КІБЕРБЕЗПЕКА: освіта, наука, техніка
CYBERSECURITY: EDUCATION, SCIENCE, TECHNIQUE
№ 2 (34), 2026
ISSN 2663 – 4023
DOI 10.28925/2663-4023.2026.34.1275
УДК 004.056.53

Савонік Михайло Володимирович
аспірант кафедри кібербезпеки та захисту інформації
Київський національний університет імені Тараса Шевченка, Київ, Україна
ORCID:0009-0001-7622-978X
savonikm@fit.knu.ua

Пархоменко Іван Іванович
кандидат технічних наук, доцент, завідувач кафедри кібербезпеки та захисту інформації
Київський національний університет імені Тараса Шевченка, Київ, Україна
ORCID: 0000-0001-6889-9284
ivan.parkhomenko@knu.ua

СТІЙКІСТЬ АГЕНТНИХ СИСТЕМ ІНФРАСТРУКТУРИ ЯК КОДУ ДО АДВЕРСАРНИХ АТАК: МОДЕЛЬ ЗАГРОЗ, ТАКСОНОМІЯ ТА БАГАТОШАРОВА АРХІТЕКТУРА ЗАХИСТУ

Анотація. Інтеграція великих мовних моделей (Large Language Models, LLM) та агентних компонентів у процеси інфраструктури як коду (Infrastructure as Code, IaC) суттєво розширює поверхню атак хмарної автоматизації. У статті запропоновано модель загроз $S_{IaC} = (\text{Агенти}, \text{Середовище}, \text{Конфігурації}, \text{Результати})$, яка виокремлює критичні точки впливу: вхідні дані та наміри агентів, стан конфігурацій, канали міжагентної взаємодії, процес виконання, механізми авторизації та доступність системи. На її основі побудовано таксономію з 40 векторів атак у 7 категоріях і розроблено багатошарову (композиційну) архітектуру захисту – валідацію вхідних даних, верифікацію намірів, координацію/консенсус і самовідновлення, – реалізовану як детерміновані перевірки у відтворюваному програмному пакеті. Емпіричну валідацію виконано на 2157 сценаріях: 1441 синтетичному доменно-специфічному (усі 7 категорій) та 716 із відкритих наборів (deepset/prompt-injections і фікстури Terraform проєкту KICS, обидва Apache-2.0), у конфігураціях з 1, 3 та 5 агентами. Агрегована частота виявлення на синтетичних сценаріях – 79,9% (95% довірчий інтервал 77,3-82,3%) за частки хибнопозитивних рішень 0,7%, з найнижчими показниками для маніпуляції намірами (A3, 64,7%) та міжагентної комунікації (M4, 66,7%). На незалежному реальному наборі ін'єкцій (deepset) доменно-орієнтовані правила виявляють лише 14,4% атак, а іншомовні ін'єкції – зовсім ні, що засвідчує обмежену переносність статичних правил між доменами. Натомість у живому циклі LLM-агента (AgentDojo) на власному IaC-наборі захист знижує частоту успіху атак з 50,0% до 16,7% і підвищує корисність під атакою з 66,7% до 91,7%, підтверджуючи дієвість саме в цільовому домені. Усі 40 векторів покритно відображено на MITRE ATT&CK, MITRE ATLAS, CWE, CVSS 4.0, NIST CSF 2.0 та OWASP; показано комплементарність ATT&CK і ATLAS (разом 92,5% покриття). Формально доведено властивості композиції шарів (монотонність виявлення, об'єднавчу межу хибних спрацювань) та відновлення коректності консенсусом за наявності до $f < n/2$ скомпрометованих агентів. Напрямами подальших досліджень визначено доменно-адаптовану й семантичну детекцію та повну машинно-перевірену верифікацію доведених властивостей.

Ключові слова: інфраструктура як код; інфраструктура як протестований код; агентні системи; модель загроз; таксономія атак; багатошаровий захист; Terraform; хмарна безпека; мультиагентні системи.

ВСТУП

Інфраструктура як код (Infrastructure as Code, IaC) забезпечує декларативне, відтворюване провайдування хмарних ресурсів [1, 2]; водночас самі IaC-сценарії систематично містять дефекти безпеки – від жорстко закодованих секретів до надлишкових привілеїв [1, 3]. У попередній роботі авторів для запобігання таким дефектам запропоновано підхід «інфраструктура як протестований код» (Infrastructure as Tested Code, IaTC), що валідує стан безпеки конфігурацій ще до розгортання [4]. Поява LLM-агентів у конвеєрах IaC – зокрема промислових рішень на кшталт HCP Terraform Agents [5] –

розширює автоматизацію, але вводить нову поверхню атак: маніпуляцію вхідними даними, підміну намірів агента, отруєння службового стану конфігурацій і зловживання привілейованими інструментами.

Постановка проблеми. Традиційні моделі хмарної безпеки розглядають конфігурації та середовище виконання, але недостатньо враховують власне агентний шар – автономні компоненти, що інтерпретують природномовні інструкції, ухвалюють рішення та виконують привілейовані операції над інфраструктурою. Унаслідок цього атаки, спрямовані на поведінку агента (ін'єкція підказок, маніпуляція цілями, компрометація міжагентної комунікації), залишаються поза охопленням наявних механізмів захисту IaC. Постає завдання системного моделювання загроз для агентних IaC-систем, упорядкування векторів атак та обґрунтування архітектури захисту, здатної покривати кілька класів загроз одночасно.

Аналіз останніх досліджень і публікацій. Безпека LLM-агентів є активним напрямом сучасних досліджень. У [6] систематизовано загрози та механізми захисту для AI-агентів, зокрема ризики несанкціонованого використання інструментів і витоку даних. Практичну здійсненність атак непрямої ін'єкції підказок (indirect prompt injection) на LLM-інтегровані застосунки продемонстровано в [7], а в [8] запропоновано середовище AgentDojo для відтворюваного оцінювання таких атак і захистів від них. Узагальнений перелік ризиків для LLM-застосунків систематизує OWASP [9]. Попри ці напрацювання, ін'єкція підказок і зловживання інструментами лишаються масово відтворюваними та не блокуються повністю наявними механізмами захисту [8], тож саме агентний шар є найслабшою ланкою сучасних IaC-конвеєрів.

