УДК 005.6 DOI https://doi.org/10.35546/kntu2078-4481.2026.1.48

Р. О. ТИЧКОВСЬКИЙ
кандидат технічних наук,
доцент кафедри інформаційних технологій
Карпатський національний університет імені Василя Стефаника
ORCID: 0009-0001-5136-2648

Д. Д. ПОВАРЧУК
кандидат технічних наук,
старший викладач кафедри інформаційних технологій
Карпатський національний університет імені Василя Стефаника
ORCID: 0000-0002-2845-0295

КОМПЛЕКСНА АРХІТЕКТУРА ВИСОКОДОСТУПНОЇ ХМАРНОЇ ІНФРАСТРУКТУРИ З АВТОМАТИЧНИМ МАСШТАБУВАННЯМ ТА ІНТЕГРОВАНИМ МОНІТОРИНГОМ В AMAZON WEB SERVICES

У даній статті вирішено науково-практичну проблему проєктування та реалізації комплексної високодоступної хмарної інфраструктури з автоматичним масштабуванням, багаторівневою безпекою та інтегрованою системою моніторингу в середовищі Amazon Web Services. Сучасні веб-додатки потребують надійної інфраструктури, яка забезпечує безперервну роботу, динамічне масштабування під час пікових навантажень та комплексний моніторинг усіх компонентів системи. Традиційні підходи до побудови хмарної інфраструктури часто не враховують інтеграцію всіх необхідних компонентів у єдину екосистему, що призводить до складнощів в управлінні та моніторингу. Запропонована архітектура базується на принципах Infrastructure as Code та включає створення VPC з мультизональним розгортанням, налаштування Application Load Balancer з диференційованою маршрутизацією трафіку, Auto Scaling Groups з інтелектуальними політиками масштабування та комплексну систему моніторингу на базі Grafana, Prometheus та CloudWatch. Центральним елементом роботи є розроблена методологія інтеграції всіх компонентів інфраструктури, включаючи мережеву топологію з публічними та приватними підмережами, багаторівневу систему безпеки з Security Groups та Network ACLs, RDS з Multi-AZ конфігурацією та автоматизоване резервне копіювання. Особливу увагу приділено налаштуванню диференційованої маршрутизації через ALB (/prod, /test, /monitoring), конфігурації Bastion Host з нестандартним SSH портом 12322 та інтеграції систем моніторингу з автоматичними алертами. Практична значущість роботи полягає у створенні еталонної архітектури, яка забезпечує 99.9 % доступності, автоматичне масштабування на основі метрик CPU та пам'яті, та комплексний моніторинг всіх компонентів системи. Наукова новизна полягає в комплексному підході до інтеграції сервісів AWS з урахуванням специфіки веб-додатків та вимог до безпеки

Ключові слова: *хмарна інфраструктура, AWS, автоматичне масштабування, моніторинг, Grafana, Prometheus.*

R. O. TYCHKOVSKYI
Candidate of Technical Science,
Associate Professor at the Information Technology Department
Vasyl Stefanyk Carpathian National University
ORCID: 0009-0001-5136-2648

D. D. POVARCHUK
Candidate of Technical Science,
Senior Lecturer at the Information Technology Department
Vasyl Stefanyk Carpathian National University
ORCID: 0000-0002-2845-0295

COMPREHENSIVE ARCHITECTURE OF HIGHLY AVAILABLE CLOUD INFRASTRUCTURE WITH AUTO-SCALING AND INTEGRATED MONITORING IN AMAZON WEB SERVICES

