

DOI: <https://doi.org/10.15276/ict.03.2026.79>

УДК 004.056

Модель метаданих оркестрації для виявлення атак у контейнеризованих середовищах Kubernetes

Сіренко Олександр Іванович

ORCID: <https://orcid.org/0000-0001-9751-2544>; olexandr.sirenko@gmail.com

Державний університет інтелектуальних технологій і зв'язку, вул. Кузнечна, 1. Одеса, 65023, Україна

АНОТАЦІЯ

Мета статті полягає у формалізації моделі події контейнеризованого застосунку, розширеної метаданими оркестрації Kubernetes, та побудові матриці відповідності цієї моделі десяти категоріям ризиків Open Web Application Security Project Kubernetes Top 10 (редакція 2025 року), побудованої за формалізованим протоколом кодування. Актуальність дослідження зумовлена тим, що традиційні системи виявлення вторгнень, побудовані на аналізі лише потоку системних викликів, не враховують оркестраційний контекст Kubernetes - приналежність до простору імен, роль у моделі керування доступом на основі ролей, профіль безпеки поду, мережеву політику, фазу життєвого циклу та зв'язок із хмарною ідентичністю, - унаслідок чого однакові системні виклики отримують різну оцінку ризику залежно від контексту. Наукова новизна полягає у типізованому формальному визначенні структури події, яка складається з часової мітки, запису системного виклику, стабільного ідентифікатора сутності та кортежу метаданих оркестрації, з двоетапною функцією зв'язування та явною семантикою відсутності або застарілих значень метаданих, а також у матриці відповідності шести типів оркестраційних метаданих десяти категоріям ризиків за трирівневою шкалою необхідності (необхідно, корисно, не потрібно), побудованій за явним і задокументованим протоколом кодування. Предметом дослідження є взаємозв'язок між типами оркестраційних метаданих і контекстною спостережуваністю категорій ризиків контейнеризованих середовищ на рівні події процесу. Методи дослідження включають формалізацію, структурний аналіз офіційних описів категорій ризиків та жадібний алгоритм визначення пріоритетності впровадження типів метаданих із повним детермінованим правилом розв'язання нічиїх. Отримані результати показують, що профіль безпеки та монтування ідентичності поду є найбільш універсальним типом метаданих, необхідним для трьох категорій ризику, тоді як три категорії - надмірно відкриті компоненти платформи, помилково налаштовані та вразливі компоненти кластера й недостатнє журналювання та моніторинг - структурно не покриваються жодним із розглянутих типів метаданих на рівні події процесу, що окреслює межу застосовності моделі. Жадібний алгоритм визначив порядок впровадження метаданих, що забезпечує повне необхідне метадане покриття семи контекстно спостережуваних категорій ризику в межах визначеної моделі вже після чотирьох кроків. Дослідження має аналітично-формалізований, а не емпіричний характер, що окреслено як його обмеження та напрям подальшої роботи.

Ключові слова: kubernetes; виявлення вторгнень; системні виклики; метадані оркестрації; контекстна спостережуваність; керування доступом на основі ролей; мережеві політики

Для цитування: Сіренко О.І. «Модель метаданих оркестрації для виявлення атак у контейнеризованих середовищах Kubernetes». *Інформатика. Культура. Техніка.* 2026; Том 3 № 2 (4): 1025–1032. DOI: <https://doi.org/10.15276/ict.03.2026.79>

Актуальність. Контейнеризація та Kubernetes стали фактичним стандартом розгортання розподілених застосунків у хмарних та гібридних інфраструктурах, що породжує специфічний клас ризиків безпеки на перетині рівня операційної системи, контейнерного середовища виконання й оркестратора.