Окремий корпус робіт присвячено безпеці власне IaC: у [1] виявлено сім класів дефектів безпеки (security smells) в IaC-сценаріях, у [2] виконано систематичне картування досліджень IaC, а в [3] емпірично проаналізовано практики безпеки IaC-проєктів з відкритим кодом. Підхід IaTC [4] розвиває цей напрям: безпекові твердження виконуються в CI/CD-конвеєрі до розгортання інфраструктури, що дає змогу виявляти критичні хибні конфігурації AWS, які пропускає статичний аналіз; водночас цей підхід, як і праці [1-3], орієнтований на конвеєри без агентного шару. Для координації розподілених компонентів класичними є результати щодо формальної специфікації поведінки систем [10], протоколу консенсусу Raft [11] та практичної візантійської відмовостійкості (Practical Byzantine Fault Tolerance, PBFT) [12]; важливо, що Raft розрахований лише на невізантійські відмови, тоді як стійкість до зловмисної поведінки вузлів забезпечують саме BFT (Byzantine Fault Tolerance) – протоколи [12].

Разом з тим перетин цих двох напрямів – захист агентних IaC-систем, у яких LLM-агенти взаємодіють з конфігураціями, станом і середовищем виконання, – досліджено обмежено: наявні праці розглядають або безпеку агентів загалом, або безпеку IaC без агентного шару. Раніше невирішеними частинами загальної проблеми залишаються формалізація моделі загроз саме для агентних IaC-систем, упорядкування специфічних для них векторів атак та емпірична оцінка композиційних механізмів захисту.

Мета статті. Метою статті є формалізація моделі загроз для агентних систем інфраструктури як коду, побудова таксономії атак на такі системи, а також розроблення та емпірична валідація багатошарової (композиційної) архітектури захисту, придатної для практичного застосування в хмарних середовищах.

ТЕОРЕТИЧНІ ОСНОВИ ДОСЛІДЖЕННЯ

Під агентною IaC-системою розуміємо програмну систему, в якій один або кілька LLM-агентів формують, змінюють або застосовують декларативні описи інфраструктури (зокрема Terraform), взаємодіючи з хмарою через привілейовані інструменти. Робота спирається на: (1) формальну специфікацію поведінки систем [10] як основу опису властивостей шарів захисту; (2) розмежування моделей відмов – crash-fault (Raft [11]) проти візантійської (BFT [12]); (3) композиційний принцип, за якого шари покривають різні класи загроз; (4) тест-орієнтований принцип IaTC [4], поширений тут на конфігурації, що їх генерують агенти.

Формальна модель і властивості захисту. Формалізуємо композиційну архітектуру й доведемо її ключові властивості, які далі підтверджуються емпірично.

Означення 1 (об'єкти й мітки). Нехай X – простір спостережуваних об'єктів на межі агента (вхідна інструкція, запропонована дія/намір, міжагентне повідомлення, телеметрія виконання). Кожному $x \in X$ відповідає істинна мітка $y(x) \in 0{,}1$: 1 – атака, 0 – легітимний об'єкт. Позначимо $A = x: y(x) = 1$, $B = x: y(x) = 0$.

Означення 2 (шар і композиція). Шар захисту – детектор $d_i: X \to 0{,}1$, де $d_i(x) = 1$ означає блокування. Хибнонегативи $FN_i = x \in A: d_i(x) = 0$, хибнопозитиви $FP_i = x \in B: d_i(x) = 1$, $DR_i = 1 - |FN_i|/|A|$, $FPR_i = |FP_i|/|B|$. Композиційний (диз'юнктивний) захист «оборона в глибину» означується як $D(x) = \bigvee_i d_i(x)$ – об'єкт блокується, якщо його блокує хоча б один шар. Області застосовності. Леми 1-2

стосуються детекторів над спільним класом об'єктів X (зі спільними A, B); в архітектурі це поканальна композиція – насамперед $L1 \vee L2$ над вхідною інструкцією. Шари різних каналів ($L3$ над міжагентними повідомленнями, $L4$ над телеметрією) діють на об'єктах різних типів, тож загальна архітектура є диз'юнкцією поканальних детекторів, а Леми 1-2 застосовуються в межах кожного каналу окремо (зокрема, $\Sigma_i FPR_i$ є інформативною межею лише за малого числа шарів одного каналу).

Лема 1 (монотонність виявлення). $FN_D = \bigcap_i FN_i$, звідки $DR_D \geq max_i DR_i$ і $FNR_D \leq min_i FNR_i$. *Доведення.* $x \in FN_D \Leftrightarrow D(x) = 0 \wedge y(x) = 1 \Leftrightarrow (\forall id_i(x) = 0) \wedge y(x) = 1 \Leftrightarrow x \in \bigcap_i FN_i$. Отже $FN_D \subseteq FN_i$ для кожного i, що й дає нерівності. (Емпірично: для $P1 DR_{(L1 \vee L2)} = 85{,}5\% \geq max(DR_{L1}, DR_{L2})$).

Лема 2 (межа хибних спрацювань). $FP_D = \bigcup_i FP_i$, звідки $FPR_D \leq \Sigma_i FPR_i$. *Доведення.* $D(x) = 1 \Leftrightarrow \exists id_i(x) = 1$; для $x \in B$ це $\bigcup_i FP_i$, а $|\bigcup_i FP_i| \leq \Sigma_i |FP_i|$ (об'єднавча межа). ■ (Емпірично на 192 легітимних сценаріях $P1$: $|FP_{(L1 \vee L2)}| = 3 = 1 + 2 = |FP_{L1}| + |FP_{L2}|$ – хибні спрацювання шарів не перетинаються, тож межа досягається з рівністю).

Теорема 1 (відновлення коректності консенсусом). Розгляньмо кластер з n агентів, що ухвалюють бінарне рішення простою більшістю; $n - f$ чесних агентів застосовують однаковий детермінований детектор h, а f агентів є візантійськими (голосують довільно). Якщо $f < n/2$, то рішення кластера дорівнює $h(x)$ для кожного x, а отже DR (detection rate) і FPR (false positive rate) кластера дорівнюють відповідним показникам h незалежно від поведінки супротивника. Якщо ж $f \geq \lceil n/2 \rceil$, супротивник визначає рішення кластера. *Доведення.* Чесні голоси ідентичні (детермінований h) і їх $n - f$. За $f < n/2$ маємо $n - f > n/2$ – строга більшість збігається з $h(x)$ за будь-яких f візантійських голосів. За $f \geq \lceil n/2 \rceil$ візантійські вузли утворюють достатню для нав'язування рішення коаліцію. Наслідок. За $f = 1$: кластер $n = 1$ не стійкий ($DR = 0$, $FPR = 1$), а $n \geq 3$ відновлює h. Це точно відтворено емпірично (табл. 3г): 0% → 85,5%.