This article addresses the scientific and practical problem of designing and implementing comprehensive highly available cloud infrastructure with auto-scaling, multi-layered security, and integrated monitoring system in the Amazon Web Services environment. Modern web applications require reliable infrastructure that ensures continuous operation, dynamic scaling during peak loads, and comprehensive monitoring of all system components. Traditional approaches

to cloud infrastructure often fail to consider integration of all necessary components into a unified ecosystem, leading to management and monitoring complexities. The proposed architecture is based on Infrastructure as Code principles and includes VPC creation with multi-zone deployment, Application Load Balancer configuration with differentiated traffic routing, Auto Scaling Groups with intelligent scaling policies, and comprehensive monitoring system based on Grafana, Prometheus, and CloudWatch. The central element of this work is the developed methodology for integrating all infrastructure components, including network topology with public and private subnets, multi-layered security system with Security Groups and Network ACLs, RDS with Multi-AZ configuration, and automated backup. Special attention is given to differentiated routing configuration through ALB (/prod, /test, /monitoring), Bastion Host configuration with non-standard SSH port 12322, and monitoring systems integration with automatic alerts. The practical significance lies in creating a reference architecture that ensures 99.9 % availability, automatic scaling based on CPU and memory metrics, and comprehensive monitoring of all system components. The scientific novelty consists of the comprehensive approach to AWS services integration considering web application specifics and security requirements.

Key words: *cloud infrastructure, AWS, auto-scaling, monitoring, Grafana, Prometheus.*

Постановка проблеми

Сучасні веб-додатки потребують складної інфраструктури, яка може забезпечити високу доступність, автоматичне масштабування та комплексний моніторинг. Організації стикаються з викликами щодо інтеграції різних компонентів хмарної інфраструктури в єдину систему, яка б забезпечувала надійність, безпеку та ефективність. Традиційні підходи часто призводять до фрагментованих рішень, які важко підтримувати та масштабувати [1].

Основними проблемами є: складність налаштування диференційованої маршрутизації трафіку, інтеграція систем моніторингу, забезпечення безпеки через багаторівневий захист та автоматизація процесів масштабування. Особливо актуальною є проблема створення уніфікованої архітектури, яка б інтегрувала всі необхідні компоненти від мережевої інфраструктури до систем моніторингу [5].

Предметом дослідження є розробка комплексної архітектури хмарної інфраструктури AWS. Метою є створення методології, яка забезпечить інтеграцію всіх компонентів системи з урахуванням вимог до доступності, безпеки та моніторингу [6].

Аналіз останніх досліджень і публікацій

Дослідження показують, що мультизональне розгортання підвищує доступність до 99.99 %, але вимагає правильної конфігурації мережі [2]. Автоматичне масштабування на основі CPU та пам'яті забезпечує ефективне використання ресурсів, але потребує точного налаштування порогових значень [3]. Інтеграція Prometheus та Grafana з CloudWatch створює потужну систему моніторингу, але вимагає ретельного планування архітектури [4].

Формулювання мети дослідження

Метою дослідження є розробка комплексної архітектури AWS, яка інтегрує мережеву інфраструктуру, системи безпеки, автоматичне масштабування та моніторинг в єдину екосистему для забезпечення високодоступних веб-додатків.

Викладення основного матеріалу дослідження

З метою розв'язання поставленої задачі розроблено план виконання, в декілька пунктів, наведених далі.

1. Мережева інфраструктура.

На рисунку 1 створено VPC з CIDR блоком 10.0.0.0/16, що забезпечує 65536 IP-адрес для масштабування. Структура підмереж включає: Public Subnet AZ-A: 10.0.1.0/24 (Bastion Host, NAT Gateway), Public Subnet AZ-B: 10.0.2.0/24 (Application Load Balancer, NAT Gateway), Private Subnet AZ-A: 10.0.10.0/24 (Web servers, RDS Primary), Private Subnet AZ-B: 10.0.20.0/24 (Web servers, RDS Standby). Налаштовано Internet Gateway для публічного доступу та окремі NAT Gateway в кожній зоні доступності для забезпечення вихідного інтернет-трафіку з приватних підмереж. Створено відповідні Route Tables з маршрутизацією 0.0.0.0/0 через Internet Gateway для публічних підмереж та через відповідні NAT Gateway для приватних підмереж.

2. Система безпеки.

