УДК 004.451:[004.75:004.43 DOI: 10.31673/2412-4338.2025.048923

Ніщеменко Дмитро Олександрович
Державний університет інформаційно-комунікаційних технологій, Київ
ORCID 0009-0006-1781-4109

Аронов Андрій Олексійович
Державний університет інформаційно-комунікаційних технологій, Київ
ORCID: 0009-0000-7868-8341

Гавор Артур Станіславович
Державний університет інформаційно-комунікаційних технологій, Київ
ORCID: 0009-0002-9705-1666

Герцюк Микола Модестович
Державний університет інформаційно-комунікаційних технологій, Київ
ORCID: 0000-0003-2946-9673

Гордієнко Катерина Олександрівна
Державний університет інформаційно-комунікаційних технологій, Київ
ORCID: 0009-0002-5997-9682

СУЧАСНІ ОПЕРАЦІЙНІ СИСТЕМИ ЯК ПЛАТФОРМА ДЛЯ ІНТЕГРАЦІЇ БЛОКЧЕЙН ТЕХНОЛОГІЙ, NOSQL-СХОВИЩ І МУЛЬТИПАРАДИГМАЛЬНОГО ПРОГРАМУВАННЯ (JAVA, PYTHON)

Анотація. *У статті представлено результати комплексного дослідження ролі сучасної операційної системи (ОС) як фундаментальної платформи для інтеграції гетерогенних технологічних стеків, що охоплюють блокчейн-мережі, NoSQL-сховища та прикладне програмне забезпечення, реалізоване на базі мультипарадигмальних мов програмування (Java, Python). Автором обґрунтовано, що на відміну від традиційних підходів, орієнтованих на прикладний рівень, стратегічний вибір, конфігурація та низькорівневе налаштування параметрів ядра ОС є визначальними чинниками забезпечення високої продуктивності, стабільності та безпеки складних розподілених систем.*

У межах дослідження систематизовано архітектурні патерни для ефективної взаємодії он-чейн та оф-чейн компонентів, а також розроблено науково-практичні рекомендації щодо оптимізації підсистем ядра Linux, зокрема в частині управління конфліктуючими ресурсами та мережевої взаємодії. Особливу увагу приділено питанням зміцнення безпеки через впровадження механізмів примусового контролю доступу (MAC) та контейнеризації в середовищах оркестрації. Ключовим науковим результатом роботи є запропонована цілісна референтна архітектура, яка інтегрує передові системні практики для проектування надійних і масштабованих IT-рішень.

Ключові слова: *операційна система, блокчейн, NoSQL, ООП, масштабовані системи, Java, Python, продуктивність, контейнеризація.*

Nishchemenko Dmytro
State university of information and communication technologies, Kyiv
ORCID: 0009-0006-1781-4109

Aronov Andriy
State university of information and communication technologies, Kyiv
ORCID: 0009-0000-7868-8341

Gavor Artur
State university of information and communication technologies, Kyiv
ORCID: 0009-0002-9705-1666

Hertsyuk Mykola
State university of information and communication technologies, Kyiv
ORCID: 0000-0003-2946-9673
Gordienko Kateryna

State university of information and communication technologies, Kyiv
ORCID: 0009-0002-5997-9682

MODERN OPERATING SYSTEMS AS A PLATFORM FOR INTEGRATING BLOCKCHAIN TECHNOLOGIES, NOSQL STORAGE AND MULTI-PARADIGM PROGRAMMING (JAVA, PYTHON)

Abstract*. This article presents a comprehensive analysis of the role of modern operating systems (OS) as a fundamental platform for the integration of heterogeneous technologies, including blockchain, NoSQL databases, and multi-paradigm programming languages (Java, Python). It addresses the critical architectural challenge of managing conflicting workloads—the compute-intensive nature of blockchain nodes versus the memory- and I/O-intensive demands of NoSQL databases. This problem is largely overlooked in existing research, which predominantly focuses on the application layer. The paper proves that the selection, configuration, and in-depth tuning of the OS are decisive factors for achieving the required performance, stability, and security in such complex distributed systems.*

A systematic approach is presented, which includes the analysis of architectural patterns, proposing a hybrid model that combines microservices, event-driven architecture (EDA), and peer-to-peer (P2P) principles. The article provides practical, expert-level recommendations for Linux kernel tuning, covering memory management, I/O scheduling (noop for NVMe SSDs), and network stack optimization. Furthermore, a multi-layered security model based on a defense-in-depth strategy is formulated, combining host hardening with Mandatory Access Control (MAC) frameworks (SELinux/AppArmor) and workload isolation via containerization. Deployment strategies for stateful applications using Kubernetes, specifically with StatefulSets, are detailed and supported by real-world case studies.