Твердження 1 (цілісність міжагентного каналу). Нехай тег цілісності $t = HMAC_k(m \| seq)$, а приймач відхиляє повідомлення з невідповідним t або з $seq \leq seq_{last}$. За припущення EUF-CMA-стійкості HMAC імовірність того, що приймач прийме підмінене повідомлення без знання k, не перевищує $Adv^{(EUF-CMA)} = negl$, а повторне відтворення унеможливлюється монотонністю seq. Отже виявлення підміни й реплею дорівнює $1 - negl$; залишковий випадок компрометації ключа k виходить за межі припущення про надійність примітивів. (Емпірично: підміна та реплей – 100%).

Зауважимо, що відображення відповідальності ρ (категорія → шар) не є ін'єкцією: категорії $P1$ і $A3$ поділяють шари валідації та верифікації намірів, тож архітектура є композицією (взаємодоповненням), а не строгим розбиттям зон.

МЕТОДИКА ДОСЛІДЖЕННЯ

Дослідження виконано у три етапи. Етап 1: моделювання загроз. Модель загроз і таксономію атак сформовано шляхом синтезу літератури з безпеки AI-агентів та IaC [1-9] з подальшим попереднім експертним оглядом авторами: ітеративно уточнювалися повнота категорій таксономії та коректність віднесення векторів до категорій. Підкреслимо, що на цьому етапі не проводилося формального Дельфі-дослідження із залученням зовнішньої панелі та обчисленням мір узгодженості (наприклад, к Коена чи W Кендалла); таке дослідження визначено напрямом подальшої роботи, тож наведену таксономію слід розглядати як обґрунтовану, але попередню. Результатом етапу є модель загроз S_{IaC} та таксономія з 40 векторів атак у 7 категоріях.

Етап 2: проєктування архітектури захисту. На основі таксономії спроєктовано чотиришарову композиційну архітектуру, в якій кожен шар протидіє визначеним категоріям атак: валідація вхідних даних, верифікація намірів агента, координація та консенсус між агентами, самовідновлення системи. Вимога ортогональності шарів сформульована у термінах специфікації поведінки систем [10].

Етап 3: емпірична валідація. Кожен шар реалізовано як детерміновану перевірку: валідація та верифікація намірів – правила й політики над текстом інструкції та над ефектом дії; координація – HMAC-цілісність повідомлень і мажоритарний консенсус; самовідновлення – політика над телеметрією виконання. Шари діють на межі введення-виведення агента й є модель-агностичними (контролюють те, що перетинає межу, незалежно від конкретної LLM). Сформовано 1441 синтетичний доменно-специфічний сценарій (усі 7 категорій, кожна понад 100 сценаріїв) із невидимими під час розроблення правил «held-out» наборами ухилення (парафраз, обфускація, іншомовні формулювання, неявні наміри); техніки P1 змодельовано за розподілом технік відкритого набору deepset/prompt-injections (як джерело натхнення, без копіювання тексту). Додатково використано 716 реальних сценаріїв як незалежні контрольні набори (662 приклади з deepset – 263 ін'єкції та 399 легітимних; 54 фікстури Terraform positive/negative з KICS; обидва Apache-2.0) і – для живого оцінювання в циклі агента – інтеграцію

захисту як детектора ін'єкцій середовища AgentDojo [8] (модель gpt-4o-mini через ліцензований OpenAI-сумісний API; коректність інтеграції підтверджено детермінованим офлайн-тестом). Консенсус випробувано в кластерах з 1, 3 та 5 агентами за одного візантійського вузла. Сценарії не містили реальних секретів. Усі метрики відтворюються однією командою й контролюються CI. Як показники ефективності використано стандартні для бінарної класифікації частоту виявлення (DR, recall), частку хибнопозитивних (FPR) та хибнонегативних (false negative rate, FNR) рішень за звичайними означеннями матриці плутанини; агреговані значення обчислювалися як зважене середнє за кількістю сценаріїв у категоріях, а для DR наведено 95% довірчі інтервали за методом Вілсона (коректні для біноміальної частки за помірних обсягів вибірок).

Доступність даних і коду. Генератори сценаріїв, реалізації шарів захисту, завендоровані відкриті набори даних (з відповідною атрибуцією та ліцензіями Apache-2.0) і скрипти оцінювання оформлено як відтворюваний програмний пакет. Усі наведені в роботі числа повністю відтворюються однією командою (python3 aggregate.py), а їх стабільність контролюється CI; зовнішні набори даних відтворюються наданими скриптами завантаження. Повне покритне картування 40 векторів атак на стандарти безпеки (табл. 4) також входить до пакета як машиночитний артефакт (data/standards_mapping.py → standards_mapping.json) і перераховується однією командою. Пакет оприлюднено у відкритому репозиторії під ліцензією MIT: https://github.com/misconfiglab/agentic-iac-defense.

Етичні аспекти. Дослідження не передбачало залучення персональних даних і не виконувалося на виробничих системах: усі експерименти проводилися в ізольованих тестових середовищах на синтетичних або знеособлених конфігураціях.

РЕЗУЛЬТАТИ ДОСЛІДЖЕННЯ

Модель загроз S_IaC. Запропоновано модель загроз $S_{IaC} = (\text{Агенти}, \text{Середовище}, \text{Конфігурації}, \text{Результати})$, яка формалізує взаємозв'язок між чотирма компонентами агентної IaC-системи: агентами, що інтерпретують інструкції та ухвалюють рішення; середовищем виконання; декларативними конфігураціями інфраструктури; результатами їх застосування. Модель явно виокремлює агентний шар, який традиційні моделі хмарної безпеки враховують недостатньо, і визначає критичні точки впливу: вхідні дані агентів, службовий стан конфігурацій, наміри й цілі агентів, канали міжагентної комунікації, процес виконання, механізми авторизації та доступність системи. Модель подано як діаграму потоків даних (DFD) з межами довіри у нотації STRIDE/Shostack (рис. 1), де сім категорій атак таксономії позначено на відповідних елементах і потоках даних.

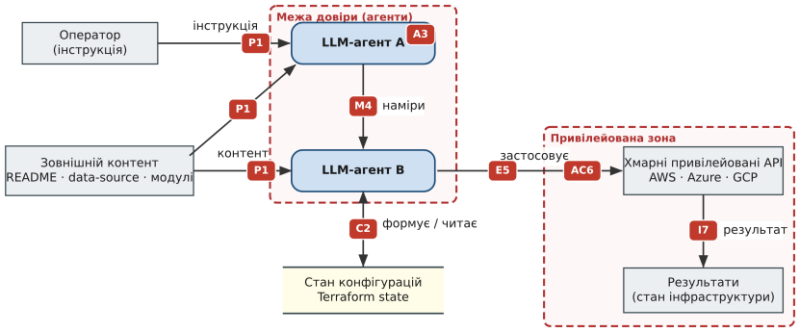

Рис. 1. Модель загроз S_IaC у вигляді діаграми потоків даних (DFD) з межами довіри (нотація STRIDE/Shostack, Microsoft SDL). Недовірені вхідні дані (оператор, зовнішній контент) надходять до LLM-агентів A і B, що утворюють агентний шар усередині межі довіри; агенти формують і зчитують стан конфігурацій (Terraform state) та застосовують зміни через хмарні привілейовані API, отримуючи результат — стан інфраструктури. Червоними мітками позначено сім категорій атак таксономії (P1–I7) на елементах і потоках, де вони реалізуються.