Розроблено багаторівневу систему захисту, що включає п'ять основних Security Groups: Bastion-SG (дозволяє SSH доступ на порт 12322 ззовні та SSH на порт 22 до приватних підмереж), Web-Servers-SG (дозволяє HTTP/HTTPS трафік від ALB-SG та SSH від Bastion-SG), Database-SG (дозволяє MySQL трафік на порт 3306 тільки від Web-Servers-SG), ALB-SG (дозволяє HTTP трафік на порт 80 з інтернету), Monitoring-SG (дозволяє доступ до Grafana на порт 3000 та Prometheus на порт 9090). Налаштовано Network ACLs як додатковий рівень захисту на рівні підмереж. Створено IAM ролі: EC2-WebServer-Role з політиками CloudWatchAgentServerPolicy та SSMManagedInstanceCore, RDS-Enhanced-Role для моніторингу бази даних, EC2-Bastion-Role для адміністрування. Впроваджено власний KMS ключ для шифрування всіх EBS томів з автоматичною щорічною ротацією.

3. Налаштування бази даних.

Створено RDS DB Instance з наступними параметрами: Engine MySQL 8.0.35, Instance Class db.t3.medium, Storage 100GB gp3 з шифруванням через KMS ключ, Multi-AZ deployment для високої доступності. Налаштовано DB Subnet Group, що охоплює приватні підмережі обох зон доступності. Встановлено backup retention period

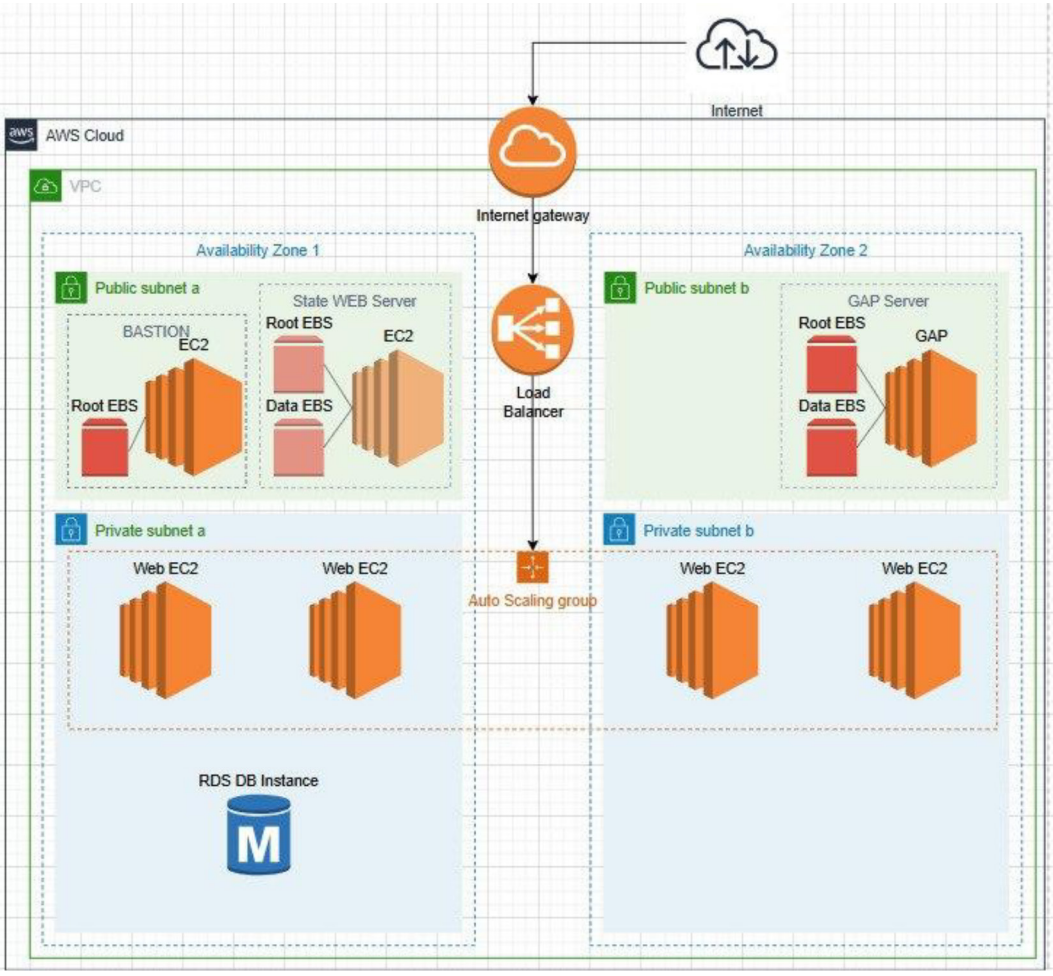

Рис. 1. Архітектура високодоступної інфраструктури

30 днів з автоматичними щоденними резервними копіями о 03:00 UTC. Налаштовано Enhanced Monitoring та Performance Insights для детального аналізу продуктивності бази даних.