The main scientific result of this research is a holistic reference architecture that synthesizes these best practices into a unified blueprint for building robust and efficient distributed systems. This architecture is based on the core principle of storing proofs on-chain and detailed state off-chain, leveraging the complementary strengths of each technology. The findings provide system architects and DevOps engineers with a scientifically grounded methodology and a practical framework for designing, optimizing, and operating complex, high-performance systems

.***Keywords:*** *operating system, blockchain, NoSQL, OOP, scalable systems, Java, Python, performance, containerization.*

1. Вступ.

Проектування сучасних інформаційних систем базується на інтеграції гетерогенних технологічних стеків, що зумовлює підвищення вимог до базової інфраструктури. У середовищах, де одночасно функціонують блокчейн-мережі, NoSQL-сховища та прикладне ПЗ на базі Java та Python, операційна система (ОС) стає активним архітектурним компонентом, відповідальним за стабільність інтеграційних процесів. Ключовою науковою проблемою є відсутність системного підходу до управління компонентами з антагоністичними профілями навантаження: обчислювально-інтенсивною природою блокчейну та високими вимогами NoSQL до підсистеми вводу-виводу (I/O).

Актуальність дослідження зумовлена необхідністю розробки цілісної методології проектування та оптимізації хостової ОС як критичного шару безпечної інтеграції розподілених технологій. Вирішення цієї проблеми потребує зміщення фокусу з прикладного аналізу на системний рівень, що передбачає глибоке налаштування ядра та механізмів розподілу ресурсів для забезпечення синергії елементів системи..

2. Аналіз літературних даних і постановка проблеми.

Інтеграція блокчейн-технологій, NoSQL-сховищ та сучасних мов програмування є активною сферою наукових досліджень, у якій можна виділити кілька ключових напрямків. Аналіз сучасних публікацій показує, що дослідження переважно зосереджені на прикладних аспектах: пропонуються гібридні архітектури для підвищення масштабованості блокчейну за допомогою NoSQL, розробляються методи для забезпечення інтероперабельності з корпоративними системами, а також вдосконалюються інженерні практики розробки децентралізованих додатків [1, 2, 3].

Попри значну увагу до архітектурних патернів, стратегій управління даними та інженерних практик, комплексний аналіз наявних публікацій виявляє суттєву науково-технічну прогалину. Фундаментальна роль хостової операційної системи як платформи, що забезпечує виконання, ізоляцію та управління ресурсами для всіх вищезазначених компонентів, послідовно залишається поза увагою. Більшість проаналізованих робіт не розглядають сучасні ОС як об'єкт дослідження, концентруючись на інтеграції технологій, але не на їхній взаємодії в рамках єдиного операційного середовища. У літературі відсутній системний аналіз того, як конфігурація ядра ОС, налаштування планувальників процесів та вводу-виводу, параметри мережевого стеку та застосування механізмів примусового контролю доступу (MAC) впливають на кінцеву продуктивність, стабільність та безпеку інтегрованого гетерогенного стеку.

3. Мета і задачі дослідження.

Виходячи з виявленої проблеми відсутності системного підходу до оптимізації операційної системи як інтеграційної платформи, метою даної статті є комплексний аналіз ролі сучасної ОС та розробка науково обґрунтованих рекомендацій для побудови високопродуктивних та безпечних розподілених систем на основі блокчейну, NoSQL та мультипарадигмального програмування.

Для досягнення поставленої мети необхідно вирішити наступні завдання:

- проаналізувати ключові функції та обов'язки операційної системи в контексті управління конфліктуючими гетерогенними навантаженнями;
- систематизувати архітектурні патерни для ефективного поєднання он-чейн та оф-чейн компонентів на системному рівні;
- розробити практичні рекомендації з оптимізації продуктивності ядра ОС, підсистем вводу-виводу та мережі;
- сформулювати багаторівневу модель безпеки, що охоплює зміцнення хостової ОС та ізоляцію навантажень;
- запропонувати цілісну референтну архітектуру та стратегії її розгортання в сучасних інфраструктурах..

4. Результати дослідження.