Системи виявлення вторгнень для контейнеризованих середовищ традиційно аналізують потік системних викликів через eBPF-інструменти на кшталт Falco. Проте один і той самий системний виклик (наприклад, `execve` чи звернення до токена `ServiceAccount`) може мати різну семантику залежно від оркестраційного контексту: виклик у поді з обмеженим профілем безпеки є значно менш підозрілим, ніж той самий виклик у поді, чий `ServiceAccount` має широкі права в кластері. Промислові рішення (плагін `k8smeta` для Falco, компонент `k8s-metacollector`) частково враховують це, збагачуючи подію метаданими Kubernetes, проте залишаються інженерними рішеннями без формалізованої моделі, яка б системно визначала, які типи метаданих є необхідними для контекстної спостережуваності кожної конкретної категорії ризику на рівні події процесу.

Каталоги ризиків (OWASP Kubernetes Top 10, MITRE ATT&CK for Containers) є описовими: вони фіксують, що є ризиком, але не визначають, якими даними спостережуваності потрібно володіти для контекстної атрибуції події до механізму відповідної категорії ризику. Це породжує дослідницьку проблему: у наявній літературі відсутня формалізована модель, яка б системно зіставляла типи оркестраційних метаданих із

категоріями ризиків на основі явних, відтворюваних критеріїв необхідності.

Мета дослідження полягає у формалізації моделі події контейнеризованого застосунку, розширеної метаданими оркестрації Kubernetes, та побудові матриці відповідності цієї моделі категоріям ризиків OWASP Kubernetes Top 10 на основі відтворюваних критеріїв необхідності.

Дослідження структуроване навколо трьох дослідницьких питань:

1) Яка формальна структура події дозволяє поєднати дані системних викликів із метаданими оркестрації без втрати семантики?

2) Які типи метаданих оркестрації є необхідними, корисними чи надлишковими для контекстної спостережуваності кожної категорії ризику OWASP Kubernetes Top 10 на рівні події процесу?

3) Який пріоритетний порядок впровадження типів метаданих найшвидше забезпечує повне покриття контекстно спостережуваних категорій ризику за визначеним критерієм?

Науковим результатом статті є: формальне визначення структури події зі стабільним ідентифікатором сутності та явною семантикою відсутніх чи застарілих значень метаданих; матриця відповідності цієї моделі десяти категоріям ризиків OWASP Kubernetes Top 10 за тривірневою шкалою необхідності, побудована за формалізованим і задокументованим протоколом кодування; жадібний алгоритм пріоритизації впровадження типів метаданих із повним детермінованим правилом розв'язання нічиїх.

Огляд суміжних робіт. eBPF дозволяє безпечно виконувати верифіковані програми в ядрі Linux без завантаження модулів ядра, що зробило його основою сучасних інструментів спостережуваності - Falco, Tracex, Tetragon [6], [7]. Falco може отримувати події системних викликів, зокрема через eBPF-зонд, і зіставляти їх із декларативними правилами [7]; плагін k8smeta та k8s-metacollector додатково збагачують подію іменами подів, просторів імен і розгортань [8]. Академічна робота Aquila [10] пропонує дворівневу архітектуру захоплення телеметрії (захоплення в ядрі + збагачення метаданими Kubernetes у користувацькому просторі), демонструючи вищу пропускну здатність порівняно з Tracex/Tetragon - емпіричне підтвердження здійсненості архітектури, покладеної в основу запропонованої структури події.

Дві роботи перетворюють послідовності системних викликів на граф-структуру (CHIDS [14]) і забезпечують безпечний аудит на рівні контейнера через LSM-гачки (saBPF [15]), проте жодна не оперує оркестраційними метаданими Kubernetes як окремими класифікованими вимірами. PodCA [9] та робота на основі машинного навчання [11] розв'язують вузькі задачі мережевого виявлення, підтверджуючи, що різні категорії ризику вимагають різних типів контекстних метаданих. Дослідження цифрової криміналістики [12] підтверджує цінність метаданих подів і вузлів для розслідування інцидентів, що додатково мотивує потребу заздалегідь формалізованої моделі.