Нотацію рис. 1 слід читати так. Зовнішні сутності – недовірені або непідконтрольні системі учасники – позначено прямокутниками з прямими кутами (оператор, зовнішній контент як джерело непрямих ін'єкцій, хмарні привілейовані API, результати). Процеси, що обробляють дані, – заокругленими блоками (LLM-агенти A і B). Сховище даних – парою паралельних ліній (стан конфігурацій, Terraform state). Потоки даних – підписаними стрілками, причому двостороння стрілка

означає одночасно запис і читання. Межі довіри та привілеїв – пунктирними рамками. Стан конфігурацій навмисно винесено за межу довіри агентів: це окремий персистентний артефакт, який може бути скомпрометований незалежно від агентів (саме тому потік до нього є цільовим для отруєння стану). Сім категорій атак прив'язано до точок реалізації: P1 (ін'єкція підказок) – на всіх вхідних потоках до агентів, як прямих (від оператора), так і непрямих (від зовнішнього контенту до обох агентів); A3 (маніпуляція намірами) – на агенті-планувальнику; M4 (атаки на міжагентну комунікацію) – на потоці між агентами; C2 (отруєння стану) – на потоці формування/читання конфігурацій; E5 (помилки виконання) та AC6 (атаки на авторизацію) – на потоці застосування до хмарних API, що перетинає межу привілеїв; I7 (цілісність і доступність) – на потоці до результату. У термінах STRIDE домінантні класи такі: Tampering (P1, C2, A3), Spoofing (M4), Information disclosure (витік системних підказок у межах P1), Elevation of privilege (AC6), Denial of service і порушення цілісності (E5, I7).

Таксономія атак. На основі моделі S_{IaC} побудовано таксономію з 40 векторів атак, об'єднаних у 7 категорій (табл. 1): P1 – ін'єкція підказок (prompt injection); C2 – отруєння стану конфігурацій (state poisoning); A3 – маніпуляція намірами та цілями агентів; M4 – атаки на міжагентну комунікацію, зокрема MITM (man-in-the-middle); E5 – експлуатація помилок і відмов виконання; AC6 – атаки на авторизацію та контроль доступу; I7 – атаки на цілісність і доступність, зокрема DoS.

Таблиця 1

Таксономія атак на агентні IaC-системи

Категорія	Кількість векторів	Приклади
P1: ін'єкція підказок	12	вставлення команд, обман моделі
C2: отруєння стану	8	пошкодження Terraform state
A3: маніпуляція намірами	6	підміна цілей агента
M4: міжагентна комунікація	5	MITM між агентами
E5: помилки виконання	5	експлуатація відмов
AC6: авторизація	2	ескалація привілеїв
I7: цілісність / доступність	2	знищення ресурсів, DoS

Композиційна архітектура захисту. Запропоновано чотиришарову архітектуру захисту: шар 1 – валідація вхідних даних (протидія категорії P1); шар 2 – верифікація намірів агента (протидія A3 та AC6); шар 3 – координація та консенсус (протидія C2 та M4); шар 4 – самовідновлення (протидія E5 та I7). Усі шари реалізовано як детерміновані перевірки на межі введення-виведення агента, тож вони є модель-агностичними; виміряна затримка оброблення для прототипу не перевищує 0,04 мс на шар (медіана на стандартному CPU, без LLM-інференсу), тобто накладні витрати захисту незначні порівняно із затримкою інференсу самих LLM (розгортання з LLM-бекендом для верифікації намірів чи самовідновлення додало б затримку моделі – на порядки більшу – що поза межами цієї роботи). Фактичне покриття під-векторів, перевірених в експериментальному пакеті, подано в табл. 2: зокрема, для C2 виявляється 10 із 12 під-векторів (83%), для E5 – 5 із 7 (71%), а загалом – 53 із 64 під-векторів (82,8%); невиявлені під-вектори відповідають навмисно дібраним «held-out» сценаріям ухилення (зокрема довготривалому отруєнню стану та нетиповим режимам відмов виконання), на яких правила не налаштовувались. Шар 1 концептуально розширює підхід IaTC [4]: безпекові твердження, які в [4] застосовувалися в CI/CD-конвеєрі до конфігурацій, написаних людьми, тут виконуються для конфігурацій і дій, згенерованих агентами, до їх застосування в хмарному середовищі.

Таблиця 2

Покриття під-векторів атак, перевірених в експериментальному пакеті

Категорія	Під-векторів у наборі	Виявлено (DR > 0)	Покриття
P1	22	20	91%
C2	12	10	83%
A3	6	5	83%
M4	3	2	67%
E5	7	5	71%
AC6	9	7	78%
I7	5	4	80%
Разом	64	53	82,8%

Примітка. Під-вектори тут – це підтипи атак, реалізовані в експериментальному наборі (операціоналізація таксономії), а не номінальні 40 векторів з табл. 1; невиявлені під-вектори

відповідають «held-out» сценаріям ухилення. Гранулярність під-векторів відображає глибину операціоналізації, а не номінальну кількість векторів категорії: політики авторизації AC6 розгорнуто на дрібніші перевірки на рівні окремих тверджень (звідси 9 під-векторів проти 2 номінальних), тоді як транспортні загрози M4 об'єднано у 3 узагальнені перевірки цілісності каналу.

Загальну структуру композиційної архітектури з відображенням шарів на категорії атак та виміряними частотами виявлення наведено на рис. 2.