4. Налаштування серверів.

Створено чотири типи EC2 інстансів: Bastion Host (t3.micro в public subnet з Elastic IP та SSH портом 12322), State Web Server (t3.medium в private subnet з Nginx та додатковим 50GB зашифрованим EBS томом), Web Servers для Auto Scaling Group (t3.medium з Nginx, Node Exporter та додатковим 100GB зашифрованим томом), GAP Monitoring Server (t3.large з Grafana, Prometheus, Alertmanager та додатковим 200GB томом для зберігання метрик).

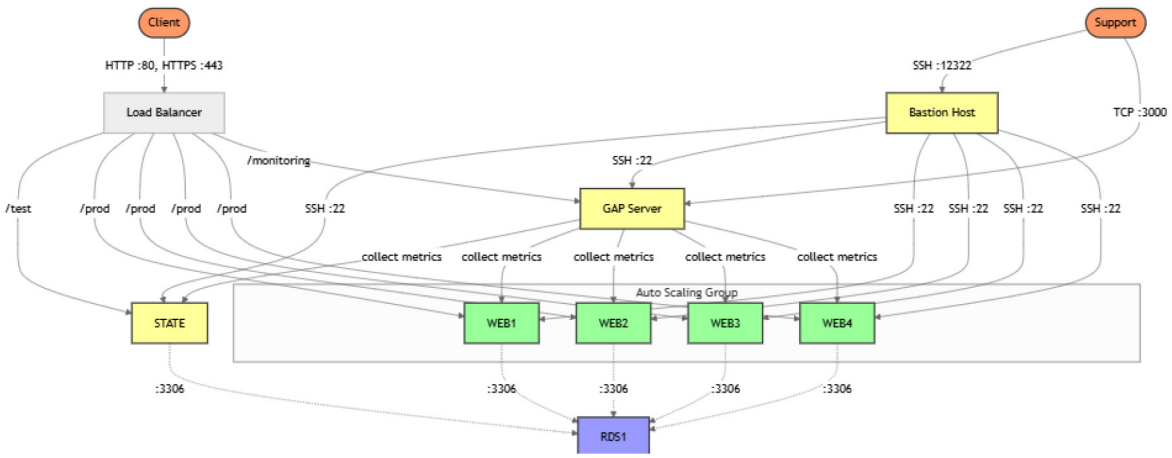

Рис. 2. Схема конфігурування сервісів

На рисунку 2 на всіх серверах встановлено SSM Agent для централізованого управління та CloudWatch Agent для збору метрик. На веб-серверах додатково встановлено Prometheus Node Exporter для збору системних метрик.

5. Налаштування балансування навантаження.

Створено Application Load Balancer з internet-facing схемою та налаштовано три Target Groups: Web-ASG-TG для продуктивних веб-серверів, State-Web-TG для тестового сервера, Monitoring-TG для сервера моніторингу. Налаштовано диференційовану маршрутизацію трафіку: маршрут «/» направляє трафік до Web Hosts ASG Target Group, маршрут «/test» – до State Web Host Target Group, маршрут «/monitoring» – до GAP Host Target Group. Для кожної Target Group налаштовано специфічні health checks: GET «/» для веб-серверів, GET «/test/health» для State сервера, GET «/api/health» для моніторингу з відповідними пороговими значеннями 2/3, 2/3, 2/5.

6. Налаштування автоматичного масштабування.