OWASP Kubernetes Top 10 (2025) визначає десять категорій ризиків кластера [1]; MITRE ATT&CK for Containers систематизує тактики й техніки супротивника, специфічні для контейнерів [2]. Керування доступом на основі ролей у Kubernetes регулює доступ через ролі та зв'язування з адитивними дозволами [3]; мережеві політики визначають ізоляцію вхідного та вихідного трафіку для подів [4]; стандарти безпеки подів визначають профілі привілейованого, базового та обмеженого рівнів [5]. Індустріальні джерела вже намагалися зіставити категорії OWASP із технічними контролями [13], проте мають описовий характер без явних критеріїв і формалізованої процедури класифікації.

Огляд дозволяє виокремити вузьку прогалину: наявні інженерні рішення збагачення подій не формалізують мінімально достатній набір типів метаданих для конкретної категорії ризику; каталоги ризиків є описовими таксономіями без формалізованого зв'язку з телеметрією; академічні моделі контекстуалізації системних викликів не оперують оркестраційними метаданими Kubernetes як окремими класифікованими вимірами. Саме відсутність матриці відповідності «тип метаданих – категорія ризику», побудованої за явним і задокументованим протоколом кодування, визначає предмет цього дослідження.

Методологія. Подія контейнеризованого застосунку визначається як четвірка

$$E = (t, syscall, eid, meta),$$

де t – часова мітка захоплення системного виклику за монотонним лічильником вузла; $syscall$ – типізований запис системного виклику (номер, назва, ідентифікатори процесу й потоку, ідентифікатори просторів імен, аргументи, код повернення); eid – стабільний ідентифікатор сутності, отриманий функцією зв'язування; $meta$ – структурований, типізований і опційний кортеж із шести типів метаданих оркестрації.

Зв'язування здійснюється не за ефемерними характеристиками процесу, а за незмінним ідентифікатором $eid = Pod.metadata.uid$, який Kubernetes ніколи не використовує повторно [16]. Функція зв'язування двоетапна: перший крок – ядро системи локально й миттєво визначає ідентифікатор контейнера за шляхом контрольної групи процесу; другий – кешована функція, що зіставляє цей ідентифікатор з eid через Watch-події Kubernetes API, за зразком архітектурного розділення «ядро → користувацький простір», емпірично підтвердженого в Aquila [10] та реалізованого в парі $k8s-metacollector/k8smeta$ [8]. Оскільки ключем кешу є ідентифікатор контейнера, а не вузол-локальний ідентифікатор простору імен, повторне використання операційною системою числових ідентифікаторів не створює хибного зв'язування.

Кожен компонент кортежу метаданих типізований із явною семантикою відсутнього значення (якщо кеш ще не містить знімка для даного eid) і застарілого значення (якщо знімок застарів за конфігурованим порогом свіжості або відповідний об'єкт позначено на видалення). Ця типізація зберігає зворотну сумісність із $syscall$ -орієнтованими правилами виявлення незалежно від наявності оркестраційного контексту.

На основі аналізу суміжних робіт та офіційної документації Kubernetes [3], [4], [5] визначено шість типів метаданих оркестрації: простір імен – приналежність поду до Namespace, мітки Pod Security Admission, селектори просторів імен [5]; контекст керування доступом на основі ролей – ідентичність ServiceAccount, пов'язані ролі, ефективні дозволені дії [3]; профіль безпеки та монтування ідентичності поду – поля securityContext, профіль Pod Security Standards, прапорець автоматичного монтування токена [5], [1]; мережева політика - застосовні об'єкти NetworkPolicy, стан ізоляції вхідного/вихідного трафіку, підтримка мережевим плагіном [4]; життєвий цикл та розміщення робочого навантаження – фаза поду, часова мітка видалення, тип контролера-власника, вузол розміщення [8]; зв'язок робочого навантаження з хмарною ідентичністю – анотація ServiceAccount, що встановлює федерацію ідентичності робочого навантаження з роллю хмарного провайдера.