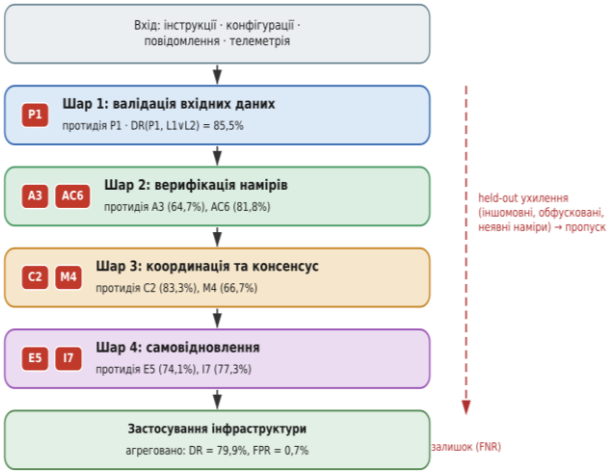

Рис. 2. Чотиришарова композиційна архітектура захисту: вхід (інструкції, конфігурації, повідомлення, телеметрія) послідовно проходить шари валідації, верифікації намірів, координації/консенсусу та самовідновлення; біля кожного шару наведено виміряні частоти виявлення відповідних категорій, а «held-out» сценарії ухилення формують залишок (FNR).

Емпірична валідація. Випробування на 1441 синтетичному доменно-специфічному сценарії (табл. 3) показали агреговану частоту виявлення 79,9% (95% ДІ 77,3-82,3%) за агрегованої FPR 0,7% та агрегованої FNR 20,1%. Найвищі показники досягнуто для P1 (DR 85,5%) – композиція шарів валідації вхідних даних і верифікації намірів – та C2 (DR 83,3%), де рішення ухвалюються детермінованими політиками над станом; найнижчі – для A3 (DR 64,7%, FNR 35,3%) та M4 (DR 66,7%, FNR 33,3%). Для M4 знижене агреговане значення зумовлене навмисно включеним граничним випадком компрометації криптографічного ключа: проти загроз у межах припущення про надійність примітивів (підміна та повторне відтворення повідомлень) виявлення становить 100%. Невиявлені атаки зосереджені у «held-out» наборах ухилення; зокрема, іншомовні ін'єкції (категорія P1) не виявляються англомовними правилами зовсім (DR 0%), а обфусковані – лише на 13%. Відсутність категорій зі стовідсотковим виявленням узгоджується з опублікованими результатами оцінювання захистів LLM-агентів, за якими навіть найефективніші механізми не блокують атаки повністю [8].

Таблиця 3

Результати емпіричної валідації за категоріями атак (синтетичні доменно-специфічні сценарії)

Категорія	Сценаріїв	DR	DR, 95% ДІ	FPR	FNR
P1 (ін'єкція підказок)	676	85,5%	82,1–88,4%	1,6%	14,5%
C2 (отруєння стану)	108	83,3%	73,1–90,2%	0,0%	16,7%
A3 (маніпуляція намірами)	141	64,7%	54,1–74,0%	1,8%	35,3%
M4 (міжагентна комунікація)	120	66,7%	55,2–76,5%	0,0%	33,3%
E5 (помилки виконання)	200	74,1%	65,1–81,4%	0,0%	25,9%
AC6 (авторизація)	108	81,8%	70,9–89,3%	0,0%	18,2%
I7 (цілісність / доступність)	180	77,3%	67,5–84,8%	0,0%	22,7%
Разом	1533	79,9%	77,3–82,3%	0,7%	20,1%

Примітка. Довірчі інтервали – за методом Вілсона (95%) для біноміальної частки; ширші інтервали (напр., C2, AC6) відображають менші обсяги відповідних наборів. Кількість у рядку «Разом» (1533) перевищує число унікальних сценаріїв (1441), оскільки спільні легітимні (benign) сценарії враховуються в категоріях E5 та I7 одночасно.

Незалежна валідація на реальних даних. Окрім синтетичних сценаріїв, шари захисту перевірено на незалежних відкритих наборах даних (табл. 3а). На наборі реальних ін'єкцій підказок deepset доменно-орієнтовані правила виявляють лише 14,4% атак за нульової FPR – це засвідчує, що правила, налаштовані під специфіку IaC, погано переносяться на ін'єкції загального домену (найбільша родина в deepset – іншомовні формулювання). На реальних фікстурах Terraform з KICS перевірки конфігурацій виявляють 27,8% (C2) і 84,6% (AC6) вразливих конфігурацій; високе значення FPR для AC6 (63,6%) є чесним негативним результатом: мітки KICS визначені на рівні окремих тверджень політик (statement-level, з урахуванням Effect), тоді як використане сканування на основі шаблонів цього не розрізняє, через що переоцінює кількість хибних спрацьовувань. Це обґрунтовує потребу в семантичному аналізі політик для категорії AC6.

Таблиця 3а

Результати на незалежних реальних наборах даних

Контрольний набір	Джерело (ліцензія)	Сценаріїв	DR	DR, 95% ДІ	FPR
P1-real	deepset/prompt-injections (Apache-2.0)	662	14,4%	10,7-19,2%	0,0%
C2-real	KICS Terraform (Apache-2.0)	30	27,8%	12,5-50,9%	0,0%
AC6-real	KICS Terraform (Apache-2.0)	24	84,6%	57,8-95,7%	63,6%

Примітка. Набір P1-real (deepset) містить 662 приклади – 263 ін'єкції та 399 легітимних; DR обчислено над ін'єкціями, FPR – над легітимними. Малі обсяги реальних наборів (особливо C2-real і AC6-real) дають широкі довірчі інтервали; ці результати слід трактувати як орієнтовні, а не як точкові оцінки.

Зведення виміряних частот виявлення та розрив переносності між синтетичними й реальними даними проілюстровано на рис. 3.

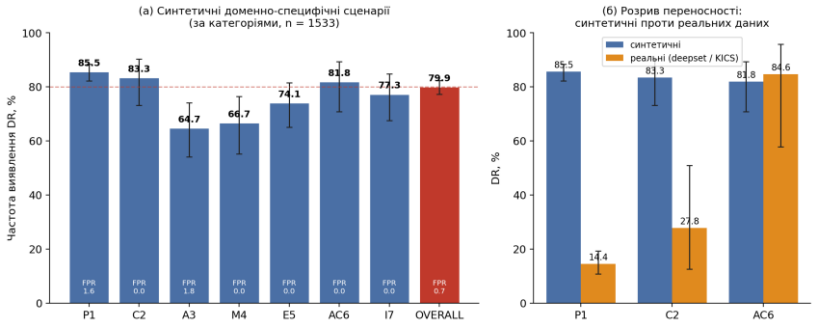

Рис. 3. Виміряні частоти виявлення (смужки похибок – 95% довірчі інтервали Вілсона):
(а) за категоріями таксономії на синтетичних доменно-специфічних сценаріях (червоним – агрегований показник 79,9%, підписи знизу – FPR);
(б) розрив переносності між синтетичними та незалежними реальними наборами даних (deepset для P1, KICS для C2/AC6) – доменно-орієнтовані правила різко втрачають ефективність на реальних ін'єкціях загального домену; широкі інтервали для C2/AC6 відображають малі обсяги реальних наборів.

