppi A. Measuring Janus temperature in ICE-land. *Proceedings of IEEE conference "Principles, Systems and Applications of IP Telecommunications (IPTComm)"*. 2018. DOI: 10.1109/IPTCOMM.2018.8567641.
3. Xhagjika V., Escoda O.D., Navarro L., Vlassov V. Media Streams Allocation and Load Patterns for a WebRTC Cloud Architecture. *IEEE 8th International conference on the network of the future (NOF)*. 2017. DOI 10.1109/NOF.2017.8251214.
4. Andrer E., Nicolas Breton∗N., Lemesle A., Roux L., Gouaillard A. Comparative Study of WebRTC Open Source SFUs for Video Conferencing. *Proceedings of IEEE conference "Principles, Systems and Applications of IP Telecommunications (IPTComm)"*. 2018. DOI: 10.1109/IPTCOMM.2018.8567642.
5. Garcна B., Lypez-Fernandez L., Gallego M., Gortazar F. Testing Framework for WebRTC Services. *Proceedings of the 9th EAI International Conference on Mobile Multimedia Communications.* 2018, Pp. 40–47. DOI: 10.4108/eai.18-6-2016.2264212.
6. Garcна B., Lypez-Fernandez L., Gallego M. Gortazar F. Analysis of video quality and end-to-end latency in WebRTC. *IEEE Globecom Workshops (GC Wkshps)*. 2016. DOI: 10.1109/GLOCOMW.2016.7848838.
7. Garcна B., Lypez-Fernandez L., Gallego M. & Paris M. WebRTC Testing: Challenges and Practical Solutions. IEEE Communications Standards Magazine. 2017. Vol. 1. Issue 2. Pp. 36-42. DOI: 10.1109/MCOMSTD.2017.1700005.
8. Bertolino A., Calabry A., Angelis G. D., Gortazar F., Lonetti F., Maes M., Tucyn G. Quality-of-Experience driven configuration of WebRTC services through automated testing. *IEEE 20th International Conference on Software Quality, Reliability and Security (QRS)*. 2020. Pp. 152-159. DOI: 10.1109/QRS51102.2020.00031.

https://doi.org/10.32782/KNTU2618-0340/2021.4.2.1.16

9. Garcна B., Lypez-Fernandez L., Gortazar F., Gallego M. Practical Evaluation of VMAF Perceptual Video Quality for WebRTC Applications. *Electronics.* 2019. No. 8. Pp. 854-869. DOI: 10.3390/electronics808085.
10. Garcна B., Lypez-Fernandez L., Gallego M., Miranda B., Jimeґnez E., Angelis G. D., Santos C., Marchetti E. Proposal to Orchestrate Test Cases. *IEEE 11th International Conference on the Quality of Information and Communications Technology (QUATIC)*. 2018. Pp. 38-46. DOI 10.1109/QUATIC.2018.00016.

Reference

1. Bai, X., Li, M., Huang, X., Tsai, W. T., & Gao, J. (2013). Vee@Cloud: The virtual test lab on the cloud. *IEEE 8th international workshop on automation of software test (AST)*. 15-18. DOI: 10.1109/IWAST.2013.6595785
2. Amirante, A., Castaidi, T., Miniero, L., Romano S. P., & Toppi, A. (2018). Measuring Janus temperature in ICE-land. *Proceedings of IEEE conference "Principles, Systems and Applications of IP Telecommunications (IPTComm)"*. DOI: 10.1109/IPTCOMM.2018.8567641.
3. Xhagjika, V., Escoda, O. D., Navarro, L., & Vlassov, V. (2017). Media Streams Allocation and Load Patterns for a WebRTC Cloud Architecture. *IEEE 8th International conference on the network of the future (NOF)*. DOI 10.1109/NOF.2017.8251214.
4. Andrer, E., Nicolas Breton∗N., Lemesle, A., Roux, L., & Gouaillard, A. (2018). Comparative Study of WebRTC Open Source SFUs for Video Conferencing. *Proceedings of IEEE conference "Principles, Systems and Applications of IP Telecommunications (IPTComm)"*. DOI: 10.1109/IPTCOMM.2018.8567642.
5. Garcна, B., Lypez-Fernandez, L., Gallego, M., & Gortazar, F. (2016). Testing Framework for WebRTC Services. *Proceedings of the 9th EAI International Conference on Mobile Multimedia Communications.* 40–47. DOI: 10.4108/eai.18-6-2016.2264212.
6. Garcна, B., Lypez-Fernandez, L., Gallego, M., & Gortazar, F. (2016). Analysis of video quality and end-to-end latency in WebRTC. *IEEE Globecom Workshops (GC Wkshps)*. DOI: 10.1109/GLOCOMW.2016.7848838.
7. Garcна, B., Lypez-Fernandez, L., Gallego, M. , & Paris, M. (2017). WebRTC Testing: Challenges and Practical Solutions. *IEEE Communications Standards Magazine*. **1**, 2, 36-42. DOI: 10.1109/MCOMSTD.2017.1700005.
8. Bertolino, A., Calabry, A., Angelis, G. D., Gortazar, F., Lonetti, F., Maes, M., & Tucyn, G. (2020). Quality-of-Experience driven configuration of WebRTC services through automated testing. *IEEE 20th International Conference on Software Quality, Reliability and Security (QRS)*. 152-159. DOI: 10.1109/QRS51102.2020.00031.
9. Garcна, B., Lypez-Fernandez, L., Gortazar, F., & Gallego, M. (2019). Practical Evaluation of VMAF Perceptual Video Quality for WebRTC Applications. *Electronics.* **8**, 854-869. DOI: 10.3390/electronics808085.
10. Garcна, B., Lypez-Fernandez, L., Gallego, M., Miranda, B., Jimeґnez, E., Angelis, G. D., Santos, C., & Marchetti, E. (2018). Proposal to Orchestrate Test Cases. *IEEE 11th International Conference on the Quality of Information and Communications Technology (QUATIC)*. 38-46. DOI 10.1109/QUATIC.2018.00016.

Магро Валерій Іванович – канд. фіз.-мат. наук, доцент, доцент кафедри прикладної радіофізики, електроніки та наноматеріалів Дніпровського національного університету імені Олеся Гончара. E-mail: magrov@i.ua, ORCID: 0000-0003-4238-6733.

https://doi.org/10.32782/KNTU2618-0340/2021.4.2.1.16

Плаксін Сергій Вікторович – доктор фіз.-мат. наук, старший науковий співробітник, завідувач відділу систем керування інституту транспортних систем і технологій «Трансмаг» Національної академії наук України. E-mail: svp@westa-inter.com, ORCID: 0000-0001-8302-0186.

Святошенко Володимир Олександрович – старший викладач Національного технічного університету «Дніпровська політехніка». E-mail: svyt22@gmail.com, ORCID: 0000-0003-4027-5706.

https://doi.org/10.32782/KNTU2618-0340/2021.4.2.1.16