Слід явно окреслити межі цього переліку: шість визначених типів метаданих не претендують на вичерпний базис усіх типів метаданих безпеки Kubernetes, а становлять свідомо обмежений набір оркестраційних метаданих рівня робочого навантаження, дібраний за трьома критеріями: відповідність окремому, самостійно конфігурованому об'єкту чи атрибуту Kubernetes API; прив'язаність до конкретного поду, а не до інфраструктурного компонента кластера; можливість спостереження на рівні події процесу без додаткового звернення до зовнішніх реєстрів. Розширення цього переліку додатковими типами метаданих є напрямом подальших досліджень і може вплинути на конкретні значення матриці та на результат жадібного алгоритму.

Для кожної пари «тип метаданих – категорія ризику» визначається одне з трьох значень на основі офіційного опису категорії ризику [1]: необхідно – об'єкт чи атрибут прямо названо в описі суті ризику або в переліку його причин, або він є безпосереднім Kubernetes API-об'єктом, через який технічно реалізується прямо названий механізм; корисно – критерій «необхідно» не задоволено, але тип метаданих покращує атрибуцію події чи оцінку масштабу впливу; не потрібно – обґрунтованого зв'язку не виявлено.

Модель, орієнтована на подію процесу, не претендує на встановлення факту існування чи повноти ризику в кластері в цілому – це питання конфігурації, яке не спостерігається на рівні окремої події. Вона претендує лише на можливість атрибутувати конкретну подію

системного виклику до механізму відповідної категорії ризику. Категорія ризику вважається контекстно спостережуваною на рівні події процесу відносно типу метаданих, якщо цей тип класифіковано як «необхідно» і його значення в момент події не є невідомим чи застарілим. Це розмежування послідовно застосовується по всій статті: там, де вживається «спостережуваність» чи «покриття» стосовно категорій ризику, мається на увазі саме контекстна спостережуваність на рівні події процесу, а не виявлення факту існування ризику в кластері.

Для ілюстрації методики наведено чотири показові рішення з Таблиці 1. Повний журнал обґрунтування для всіх шістдесяти пар «тип метаданих – категорія ризику» ведеться дослідниками як робоча документація проєкту за тим самим протоколом і може надаватися рецензентам чи читачам за запитом.

Категорія K02 «Надмірно дозвоільні конфігурації авторизації» формулює суть ризику саме як надлишкові права ролей чи прив'язки, що дозволяють діям, які виходять за межі потреб робочого навантаження [1] – це прямо визначається типом метаданих контексту керування доступом на основі ролей.

Категорія K04 «Відсутність забезпечення дотримання політик на рівні кластера» визначається як відсутність механізмів централізованого admission control [1]; оскільки профіль Pod Security Admission технічно реалізується через мітку рівня простору імен, саме цей тип метаданих фіксує об'єкт, через який централізоване забезпечення стає технічно можливим.

Категорія K05 «Відсутність контролю мережевої сегментації» прямо визначається відсутністю застосовних об'єктів NetworkPolicy [1] – стандартного об'єкта Kubernetes API, що визначає стан мережевої ізоляції.

Категорія K09 «Ненадійні механізми автентифікації» охоплює облікові дані, що можуть бути використані для автентифікації без належного контролю [1]; змонтований за замовчуванням токен ServiceAccount є дійсними обліковими даними, автоматична неконтрольована наявність яких у поді є прямим механізмом цього ризику.

На основі множини покриття для кожного типу метаданих визначається повне впорядкування всіх шести типів, що на кожному кроці максимізує приріст щойно покритих категорій ризику. Нічия розв'язується за допоміжним критерієм додаткового покриття категорій, позначених «корисно», а за збереження нічий – за зростанням канонічного індексу типу метаданих (третинне, повністю детерміноване правило). Задача споріднена з максимальним покриттям множин, але не є ним у строгому сенсі: метою є повне впорядкування всіх шести елементів, а не мінімальна підмножина.