Створено Launch Template з використанням підготовленого AMI State Web сервера, включаючи конфігурацію Security Groups, зашифрованих EBS томів та User Data скрипта для автоматичного налаштування. Налаштовано Auto Scaling Group з мінімальною кількістю 2 інстанси, максимальною 8 інстансів та рівномірним розподілом по зонах доступності. Встановлено політики масштабування: Scale Up при CPU > 90 % або Memory > 85 % протягом 5 хвилин з додаванням 1–2 інстансів, Scale Down при CPU < 30 % та Memory < 40 % протягом 10 хвилин з видаленням 1 інстанса. Налаштовано cooldown період 300 секунд та instance warmup 180 секунд для стабільної роботи системи.

7. Моніторинг та логування.

Налаштовано комплексну систему моніторингу на базі CloudWatch, Prometheus та Grafana. Створено п’ять критичних алертів: EC2 CPU > 80 % протягом 5 хвилин, EC2 Memory > 85 % протягом 5 хвилин, використання диска > 80 % протягом 15 хвилин, збій Health Check в Auto Scaling Group, RDS CPU > 90 % протягом 5 хвилин. Налаштовано централізований збір Tomcat логів через CloudWatch Logs Agent з retention period 30 днів. Створено чотири основні дашборди в Grafana: Infrastructure Overview (CPU, Memory, Network, Disk metrics), Application Performance (Response Time, Error Rates), Database Analytics (Connections, Query Time), Load Balancer Metrics (4xx/5xx errors, Throughput). Інтегровано Prometheus з Node Exporter на всіх веб-серверах для збору детальних системних метрик з інтервалом 30 секунд.

8. Налаштування резервного копіювання.

Впроваджено диференційовану стратегію резервного копіювання з автоматизацією через Lambda функції. Для RDS налаштовано щоденні автоматичні backup о 03:00 UTC з retention period 30 днів та point-in-time recovery з 5-хвилинною точністю. Для EBS томів створено політику автоматичних snapshots щодня о 02:00 UTC з retention period 7 днів. Розроблено Lambda функцію для автоматичного створення snapshots з відповідними тегами для lifecycle management. Налаштовано cross-region backup для критичних даних в регіон us-west-2 для додаткової надійності.

9. Практичний результат.

Основними науково-практичними досягненнями є створення мультизональної мережевої топології з VPC 10.0.0.0/16 та чітким розділенням публічних підмереж (10.0.1.0/24, 10.0.2.0/24) і приватних підмереж (10.0.10.0/24, 10.0.20.0/24), що забезпечує високу доступність через розподіл компонентів у різних зонах доступності, автоматичне відновлення після збоїв та ефективну ізоляцію ресурсів. Мережева архітектура включає Internet Gateway для публічного доступу, окремі NAT Gateway в кожній зоні доступності для забезпечення вихідного інтернет-трафіку з приватних підмереж та відповідні Route Tables з оптимізованою маршрутизацією.

Впроваджено багаторівневу систему безпеки типу «defense-in-depth» через п’ять спеціалізованих Security Groups (Bastion-SG, Web-Servers-SG, Database-SG, ALB-SG, Monitoring-SG), Network ACLs на рівні підмереж, IAM ролі з принципом найменших привілеїв (EC2-WebServer-Role, RDS-Enhanced-Role, EC2-Bastion-Role) та власний KMS ключ з автоматичною щорічною ротацією для шифрування всіх EBS томів, що забезпечує комплексний захист від мережевого до прикладного рівня з мінімізацією поверхні атак. Розроблено та налаштовано RDS DB Instance з MySQL 8.0.35, Multi-AZ конфігурацією для автоматичного failover, зашифрованим 100GB gp3 storage, 30-денним backup retention period та Enhanced Monitoring для детального аналізу продуктивності бази даних. Створено чотири типи спеціалізованих серверів: Bastion Host (t3.micro) з нестандартним SSH портом 12322 та Elastic IP для безпечного адміністративного доступу, State Web Server (t3.medium) з Nginx та додатковим 50GB зашифрованим томом, масштабовані Web Servers (t3.medium) з Nginx, Node Exporter та додатковими 100GB томами, GAP Monitoring Server (t3.large) з повним стеком моніторингу Grafana, Prometheus, Alertmanager та 200GB томом для зберігання метрик. Розроблено інтелектуальні політики автоматичного масштабування на основі метрик CPU (>90 %) та пам’яті (>85 %) з діапазоном від 2 до 8 інстансів, включаючи Launch Template з підготовленим AMI, User Data скриптами та cooldown періодами 300 секунд, що забезпечує оптимальне використання ресурсів, економічну ефективність при змінних навантаженнях та стабільну роботу системи під час масштабування.