Побудова ефективних та надійних систем на основі гетерогенних технологій, таких як блокчейн та NoSQL, вимагає комплексного підходу. Для вирішення поставленого завдання необхідно послідовно розглянути ключові аспекти системної інтеграції: від фундаментальної ролі операційної системи як платформи та вибору архітектурних патернів, до методів глибокої оптимізації продуктивності, побудови багаторівневої моделі безпеки та сучасних стратегій розгортання й експлуатації.

4.1. Операційна система як фундаментальний шар інтеграції.

У межах проектування високонавантажених систем на основі гетерогенних технологій операційна система (ОС) розглядається як активний архітектурний рівень, що забезпечує детерміноване управління ресурсами та гарантування заданих параметрів якості обслуговування (QoS). ОС виконує функції арбітражу фізичних потужностей (CPU, RAM, IOPS, мережева пропускна здатність), що є критично важливим при паралельному функціонуванні вузлів блокчейну та NoSQL-сховищ.

Аналіз конфліктів профілів навантаження Дослідження виявило фундаментальну антагоністичність профілів споживання ресурсів ключовими компонентами стеку. Блокчейн-вузли характеризуються високою інтенсивністю обчислень (compute-intensive) під час валідації транзакцій та значним навантаженням на підсистему вводу-виводу (I/O-bound) при записі блоків. В свою чергу, NoSQL-сховища орієнтовані на максимізацію використання оперативної пам'яті (memory-bound) для кешування та інтенсивну мережеву взаємодію (network-intensive).

Відсутність ізоляції цих компонентів призводить до деградації продуктивності: надмірна I/O-активність блокчейн-вузла збільшує латентність операцій персистентності бази даних, а висока утилізація RAM з боку NoSQL ініціює механізми підкачки (swap) для блокчейн-клієнта.Запропонована системна архітектура має чітку багаторівневу структуру, де операційна система виступає як фундаментальний шар управління ресурсами. У просторі користувача паралельно функціонують гетерогенні компоненти: вузол блокчейну, база даних NoSQL та прикладні сервіси, розроблені з використанням мультипарадигмальних мов програмування, як-от Java та Python (див. рис. 1).

Таким чином, ізоляція ресурсів на рівні ОС має розглядатися як основний архітектурний принцип. Сучасні функції ядра, такі як контрольні групи (cgroups) в Linux та пріоритети планування процесів (nice), є не просто опціями, а необхідними архітектурними інструментами для гарантування того, що один компонент системи не може негативно вплинути на інший.

Фундаментальним архітектурним рішенням є вибір ОС. Порівняльний аналіз за ключовими критеріями вказує на значні переваги дистрибутивів на базі ядра Linux завдяки вищій продуктивності, більш зрілій екосистемі для цільового технологічного стеку, потужним засобам безпеки та нижчій загальній вартості володіння.

Деконструкція поняття «Blockchain Operating System» (BOS) Необхідно чітко диференціювати універсальні ОС та спеціалізовані рішення класу BOS. Останні класифікуються як децентралізоване проміжне програмне забезпечення (middleware), що надає абстракції для розгортання dApps (смарт-

контракти, консенсус, ідентифікація). Хостова ОС (наприклад, Linux) залишається фундаментальним рівнем, що керує апаратними ресурсами для функціонування самого ПЗ блокчейн-вузла.