Алгоритм максимізує приріст покриття на кожному кроці, неявно припускаючи однакову вартість впровадження для всіх шести типів метаданих. Це припущення варто зробити явним: отримання одних типів метаданих (наприклад, приналежності до простору імен) є технічно тривіальним, тоді як обчислення ефективних дозволів керування доступом чи встановлення зв'язку з хмарною ідентичністю може вимагати суттєво складнішої інтеграції. Тому запропоноване впорядкування є базовою пріоритизацією за умови рівної вартості впровадження кожного типу метаданих, а не пріоритизацією, що враховує реальну інженерну складність реалізації; введення зваженого критерію потребує окремої, емпірично обґрунтованої оцінки вартості й виходить за межі цього дослідження.

Результати. Застосування описаної методики до шести типів метаданих і десяти категорій ризиків OWASP Kubernetes Top 10 [1] дало матрицю, наведену в Таблиці 1.

Матриця має неоднорідну структуру: жоден тип метаданих не є необхідним для всіх категорій ризику.

Матриця містить три варіанти значень: Н - необхідно, У - корисно, «-» - не потрібно. Профіль безпеки та монтування ідентичності поду є найбільш універсальним - необхідний одразу для трьох категорій; контекст керування доступом на основі ролей необхідний для двох; решта типів – для однієї категорії кожен. Три категорії – «Надмірно відкриті компоненти платформи», «Помилково налаштовані та вразливі компоненти кластера» й

«Недостатнє журналювання та моніторинг» – не мають жодної позначки «необхідно»: перші дві визначаються властивостями інфраструктурних компонентів кластера, а не оркестраційним контекстом окремого поду, тоді як остання визначається конфігурацією кластерного конвеєра журналювання. Це окреслює структурну межу застосовності моделі, орієнтованої на подію процесу, а не її похибку.

Таблиця 1. Матриця відповідності типів метаданих оркестрації категоріям ризиків OWASP Kubernetes Top 10 (2025)

Тип метаданих	K01	K02	K03	K04	K05	K06	K07	K08	K09	K10
M1 - Простір імен	У	У	У	Н	У	–	–	У	У	У
M2 - RBAC-контекст	–	Н	Н	У	–	У	–	У	У	У
M3 - Профіль безпеки / ід-ті	Н	У	У	Н	У	–	–	У	Н	У
M4 - Мережева політика	–	–	–	У	Н	У	–	У	–	–
M5 - Життєвий цикл/розм.	У	–	–	–	–	–	У	–	–	У
M6 - Хмарна ідентичність	У	У	У	–	–	–	–	Н	У	У

Джерело: розроблено автором

Таблиця 2. Покроковий результат жадібного алгоритму пріоритизації впровадження метаданих

Крок	Обраний тип	Новопокриті категорії	Приріст	Сукупне покриття
1	M3 (профіль безпеки/ід-ті)	K01, K04, K09	3	3 / 7
2	M2 (RBAC-контекст)	K02, K03	2	5 / 7
3	M4 (мережева політика)	K05	1	6 / 7
4	M6 (хмарна ідентичність)	K08	1	7 / 7
5	M5 (життєвий цикл/розм.)	- (лише «Корисно»)	0	7 / 7
6	M1 (простір імен)	- (лише «Корисно»)	0	7 / 7

Джерело: розроблено автором

Алгоритм застосовано над сімома категоріями, що мають хоча б одну позначку «необхідно». Профіль безпеки та монтування ідентичності поду обрано першим (приріст трьох категорій), контекст керування доступом на основі ролей – другим (приріст двох категорій); повне необхідне метадане покриття семи контекстно спостережуваних категорій у межах визначеної моделі досягається вже після чотирьох кроків, тоді як два останні типи метаданих не додають нового необхідного покриття, але залишаються корисними для ширшого кола категорій (Рис. 1).



Рис. 1. Накопичення покриття категорій ризику за кроками жадібного алгоритму пріоритизації

Джерело: розроблено автором

Обговорення. Запропонована структура події відповідає на перше дослідницьке питання завдяки типізованому й опційному характеру метаданих, збереженню повної структури системного виклику незалежно від наявності контексту та архітектурному розділенню захоплення в ядрі від асинхронного збагачення в користувацькому просторі - принцип, емпірично підтверджений незалежно одна від одної роботами Aquila [10] та Falco/k8smeta [8].