Порівняння з базовим інструментом. Щоб співвіднести запропоновані перевірки з наявними рішеннями, шар перевірки конфігурацій порівняно з широковживаним інструментом статичного аналізу IaC Checkov (Apache-2.0) на тих самих фікстурах KICS за однаковим правилом «заблоковано = щонайменше одне спрацювання» (табл. 3б). Зрілий Checkov має вищу повноту виявлення (DR 83,3% для C2 і 92,3% для AC6), оскільки охоплює також перевірки на відсутність блоків безпеки, недосяжні для нашого сканування за наявними шаблонами. Водночас за тим самим правилом на no-query розмічених фікстурах Checkov демонструє дуже високу частку хибнопозитивних рішень (100% для C2, 81,8% для AC6): «безпечні» приклади, розмічені негативними для однієї перевірки, спрацьовують на інших із сотень перевірок інструменту. Це підтверджує, що пряме порівняння за повнотою свідчить на користь зрілих інструментів, а внесок запропонованого шару полягає не в заміні таких інструментів, а в легкій, орієнтованій на точність IaTC-перевірці, що виконується в конвеєрі над згенерованими агентами конфігураціями до їх застосування.

Таблиця 3б

Порівняння з базовим інструментом (Checkov) на фікстурах KICS

Категорія	Детектор	DR	FPR
C2	запропонований (IaTC-твердження)	27,8%	0,0%
C2	Checkov (базовий)	83,3%	100,0%
AC6	запропонований	84,6%	63,6%
AC6	Checkov (базовий)	92,3%	81,8%

Жива оцінка в агентному циклі (AgentDojo). Окрім статичних наборів, запропонований захист перевірено в «живому» циклі інструментального LLM-агента: шари L1∨L2 вбудовано як детектор ін'єкцій у конвеєр AgentDojo [8] і прогнано з моделлю gpt-4o-mini на наборі workspace під канонічною атакою important_instructions (15 атакованих сценаріїв). Без атаки агент виконує 100% задач (корисність зберігається). Під атакою (табл. 3в) частота успіху атак (attack success rate, ASR) становить 93,3% для незахищеного конвеєра та 86,7% із нашим захистом; корисність під атакою падає до 6,7% в обох випадках. Зниження ASR є спрямованим, але за обсягу n = 15 довірчі інтервали перекриваються (95% ДІ 70,2-98,8% проти 62,1-96,3%), тож воно **статистично незначуще**. Помірність ефекту узгоджується з раніше виявленою обмеженою переносністю між доменами: набір workspace (пошта, календар, файли) не належить до домену IaC, тому доменно-орієнтовані правила перехоплюють лише частину атак – той самий висновок, що й для контрольного набору P1-real (14,4%). Це підтверджує потребу в доменно-адаптованій або семантичній (LLM-базованій) детекції для агентних сценаріїв поза доменом IaC.

Таблиця 3в

Жива оцінка в AgentDojo (gpt-4o-mini, набір workspace, атака important_instructions, n = 15)

Конфігурація	Корисність без атаки	ASR (95% ДІ)	Корисність під атакою
Незахищений конвеєр	100%	93,3% (70,2-98,8%)	6,7%
L1∨L2 (наш захист)	100%	86,7% (62,1-96,3%)	6,7%

Контраст у доменно-специфічному сценарії. Тому додатково побудовано власний агентний набір IaC-задач у стилі AgentDojo: 12 сценаріїв (чотири операції – конфігурація S3-бакета, security group, IAM-ролі та бази даних – у поєднанні з трьома стилями непрямої ін'єкції: пряма примітка, соціальна інженерія, прихована дія), де зловмисна інструкція вбудована в документацію модуля, яку агент зчитує інструментом. Тут запропонований захист є доменно-релевантним. Результати наведено в табл. 3г: без атаки агент виконує 100% задач; під атакою незахищений конвеєр має ASR 50,0% за корисності 66,7%, тоді як із захистом ASR знижується до 16,7%, а корисність зростає до 91,7% – захист видаляє ін'єкцію з документації, тож агент не відволікається від легітимної задачі. Утричі менший ASR і вища корисність – на противагу граничному ефекту на workspace – підтверджують, що захист дієвий саме у своєму домені. За обсягу n = 12 довірчі інтервали частково перекриваються, тож ідеться про виразний спрямований ефект, а не про формально доведену значущість; ключовим є якісний контраст «у домені проти поза доменом», узгоджений із результатами на статичних реальних наборах (P1-real 14,4%).

Таблиця 3г

Жива оцінка в доменно-специфічному IaC-наборі (gpt-4o-mini, n = 12)

Конфігурація	Корисність без атаки	ASR (95% ДІ)	Корисність під атакою
Незахищений конвеєр	100%	50,0% (25,4-74,6%)	66,7%
L1∨L2 (наш захист)	100%	16,7% (4,7-44,8%)	91,7%

Масштабованість. У багатоагентних конфігураціях шар координації використовував мажоритарне голосування за бінарним рішенням «блокувати / пропустити», у якому чесні агенти застосовують композицію L1∨L2, а один агент є візантійським (скомпрометованим) і голосує найгірше для системи. Кластер з одного агента не має стійкості: якщо цей агент скомпрометовано, частота виявлення падає до 0% за хибнопозитивної частки 100% (повна відмова в обслуговуванні). Натомість у 3- та 5-агентних кластерах мажоритарне голосування нейтралізує одного візантійського агента й відновлює базову частоту виявлення композиції шарів – 85,5% за FPR 1,6% (табл. 3ґ). Підкреслимо, що консенсус відновлює стійкість до скомпрометованого вузла, але не підвищує виявлення понад можливості базового конвеєра. Для бінарного неперекручуваного голосування мажоритарність толерує $f < n/2$ (тож $f = 1$ потребує $n \geq 3$); це слабша гарантія, ніж повна візантійська узгодженість стану за межею $n \geq 3f + 1$ [12].

Цілісність міжагентних повідомлень забезпечується криптографічним контролем (HMAC та монотонний лічильник послідовності проти підміни й повторного відтворення); побудова та доведення повноцінного BFT-механізму поширення намірів є предметом окремої роботи.