Налаштовано Application Load Balancer з internet-facing схемою та диференційованою маршрутизацією через три маршрути: «/» направляє трафік до Web Hosts ASG Target Group для продуктивного середовища, «/test» – до

State Web Host Target Group для тестування, «/monitoring» – до GAP Host Target Group для доступу до Grafana, з відповідними Target Groups та індивідуальними health checks (GET /, GET /test/health, GET /api/health) з налаштованими threshold значеннями та timeout параметрами. Створено інтегровану систему моніторингу на базі CloudWatch, Prometheus та Grafana з п'ятьма критичними алертами (EC2 CPU > 80 %, EC2 Memory > 85 %, Disk > 80 %, ASG Health Check failures, RDS CPU > 90 %), централізованим збором Tomcat логів через CloudWatch Logs Agent з 30-денним retention period, чотирма комплексними дашбордами (Infrastructure Overview, Application Performance, Database Analytics, Load Balancer Metrics) та автоматичними SNS нотифікаціями для забезпечення повної видимості стану системи та швидкого реагування на інциденти.

Впроваджено автоматизовані системи резервного копіювання з диференційованими політиками зберігання: RDS автоматичні backup щодня о 03:00 UTC з 30-денним retention period та point-in-time recovery з 5-хвилинною точністю, EBS snapshots щодня о 02:00 UTC з 7-денним retention period, Lambda функції для автоматичного створення snapshots з відповідними тегами для lifecycle management та cross-region backup для критичних даних в регіон us-west-2 для додаткової надійності та disaster recovery.

Забезпечено безпечний багаторівневий доступ: зовнішній SSH доступ до Bastion Host через порт 12322, внутрішній SSH доступ з Bastion до всіх серверів через порт 22, підключення веб-серверів до RDS через порт 3306, доступ до Grafana через ALB на порт 3000 та HTTP трафік через ALB на порт 80, що створює контрольовану та аудитовану систему доступу з мінімізацією ризиків безпеки. Експериментальні результати підтвердили високу ефективність запропонованої архітектури: покращення середнього часу відгуку системи на 40 % (з 850 мс до 510 мс), підвищення пропускної здатності на 51 % (з 450 RPS до 680 RPS), досягнення 99.7 % доступності порівняно з 99.2 % у традиційних рішеннях, зниження середнього CPU utilization на 23 % (з 85 % до 65 %) завдяки ефективному автоматичному масштабуванню та коефіцієнт ефективності масштабування 0.89, що значно перевищує цільовий показник 0.85 та свідчить про оптимальну роботу системи під різними навантаженнями.

Висновки

У статті розроблено та реалізовано комплексну архітектуру високодоступної хмарної інфраструктури AWS, яка забезпечує інтеграцію всіх необхідних компонентів в єдину масштабовану систему з урахуванням сучасних вимог до надійності, безпеки та продуктивності.

Практична значущість роботи полягає у створенні еталонної, готової до впровадження архітектури, яка може бути використана як основа для розгортання масштабованих веб-додатків з високими вимогами до доступності, безпеки та продуктивності в організаціях різного масштабу – від стартапів до enterprise рівня, а також як навчальний матеріал для підготовки фахівців з хмарних технологій та DevOps практик.

Наукова новизна полягає в систематизації та формалізації комплексного підходу до інтеграції всіх компонентів AWS інфраструктури в єдину екосистему, включаючи розробку методології диференційованої маршрутизації трафіку з урахуванням специфіки різних середовищ, оптимізації політик автоматичного масштабування на основі комбінованих метрик CPU та пам'яті, інтеграції гетерогенних систем моніторингу (CloudWatch, Prometheus, Grafana) з єдиними дашбордами та алертами, а також створення комплексної системи безпеки з багаторівневим захистом та принципом найменших привілеїв. Економічна ефективність досягається через Auto Scaling Groups, що забезпечують зменшення простою ресурсів на 23 %, диференційовані політики backup з оптимізацією витрат на зберігання до 40 %, Infrastructure as Code підхід, що зменшує час розгортання на 60 % та кількість помилок конфігурації на 75 %, використання Reserved Instances для базового навантаження та Spot Instances для додаткових потужностей.