Відповідь на друге дослідницьке питання показує, що типи метаданих оркестрації не є взаємозамінними: централізуюча роль профілю безпеки поду узгоджується з тим, що охоплені ним характеристики (політика безпеки, профіль стандартів безпеки подів) пов'язані і з посиленням захисту, і з централізованим забезпеченням дотримання політик, тоді як параметри монтування токена ServiceAccount, які цей тип метаданих фіксує окремо, – з наявністю облікових даних, придатних для автентифікації [1]. Водночас відсутність необхідного покриття для трьох категорій демонструє чітку межу застосовності моделі на двох рівнях – інфраструктурних компонентів кластера та конфігурації конвеєра журналювання, – і модель не повинна позиціонуватися як універсальний заміник інших рівнів спостережуваності.

Отримане впорядкування типів метаданих (відповідь на третє дослідницьке питання) має практичну інтерпретацію: пріоритетне впровадження профілю безпеки та контексту керування доступом на основі ролей дає найбільший приріст покриття на одиницю впровадженого типу метаданих, узгоджуючись із загальною рекомендацією індустріальних джерел зосереджуватися спершу на конфігурації робочих навантажень і контролі авторизації [1], [13], проте, на відміну від них, отримане тут впорядкування виведено формально.

Порівняно з описовими зіставленнями наявної індустріальної літератури [13] та академічними моделями контекстуалізації системних викликів (CHIDS [14], saBPF [15]), запропонована матриця оперує явно визначеними типами метаданих, використовує трирівневу шкалу відповідності та супроводжується задокументованим протоколом кодування. Дослідження має суттєві обмеження.

По-перше, класифікація «корисно»/«не потрібно» спирається на судження дослідників, а не на статистично підтверджену міжекспертну згоду; емпіричної оцінки не проведено; редакція OWASP Kubernetes Top 10 2025 р. продовжує приймати зворотний зв'язок і може зазнавати уточнень [1].

По-друге, перелік типів метаданих є свідомо обмеженим, а не вичерпним базисом метаданих безпеки Kubernetes: розширення додатковими типами метаданих потенційно змінить конкретні значення матриці та результат жадібного алгоритму.

По-третє, жадібний алгоритм ранжує типи метаданих за припущення рівної вартості їх впровадження; оцінка реальної інженерної вартості кожного типу є окремою емпіричною задачею. Природний напрям подальшої роботи – емпірична валідація запропонованої структури на реальному eBPF-конвеєрі, розширення моделі метаданими інфраструктурного рівня та оцінка вартості впровадження для зваженої пріоритизації.

Висновки. У статті сформульовано й розв'язано дослідницьку проблему відсутності формалізованої моделі, яка б системно зіставляла типи оркестраційних метаданих Kubernetes із категоріями ризиків контейнеризованих середовищ.

Щодо першого дослідницького питання запропоновано типізовану формальну структуру події зі стабільним ідентифікатором сутності, двоетапною функцією зв'язування та явною семантикою відсутніх чи застарілих значень метаданих; архітектурна здійсненність структури обґрунтована незалежними технічними прецедентами.

Щодо другого дослідницького питання визначено шість типів метаданих оркестрації та за формалізованим і задокументованим протоколом кодування побудовано матрицю їх відповідності десяти категоріям ризиків OWASP Kubernetes Top 10 за трирівневою шкалою необхідності. Результат показав неоднорідну структуру контекстної спостережуваності:

профіль безпеки та монтування ідентичності поду необхідний для трьох категорій, контекст керування доступом на основі ролей – для двох, тоді як три категорії не покриваються жодним типом метаданих як необхідним, що окреслює межу застосовності моделі.

Щодо третього дослідницького питання розроблено жадібний алгоритм пріоритизації з повним детермінованим правилом розв'язання нічиїх, який визначив порядок впровадження типів метаданих, що забезпечує повне необхідне метадане покриття контекстно спостережуваних категорій ризику вже після чотирьох кроків.