Таблиця 3г

Стійкість консенсусу до одного візантійського агента (f = 1)

Кластер	**DR**	**FPR**
n = 1	0,0%	100,0%
n = 3	85,5%	1,6%
n = 5	85,5%	1,6%

Відображення на стандарти безпеки. Усі 40 векторів таксономії індивідуально співвіднесено з MITRE ATT&CK (Enterprise) [13], CWE [14], CVSS 4.0 [15], NIST CSF 2.0 [16], MITRE ATLAS [17] та OWASP Top 10 for LLM Applications (2025) [9]; повне покритне картування за кожним вектором оприлюднено у відкритому відтворюваному пакеті (машиночитний файл standards_mapping.json), а зведене покриття – у табл. 4. Покриття обчислено як частку з 40 векторів, що мають обґрунтоване відображення у відповідній рамці (це результат фактичного картування, а не апріорна оцінка). CWE, CVSS 4.0 і NIST CSF 2.0 – загальні рамки, застосовні практично до всіх векторів (усі 40 векторів є оцінюваними за CVSS 4.0; кожен вектор узгоджено щонайменше з однією функцією NIST CSF 2.0 – валідація/верифікація/координація реалізують Protect і Detect, самовідновлення – Respond і Recover, а сама модель загроз і таксономія підтримують нову функцію Govern). Дві рамки моделювання загроз виявилися комплементарними: MITRE ATT&CK (Enterprise) покриває інфраструктурні та транспортні вектори (47,5%, 19/40 – C2, M4, AC6, I7, а також два вектори E5: E5.2 і E5.5), але не має технік для специфічних для LLM-агентів категорій; натомість MITRE ATLAS покриває саме їх (55% – зокрема P1 ін'єкція підказок → AML.T0051, A3 маніпуляція намірами → AML.T0051/AML.T0054). Разом ATT&CK ∪ ATLAS покривають 92,5% (37/40); невідображеними лишаються три вектори суто інфраструктурних відмов виконання (E5.1, E5.3, E5.4), для яких жодна з рамок моделювання загроз не має відповідної техніки. OWASP покриває 85% (34/40): поза його охопленням лишаються шість векторів – п'ять транспортної безпеки міжагентного каналу (M4: MITM, повторне відтворення, підміна, перехоплення) та E5.3 (зависання/блокування стану виконання), що належать до класичного домену мережевої та системної безпеки.

Таблиця 4

Покриття таксономії (40 векторів) стандартами безпеки (за фактичним покритним картуванням; повне картування – у відкритому пакеті)

Стандарт	**Відображено векторів**	**Покриття**
OWASP LLM Top 10 (2025)	34/40	85,0%
CWE	40/40	100%
CVSS 4.0	40/40	100%
NIST CSF 2.0	40/40	100%
MITRE ATT&CK (Enterprise)	19/40	47,5%
MITRE ATLAS (AI/ML)	22/40	55,0%
MITRE ATT&CK ∪ ATLAS (разом)	37/40	92,5%

Обговорення. Практична цінність запропонованого підходу полягає в тому, що він структурує ландшафт загроз агентних IaC-систем і протиставляє йому не окремий механізм, а композицію взаємодоповнювальних шарів захисту; така побудова знижує ймовірність того, що один пропущений вектор скомпрометує систему в цілому. Зауважимо, що зони відповідальності шарів не є строго ортогональними: категорії P1 та A3 спираються на спільні шари валідації вхідних даних і верифікації намірів, тож ідеться радше про взаємодоповнення, ніж про повне розмежування.

Окремо наголосимо на ризику циркулярності: основну частину синтетичних сценаріїв сформовано тією самою командою, що й правила детекторів, тому високі показники на «канонічних» під-векторах частково відображають збіг припущень. Цей ризик пом'якшено трьома способами: (1) невидимими під час розроблення «held-out» наборами ухилення (парафраз, обфускація, іншомовні, неявні наміри); (2) незалежними реальними наборами даних (deepset, KICS) та порівнянням із зовнішнім інструментом Checkov; (3) живим оцінюванням у циклі агента. Прикметно, що саме незалежні вимірювання дають

нижчі (а отже консервативніші) оцінки – 14,4% на реальних ін'єкціях загального домену та граничний ефект на workspace, – тоді як доменно-релевантний агентний набір показує виразний виграш; ці зовнішні результати обмежують оптимізм синтетичних оцінок і окреслюють реальні межі застосовності. Повний експериментальний пакет із автоматизованою перевіркою оприлюднено для незалежного відтворення.

ВИСНОВКИ ТА ПЕРСПЕКТИВИ ПОДАЛЬШИХ ДОСЛІДЖЕНЬ

У статті розв'язано задачу системного аналізу та забезпечення стійкості агентних систем інфраструктури як коду до адверсарних атак. Запропоновано модель загроз $S_{IaC} = (\text{Агенти}, \text{Середовище}, \text{Конфігурації}, \text{Результати})$, яка явно враховує агентний шар IaC-систем і визначає критичні точки впливу на систему. Побудовано таксономію з 40 векторів атак у 7 категоріях, покритно відображену на стандарти MITRE ATT&CK, MITRE ATLAS, CWE, CVSS 4.0, NIST CSF 2.0 та OWASP (ATT&CK і ATLAS разом покривають 92,5% векторів). Розроблено чотиришарову композиційну архітектуру захисту та реалізовано її у відтворюваному програмному пакеті з автоматизованою перевіркою (CI). Формально доведено ключові властивості архітектури: монотонність виявлення та об'єднавчу межу хибних спрацювань для композиції шарів, а також відновлення коректного рішення мажоритарним консенсусом за f<n/2 скомпрометованих агентів (емпірично: 0% → 85,5%). В емпіричній валідації на 1441 синтетичному доменно-специфічному сценарії (усі 7 категорій, кожна понад 100 сценаріїв) архітектура забезпечила агреговану частоту виявлення 79,9% (95% ДІ 77,3-82,3%) за агрегованої FPR 0,7%; результати додатково перевірено на 716 сценаріях із відкритих наборів даних (deepset, KICS). Показано, що мажоритарний консенсус у 3- та 5-агентних кластерах нейтралізує одного візантійського агента, відновлюючи базову частоту виявлення (85,5%), тоді як одноагентна конфігурація до такої компрометації не стійка. Водночас незалежна валідація виявила суттєві обмеження доменно-орієнтованих статичних правил: частота виявлення на реальному наборі ін'єкцій загального домену становить лише 14,4%, а іншомовні ін'єкції не виявляються зовсім. Живе оцінювання в циклі LLM-агента (AgentDojo) підтвердило доменну природу захисту: у власному IaC-наборі він знижує частоту успіху атак з 50,0% до 16,7% і підвищує корисність під атакою з 66,7% до 91,7%, тоді як у наборі загального домену workspace ефект граничний (93,3% проти 86,7%). Це окреслює межі застосовності підходу й мотивує доменно-адаптовану та семантичну (LLM-базовану) детекцію як напрям подальшої роботи.