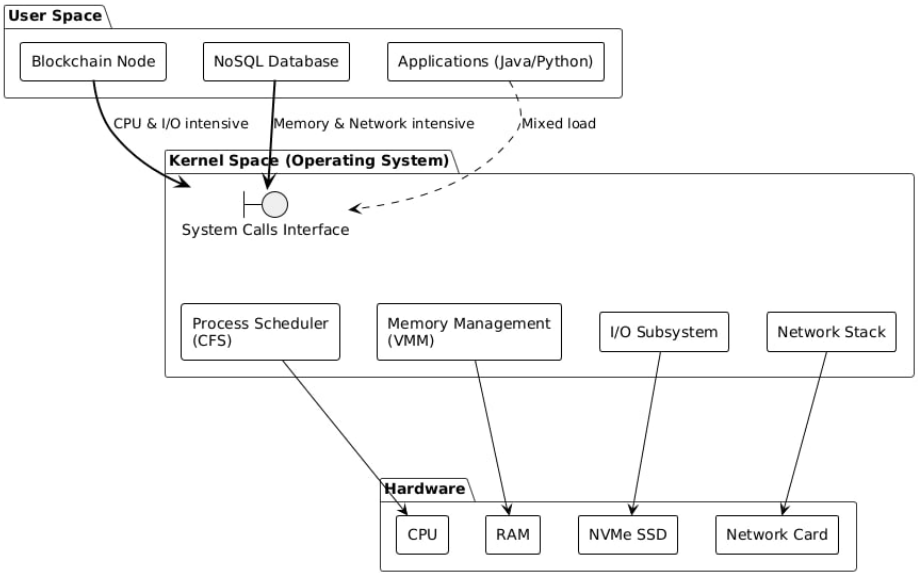

Рис. 1. Архітектурне положення ОС у гетерогенному стеку.

Ієрархічна структура стеку:

1. Апаратне забезпечення.
2. Host OS (Linux) управління фізичними ресурсами та ізоляція.
3. Blockchain Node / BOS рівень логіки розподіленого реєстру.
4. Decentralized Applications (dApps) прикладний рівень.

Таким чином, системна оптимізація хостової ОС є першочерговим завданням для забезпечення стабільності та масштабованості інтегрованих блокчейн-рішень..

4.2. Архітектурні патерни для системної інтеграції.

Ефективна інтеграція різнорідних компонентів вимагає застосування комбінованого підходу до проєктування архітектури, оскільки жоден окремий патерн не є достатнім для таких складних гібридних систем. Оптимальна архітектура є композитною та поєднує сильні сторони кількох моделей для досягнення гнучкості, масштабованості та слабкої зв'язаності компонентів.

Гібридна архітектура програмного забезпечення. Основою для блокчейн-компонента слугує Peer-to-Peer (P2P) патерн, оскільки він є фундаментальним для функціонування децентралізованих мереж. Для розробки оф-чейн компонентів, що реалізують бізнес-логіку на Java та Python, найбільш доцільною є мікросервісна архітектура. Вона дозволяє незалежно розробляти, масштабувати та розгортати різні функціональні модулі, що ідеально узгоджується з подальшою контейнеризацією.

Ключовим елементом, що поєднує он-чейн та оф-чейн світи, є подієво-орієнтована архітектура (Event-Driven Architecture, EDA). Події в блокчейні, наприклад, підтвердження нової транзакції або виконання смарт-контракту, можуть ініціювати асинхронні дії в оф-чейн мікросервісах. Такий підхід створює реактивну та слабкозв'язану систему, де комунікація між компонентами відбувається через подієву магістраль, що дозволяє ефективно керувати затримками та ймовірнісною фінальністю блокчейн-транзакцій.

Взаємодія між децентралізованими та централізованими компонентами системи організована за гібридною архітектурною моделлю, що поєднує мікросервісний та подієво-орієнтований підходи (див. рис. 2). Он-чейн вузол блокчейну генерує події (наприклад, про підтвердження транзакції), які публікуються у подієву магістраль, що виступає як слабкозв'язаний посередник. Оф-чейн мікросервіси, реалізовані на Java або Python, підписуються на ці події, обробляють бізнес-логіку та оновлюють детальний стан системи у сховищі NoSQL.

Архітектура шару даних: синергія блокчейну та NoSQL. Ефективне управління даними базується на принципі розділення: децентралізована фіксація доказів (он-чейн) та зберігання стану в оф-чейн репозиторіях. Блокчейн функціонує як незмінна аудиторська система, що оперує виключно компактними структурами (хеші транзакцій, цифрові підписи), тоді як NoSQL-сховища (MongoDB, Cassandra) забезпечують масштабоване зберігання об’ємних прикладних даних (профілі, телеметрія). Ця симбіотична модель нівелює обмеження блокчейну щодо швидкодії та вартості

зберігання.