Отримані результати можуть слугувати формалізованою основою для проєктування конвеєрів телеметрії безпеки Kubernetes, які додають оркестраційний контекст цілеспрямовано, а не за принципом збору максимально можливого обсягу метаданих без пріоритизації.

Конфлікт інтересів: Автор декларує, що не має конфлікту інтересів, зокрема фінансового, особистого, авторського чи будь-якого іншого характеру, який міг би вплинути на дослідження, а також на результати, опубліковані в цій статті

Фінансування: Дослідження проводилося без фінансової підтримки

Доступність даних: Усі дані доступні в цифровій або графічній формі в основному тексті рукопису

Використання засобів штучного інтелекту: Під час підготовки рукопису використовувався інструмент (Claude, Anthropic, Sonnet 5) штучного інтелекту (ШІ) для граматичної та стилістичної перевірки тексту. Усі фрагменти, опрацьовані за його допомогою, були перевірені й відредаговані автором. ШІ не використовувався для генерації тексту, формул і таблиць, формування наукових положень, отримання результатів або формулювання висновків. Наукові положення, результати та висновки є власним інтелектуальним внеском автора

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. “OWASP Kubernetes Top 10”. *OWASP Foundation*. 2025. – Available from: <https://kubernetes-top10.owasp.org/>. – [Accessed: March 17, 2026].
2. “ATT&CK for Containers Matrix”. *MITRE Corporation*. 2026. – Available from: <https://attack.mitre.org/matrices/enterprise/containers>. – [Accessed: Aug. 17, 2026].
3. “Using RBAC Authorization”. *Kubernetes documentation*. – Available from: <https://kubernetes.io/docs/reference/access-authn-authz/rbac>. – [Accessed: March 17, 2026].
4. “Network Policies”. *Kubernetes documentation*. – Available from: <https://kubernetes.io/docs/concepts/services-networking/network-policies>. – [Accessed: March 17, 2026].
5. The Kubernetes Authors. “Pod Security Standards”. *Kubernetes documentation*. – Available from: <https://kubernetes.io/docs/concepts/security/pod-security-standards>. – [Accessed: March 17, 2026].
6. “What is eBPF?” – Available from: <https://ebpf.io>. – [Accessed: March 17, 2026].
7. “What is Falco? Open source runtime threat detection”. *The Falco Project*. – Available from: <https://falco.org>. [Accessed: March 17, 2026].
8. Lacuku, A. “How to Deploy Falco with k8s-metacollector + k8smeta Plugin”. *Falco Blog*. 2024. – Available from: <https://falco.org/blog/falco-k8smeta-plugin>. – [Accessed: March 17, 2025].
9. Hussain, A., Aziz, A., Syed, H. J. & Raza, S. “Preventing IP spoofing in Kubernetes using eBPF”. *Computers, Materials & Continua*. 2025; 84 (2): 3105–3124. DOI: <https://doi.org/10.32604/cmc.2025.062628>.
10. Shin, J., Kim, J. & Nam, J. “Aquila: Efficient in-kernel system call telemetry for cloud-native environments”. *Sensors*. 2025; 25 (21): 6511. DOI: <https://doi.org/10.3390/s25216511>.
11. Sivaraman, H. “ML-Based threat detection for container network security in Kubernetes”. *International Journal of Innovative Research and Creative Technology*. 2024; 10 (1): 1–8. DOI: <https://doi.org/10.5281/zenodo.14250606>.
12. Elie, L. “Optimising digital forensics investigations in containers as a service environments”. *M.S. thesis, School of Computing, National College of Ireland*. Dublin, Ireland. 2024. – Available from: <https://norma.ncirl.ie/8171>. – [Accessed: March 17, 2025].

13. Kosaka, G. “Security controls for the OWASP Kubernetes Top 10”. *SUSE Communities Blog*. 2024. – Available from: <https://www.suse.com/c/security-controls-for-the-owasp-kubernetes-top-10>. – [Accessed: March 17, 2025].