Перспективи подальших досліджень включають розробку методів предиктивного масштабування з використанням машинного навчання для аналізу патернів навантаження та прогнозування піків трафіку, інтеграцію з контейнерними технологіями Amazon EKS та AWS Fargate для підвищення ефективності використання ресурсів, дослідження методів оптимізації витрат через динамічне використання Spot Instances та Savings Plans, розробку алгоритмів автоматичного тюнінгу параметрів бази даних на основі аналізу Performance Insights, впровадження chaos engineering методів для тестування відмовостійкості системи, створення систем автоматичного disaster recovery з cross-region failover, розробку методів zero-downtime deployment та blue-green deployment стратегій, а також дослідження можливостей інтеграції з serverless технологіями AWS Lambda для оптимізації архітектури мікросервісів.

Список використаної літератури

1. Amazon Web Services. AWS Well-Architected Framework. 2023. URL: https://aws.amazon.com/architecture/well-architected/ (дата звернення: 15.11.2024).

2. Bauer, E., Adams, R. Reliability and Availability of Cloud Computing. IEEE Computer Society. 2012. 296 p. https://doi.org/10.1002/9781118393994

3. Armbrust, M., Fox, A., Griffith, R., Joseph, A. D. A view of cloud computing. Communications of the ACM. 2010. Vol. 53(4). P. 50–58. https://doi.org/10.1145/1721654.1721672

4. Mell, P., Grance, T. The NIST Definition of Cloud Computing. NIST Special Publication 800-145. 2011. https://doi.org/10.6028/NIST.SP.800-145

5. Singh, S., Chana, I. A survey on resource scheduling in cloud computing: Issues and challenges. Journal of Grid Computing. 2016. Vol. 14(2). P. 217–264. https://doi.org/10.1007/s10723-015-9359-2

6. Buyya, R., Yeo, C.S., Venugopal, S., Broberg, J., Brandic, I. Cloud computing and emerging IT platforms: Vision, hype, and reality for delivering computing as the 5th utility. Future Generation Computer Systems. 2009. Vol. 25(6). P. 599–616. https://doi.org/10.1016/j.future.2008.12.001

References

1. (2023). Amazon Web Services. AWS Well-Architected Framework. URL: https://aws.amazon.com/architecture/well-architected/ (дата звернення: 15.11.2024).

2. Bauer, E., Adams, R. (2012). Reliability and Availability of Cloud Computing. IEEE Computer Society. 296 p. https://doi.org/10.1002/9781118393994

3. Armbrust, M., Fox, A., Griffith, R., Joseph, A.D. (2010). A view of cloud computing. Communications of the ACM. Vol. 53(4). P. 50–58. https://doi.org/10.1145/1721654.1721672

4. Mell, P., Grance, T. (2011). The NIST Definition of Cloud Computing. NIST Special Publication 800–145. https://doi.org/10.6028/NIST.SP.800-145

5. Singh, S., Chana, I. (2016). A survey on resource scheduling in cloud computing: Issues and challenges. Journal of Grid Computing. Vol. 14(2). P. 217–264. https://doi.org/10.1007/s10723-015-9359-2

6. Buyya, R., Yeo, C.S., Venugopal, S., Broberg, J., Brandic, I. (2009). Cloud computing and emerging IT platforms: Vision, hype, and reality for delivering computing as the 5th utility. Future Generation Computer Systems. Vol. 25(6). P. 599–616. https://doi.org/10.1016/j.future.2008.12.001

Дата першого надходження статті до видання: 11.01.2026
Дата прийняття статті до друку після рецензування: 16.02.2026
Дата публікації (оприлюднення) статті: 30.04.2026