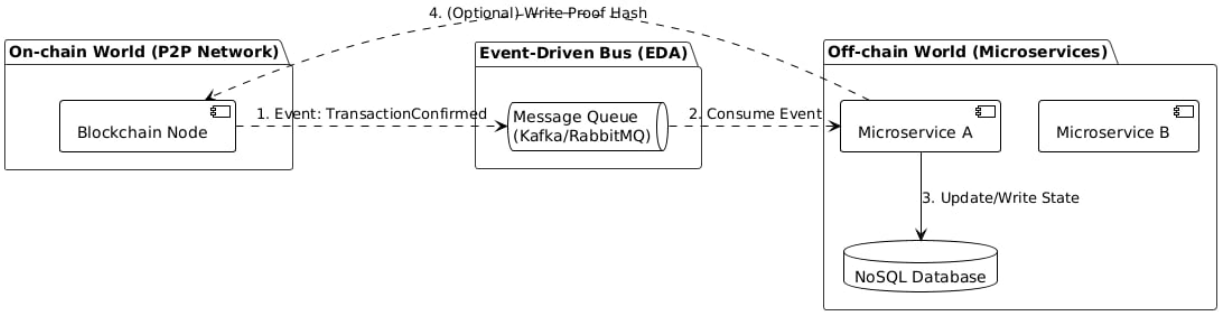

Рис. 2. Гібридна архітектура інтегрованої системи.

Програмна інтеграція компонентів реалізується через стандартизовані інтерфейси (JSON-RPC) та спеціалізовані бібліотеки (Web3j, Web3.py) за моделлю «клієнт-сервер». Вибір мов Java та Python архітектурно обґрунтований: механізми ООП (зокрема інкапсуляція) корелюють із мікросервісним патерном, що спрощує контейнеризацію та ізоляцію на рівні ОС. Використання високорівневих абстракцій уніфікує мережевий трафік і системні виклики, полегшуючи налаштування політик безпеки (MAC) та управління ресурсами операційної системи.

4.3. Управління ресурсами та оптимізація продуктивності.

Досягнення максимальної продуктивності інтегрованого стеку неможливе без глибокої оптимізації операційної системи, налаштування якої мають бути узгоджені зі специфічними профілями навантаження ключових компонентів. Процес вимагає як правильного підбору апаратного забезпечення, так і тонкого налаштування ядра та його підсистем.

Таблиця 1.

Рекомендовані параметри ядра Linux для високопродуктивних навантажень

Параметр	**Рекомендоване значення**	**Обґрунтування та вплив**
vm.swappiness	1	Запобігає свопінгу критичних сторінок пам'яті баз даних, зменшуючи затримку.
transparent_hugepage/enabled	never	Уникає затримок, пов'язаних з дефрагментацією пам'яті, що є проблемою для баз даних.
vm.dirty_background_ratio	5	Починає фоновий запис "брудних" сторінок раніше, згладжуючи піки I/O.
vm.dirty_ratio	10-20	Обмежує загальний обсяг "брудних" сторінок, запобігаючи тривалим блокуванням I/O.
fs.file-max	100000+	Збільшує максимальну кількість відкритих файлів у системі, що необхідно для баз даних та мережевих сервісів.
ulimit -n (nofile)	65536+	Збільшує ліміт відкритих файлів для кожного процесу.

Експлуатація повноцінних блокчейн-вузлів характеризується високою інтенсивністю використання дискової підсистеми та обчислювальних потужностей. Критичною вимогою є застосування накопичувачів NVMe SSD з високим ресурсом перезапису (TBW) для забезпечення необхідного рівня IOPS при роботі з леджером [4]. Навантаження на CPU зумовлене безперервною реалізацією енергоємних криптографічних алгоритмів (зокрема SHA-256 та ECDSA). Встановлено, що для детермінованої послідовної обробки транзакцій вирішальне значення має висока тактова частота окремого ядра, а не загальний рівень паралелізму процесора.

Профіль навантаження цих систем зміщений у бік підсистеми оперативної пам'яті. Для забезпечення низької латентності операцій NoSQL-бази та оф-чейн сервіси активно використовують механізми кешування та внутрішньо-оперативні структури даних (наприклад, хеш-таблиці), що обумовлює необхідність пріоритетної оптимізації управління RAM на рівні ядра ОС для запобігання деградації продуктивності. Крім того, параметри мережевого стеку є визначальними для підтримки високої пропускної здатності під час реплікації та шардингу в межах кластера [5, 6].

4.4. Стратегії розгортання та експлуатації.

Безпека інтегрованого стеку вимагає застосування глибокоешелонованої оборони (defense-in-depth), що охоплює кілька рівнів: від фундаментального зміцнення хостової ОС до ізоляції робочих навантажень та пом'якшення специфічних для технологій загроз.

Для фундаментального зміцнення ОС стандартних дозволів DAC недостатньо, тому рекомендується застосування фреймворків примусового контролю доступу (MAC), як-от SELinux на основі міток, для максимального гранулярного контролю, або AppArmor, на основі шляхів, для простішого впровадження та управління.