14. El Khairi, A., Caselli, M., Knierim, C., Peter, A. & Continella, A. “Contextualizing system calls in containers for anomaly-based intrusion detection”. *Proc. 2022 Cloud Computing Security Workshop (CCSW '22)*, Los Angeles, CA, USA. 2022. p. 9–21. DOI: <https://doi.org/10.1145/3560810.3564266>.

15. Lim, S. Y., Stelea, B., Han, X. & Pasquier, T. “Secure namespaced kernel audit for containers”. *Proc. ACM Symp. on Cloud Computing (SoCC '21)*. Seattle, WA, USA. 2021. p. 518–532. DOI: <https://doi.org/10.1145/3472883.3486976>.

16. “Object names and IDs”. *Kubernetes documentation*. – Available from: <https://kubernetes.io/docs/concepts/overview/working-with-objects/names>. – [Accessed: March 17, 2026].

Отримано 02.09.2026

Отримано після перегляду 05.10.2026

Прийнято 12.10.2026

DOI: <https://doi.org/10.15276/ict.03.2026.73>

UDC 004.056

A model of orchestration metadata for attack detection in containerized kubernetes environments

Oleksandr I. Sirenko

ORCID: <https://orcid.org/0000-0001-9751-2544>; olexandr.sirenko@gmail.com

State University of Intelligent Technologies and Telecommunications, 1, Kuznechna street. Odesa, 65023, Ukraine

ABSTRACT

The aim of the article is to formalize an event model of a containerized application extended with Kubernetes orchestration metadata, and to build a correspondence matrix, constructed through a formalized coding protocol, between this model and the ten risk categories of the Open Web Application Security Project Kubernetes Top 10. The relevance of the study stems from the fact that traditional intrusion detection systems built solely on system-call analysis do not account for Kubernetes orchestration context - namespace membership, role-based access control role, pod security profile, network policy, workload lifecycle phase, and cloud-identity binding - so identical system calls receive different risk assessments depending on context. The scientific novelty lies in the typed formal definition of the event structure with a stable entity identifier, a two-stage binding function, and explicit semantics for missing or stale metadata values, together with a correspondence matrix between six orchestration-metadata types and ten risk categories on a three-level necessity scale, built through an explicit, documented coding protocol. The subject of the study is the relationship between orchestration-metadata types and the contextual observability of containerized-environment risk categories at the process-event level - the ability to attribute a process event to a risk category's mechanism, not to detect the presence of the risk in the cluster. The research methods include formalization, structural analysis of the official OWASP Kubernetes Top 10 category descriptions, and a greedy algorithm for prioritizing metadata-type implementation with a full deterministic tie-breaking rule. The results show that the pod security profile and identity-mount metadata type is the most universal, being necessary for three risk categories, whereas three categories - overly exposed platform components, misconfigured and vulnerable cluster components, and inadequate logging and monitoring - are structurally not contextually observable by any of the examined metadata types at the process-event level, outlining the model's boundary of applicability. The greedy algorithm determined an implementation order that achieves full necessary-metadata coverage of the seven contextually observable risk categories, within the defined model, after just four steps. The study is analytical and formal rather than empirical in nature, which is stated as its limitation and direction for future work.

Keywords: Kubernetes; intrusion detection; system calls; orchestration metadata; contextual observability; role-based access control; network policies

ІНФОРМАЦІЯ ПРО АВТОРА



Сіренко Олександр Іванович - аспірант, Державний університет інтелектуальних технологій і зв'язку, вул. Кузнечна, 1. Одеса, 65023, Україна

Галузь досліджень: Кібербезпека та захист інформації

Oleksandr I. Sirenko - PhD student, State University of Intelligent Technologies and Telecommunications, 1, Kuznechna Str. Odesa, 65023, Ukraine

ORCID: <https://orcid.org/0000-0001-9751-2544>; olexandr.sirenko@gmail.com

Research field: Cybersecurity and Information Security