Безпека та ізоляція контейнерів. Контейнеризація за допомогою Docker надає хороший рівень ізоляції за замовчуванням, але він не є абсолютним. Основні примітиви безпеки включають: простори імен ядра (ізоляція процесів, мережі), контрольні групи (cgroups для обмеження ресурсів) та відкинуті можливості (capabilities для обмеження привілейованих системних викликів). Найкращі практики безпеки вимагають запускати процеси в контейнерах від імені непривілейованого користувача, використовувати довірені образи та файлові системи "тільки для читання", де це можливо.

Ключовим моментом є те, що ізоляція контейнерів повністю покладається на безпеку реалізацій цих механізмів у ядрі Linux. Уразливість у ядрі може дозволити зловмиснику "втекти" з контейнера. Саме тому MAC надає другий, незалежний шар захисту. Навіть якщо зловмисник зможе обійти ізоляцію просторів імен, політика MAC на хості все одно обмежить його дії, не дозволяючи отримати доступ до критичних файлів чи виконати несанкціоновані операції. Ця комбінація забезпечує надійну глибокоешелоновану оборону.

Системні аспекти безпеки відіграють вирішальну роль у захисті від специфічних для блокчейну загроз. Щоб унеможливити крадіжку приватних ключів, яка є основним ризиком, застосовуються механізми контролю на рівні ОС, такі як суворі права доступу до файлів (DAC, MAC) та ізоляція вузла в зміцненому контейнері. Проти мережевих атак типу DDoS та Sybil ефективним є використання хостових фаєрволів для обмеження швидкості та фільтрації трафіку на рівні операційної системи. Водночас ризики атак на ланцюг постачання знижуються завдяки запуску вузла в контейнері з мінімальним образом, що зменшує поверхню атаки, та застосуванню сканерів і політик безпеки для контролю над сторонніми компонентами.

4.5. Багаторівнева модель безпеки.

Захист гетерогенного стеку базується на принципі глибокоешелонованої оборони (defense-in-depth), що передбачає перехід від дискреційного (DAC) до примусового контролю доступу (MAC). Використання SELinux або AppArmor забезпечує гранулярний контроль суб’єктів та об’єктів, мінімізуючи ризики компрометації хостової ОС.

Ізоляція прикладних компонентів реалізується через механізми контейнеризації (namespaces, cgroups, capabilities). Безпека середовища посилюється запуском процесів від непривілейованих користувачів, використанням верифікованих образів та read-only файлових систем. Оскільки контейнерна ізоляція критично залежить від цілісності ядра, механізми MAC створюють незалежний шар захисту, що блокує несанкціонований доступ навіть у разі «втечі» з контейнера (container escape).

Ефективність контейнерної ізоляції безпосередньо залежить від цілісності ядра Linux, оскільки вразливості на рівні ядра можуть призвести до компрометації хоста через вихід за межі контейнера (container escape). У цьому контексті механізми MAC формують незалежний шар захисту: навіть за умови обходу ізоляції просторів імен, політики примусового контролю на рівні хоста блокують несанкціонований доступ до критичних системних ресурсів. Така комбінація технологій забезпечує синергетичний ефект у створенні відмовостійкого середовища.

Системні методи захисту відіграють визначальну роль у нівелюванні специфічних ризиків блокчейн-мереж. Захист конфіденційності приватних ключів гарантується через сувору ізоляцію вузла в зміцненому контейнері та застосування багаторівневих прав доступу до файлових систем. Для протидії мережевим загрозам, таким як DDoS-атаки та атаки Сивілли (Sybil attacks), застосовуються хостові фаєрволи для фільтрації трафіку та обмеження інтенсивності запитів на рівні ОС. Ризики, пов'язані з атаками на ланцюг постачання (supply chain attacks), мінімізуються шляхом використання дистрибутивів з мінімальним набором системних компонентів та безперервного сканування образів на предмет вразливостей сторонніх бібліотек. Таким чином, комплексна конфігурація операційної системи є базовим інструментом гарантування цілісності та доступності

розподілених блокчейн-рішень.

5. Висновки

Проведене дослідження підтверджує, що в архітектурі сучасних розподілених систем, які базуються на інтеграції блокчейну, NoSQL-сховищ та мультипарадигмальних додатків, операційна система (ОС) трансформується у функціонально активний компонент. Ефективність управління антагоністичними профілями навантаження, загальна продуктивність та безпека технологічного стеку безпосередньо корелюють із якістю низькорівневої конфігурації хостової ОС. На відміну від підходів, орієнтованих виключно на прикладний рівень, у роботі обґрунтовано парадигму системного проектування, де ОС розглядається як фундаментальний шар для забезпечення цілісності та відмовостійкості комплексних рішень.

Запропонована референтна архітектура синтезує передові практики системного адміністрування та розробки програмного забезпечення. Вона передбачає використання дистрибутивів на базі ядра Linux з автоматизованими профілями оптимізації та реалізацію концепції глибокоешелонованої оборони через механізми примусового контролю доступу (MAC) та контейнеризацію процесів з жорсткою лімітацією ресурсів. Оркестрація компонентів у середовищі Kubernetes із застосуванням контролерів StatefulSets та персистентних томів у режимі ReadWriteOnce гарантує стабільність stateful-додатків. Інтеграційна логіка базується на гібридній моделі, що поєднує мікросервісний та подієво-орієнтований підходи для забезпечення слабкої зв’язності он-чейн та оф-чейн сегментів. При цьому дотримання принципу розділення даних — фіксація доказів у блокчейні та збереження повного стану в NoSQL-сховищах — дозволяє досягти оптимального балансу між незмінністю та масштабованістю.

Перспективи подальших розвідок у даному напрямку пов'язані з еволюцією самої операційної системи та децентралізованих технологій. Нові технології, такі як eBPF, відкривають можливості для безпечного, програмованого та високоефективного моніторингу, мережевого управління та забезпечення безпеки безпосередньо всередині ядра ОС, що може кардинально змінити підходи до управління складними навантаженнями. Тенденції, як-от моделі спільної безпеки, наприклад EigenLayer, де економічна безпека одного блокчейну може бути розширена для захисту інших сервісів, підвищують цінність та критичність запущених вузлів.

Список використаної літератури

1. Engineering Blockchain-based Software Systems: Foundations, Survey, and Future Directions / [M. Fahmideh] // ACM Computing Surveys. 2022. Vol. 55, Issue 2. P. 1–38.
2. Development of an Hybrid Blockchain and NoSQL Platform to Improve Data Management [Electronic resource] / [F. Carrozzino, M. Fiore, M. Mongiello] // arXiv.org. – 2023. – arXiv:2305.03592v1. – Access mode: https://arxiv.org/abs/2305.03592 (date of access: 03.10.2025). – Title from the screen.
3. Blockchain-Integrated Databases: A Framework for Immutable and Secure Data Management / S. Saha // The American Journal of Engineering and Technology. – 2025. – Vol. 7, Issue 7. – P. 102–115.
4. Ethereum Node Hardware Requirements (2025 Edition) [Electronic resource] / Cherry Servers. – 2025. – Access mode: https://www.cherryservers.com/blog/ethereum-node-requirements (date of access: 03.10.2025). – Title from the screen.
5. Linux System Performance Tuning: Optimizing CPU, Memory, and Disk [Electronic resource] / Linux Journal. – 2022. – Access mode: https://www.linuxjournal.com/content/linux-system-performance-tuning-optimizing-cpu-memory-and-disk (date of access: 03.10.2025). – Title from the screen.
6. Performance Tuning on Linux: Optimizing Bi Tools and Databases for Maximum Performance / [T. Hussain, P. Prasad] // ResearchGate. – 2019.
7. Tuning Linux for MongoDB: Automated Tuning on Redhat and CentOS [Electronic resource] / Percona Blog. – 2021. – Access mode: https://www.percona.com/blog/tuning-linux-for-mongodb-automated-tuning-redhat-and-centos/ (date of access: 03.10.2025).
8. AppArmor vs SELinux: Compare the Differences in Linux Security [Electronic resource] / TuxCare. – 2023. – Access mode: https://tuxcare.com/blog/selinux-vs-apparmor/ (date of access: 03.10.2025). – Title from the screen.
9. Operating staking nodes on Kubernetes [Electronic resource] / Coinbase Blog. – 2023. – Access mode: https://www.coinbase.com/blog/operating-staking-nodes-on-kubernetes (date of access: 03.10.2025). – Title from the screen.
10. Interoperability: Blockchain's Solution to SOA & Traditional Service-Based Integration Challenges / [Z. A. Shaikh, K. Dahri, A. A. Memon] // 2024 International Conference on Human-Machine Interaction and Big Data (KHI-HTC). – IEEE, 2024. – P. 1–6.