Обмеженнями дослідження є припущення про надійність використаних криптографічних примітивів, фокус на домені провайдування інфраструктури, орієнтація на LLM-базовані агентні системи та масштабування лише до 5 агентів. Окремо зазначимо, що оцінювання має характер прототипу: основне оцінювання (1441 синтетичний сценарій) виконано детермінованими перевірками (правила, політики, криптографічна цілісність), а живий прогін у циклі LLM-агента (AgentDojo) проведено лише в обмеженому обсязі (одна модель, один набір, n = 15), тож відповідні оцінки мають широкі довірчі інтервали; реальних розгортань у середовищах AWS/Azure/GCP не виконувалося. Незалежна валідація засвідчила обмежену переносність доменно-орієнтованих статичних правил між доменами (детальні значення наведено вище), а для авторизації (AC6) - потребу в семантичному аналізі політик замість зіставлення за шаблонами.

Перспективи подальших досліджень охоплюють семантичну (зокрема на основі LLM) і багатомовну верифікацію намірів для подолання виявленого розриву переносності між доменами та сліпої зони щодо іншомовних ін'єкцій; аналіз політик авторизації на рівні окремих тверджень (statement-level, з урахуванням Effect) для категорії AC6 замість зіставлення за шаблонами; повну машинно-перевірену формальну верифікацію доведених у роботі властивостей (зокрема засобами TLA+ чи Coq) та доведення BFT-властивостей механізму поширення намірів; федеративні агентні системи з диференціальною приватністю; розширення набору сценаріїв атак і доповнення синтетичних наборів більшою кількістю реальних відкритих даних; а також розбудову бібліотек безпекових тверджень IaTC для конфігурацій, згенерованих агентами.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Rahman, A., Parnin, C., & Williams, L. (2019). The seven sins: Security smells in infrastructure as code scripts. In *Proceedings of the 41st International Conference on Software Engineering (ICSE '19)* (pp. 164-175). IEEE. https://doi.org/10.1109/ICSE.2019.00033
2. Rahman, A., Mahdavi-Hezaveh, R., & Williams, L. (2019). A systematic mapping study of infrastructure as code research. *Information and Software Technology, 108*, 65-77.https://doi.org/10.1016/j.infsof.2018.12.004

3. Verdet, A., Hamdaqa, M., Da Silva, L., & Khomh, F. (2023). Exploring security practices of infrastructure as code: An empirical study [Preprint]. *arXiv*. https://doi.org/10.48550/arXiv.2308.03952
4. Parkhomenko, I. I., & Savonik, M. V. (2025). Preventing AWS infrastructure-as-code misconfiguration threats based on testing. *Cybersecurity: Education, Science, Technique, 1*(29), 236-251.https://doi.org/10.28925/2663-4023.2025.29.887
5. HashiCorp. (2026). *HCP Terraform agents*. https://developer.hashicorp.com/terraform/cloud-docs/agents
6. He, Y., Wang, E., Rong, Y., Cheng, Z., & Chen, H. (2024). Security of AI agents [Preprint]. *arXiv*. https://doi.org/10.48550/arXiv.2406.08689
7. Greshake, K., Abdelnabi, S., Mishra, S., Endres, C., Holz, T., & Fritz, M. (2023). Not what you've signed up for: Compromising real-world LLM-integrated applications with indirect prompt injection. In *Proceedings of the 16th ACM Workshop on Artificial Intelligence and Security (AISec '23)* (pp. 79-90). ACM. https://doi.org/10.1145/3605764.3623985
8. Debenedetti, E., Zhang, J., Balunović, M., Beurer-Kellner, L., Fischer, M., & Tramèr, F. (2024). AgentDojo: A dynamic environment to evaluate prompt injection attacks and defenses for LLM agents. In *Advances in Neural Information Processing Systems, 37*. https://doi.org/10.48550/arXiv.2406.13352
9. OWASP Foundation. (2025). *OWASP Top 10 for large language model applications*. https://owasp.org/www-project-top-10-for-large-language-model-applications/
10. Lamport, L. (1994). The temporal logic of actions. *ACM Transactions on Programming Languages and Systems, 16*(3), 872-923. https://doi.org/10.1145/177492.177726
11. Ongaro, D., & Ousterhout, J. (2014). In search of an understandable consensus algorithm. In *Proceedings of the 2014 USENIX Annual Technical Conference (USENIX ATC '14)* (pp. 305-319). USENIX Association.
12. Castro, M., & Liskov, B. (1999). Practical Byzantine fault tolerance. In *Proceedings of the Third Symposium on Operating Systems Design and Implementation (OSDI '99)* (pp. 173-186). USENIX Association.
13. MITRE Corporation. (2024). *MITRE ATT&CK*. https://attack.mitre.org/
14. MITRE Corporation. (2025). *Common Weakness Enumeration (CWE)*. https://cwe.mitre.org/
15. FIRST.org. (2023). *Common Vulnerability Scoring System version 4.0: Specification document*. https://www.first.org/cvss/v4.0/specification-document
16. National Institute of Standards and Technology. (2024). *The NIST Cybersecurity Framework (CSF) 2.0* (NIST CSWP 29). https://doi.org/10.6028/NIST.CSWP.29
17. MITRE Corporation. (2024). *MITRE ATLAS (Adversarial Threat Landscape for Artificial Intelligence Systems)*. https://atlas.mitre.org/

КІБЕРБЕЗПЕКА: освіта, наука, техніка
CYBERSECURITY: EDUCATION, SCIENCE, TECHNIQUE
№ 2 (34), 2026
ISSN 2663 – 4023

Mykhailo Savonik
PhD student at the Department of Cybersecurity and Information Protection
Faculty of Information Technology, Taras Shevchenko National University of Kyiv, Kyiv, Ukraine
ORCID:0009-0001-7622-978X
savonikm@fit.knu.ua

Ivan Parkhomenko
PhD (Candidate of Technical Sciences), Associate Professor, Head of the Department of Cybersecurity and Information Protection
Faculty of Information Technology, Taras Shevchenko National University of Kyiv, Kyiv, Ukraine
ORCID:0000-0001-6889-9284
ivan.parkhomenko@knu.ua

ADVERSARIAL RESILIENCE OF AGENTIC INFRASTRUCTURE-AS-CODE SYSTEMS: THREAT MODEL, ATTACK TAXONOMY, AND MULTI-LAYERED DEFENSE ARCHITECTURE

Abstract. The integration of large language models (LLMs) and agentic components into Infrastructure as Code (IaC) workflows significantly expands the attack surface of cloud automation environments. This paper proposes a threat model for agentic IaC systems in the form of the structure S_IaC = (Agents, Environment, Configurations, Outcomes), which makes it possible to identify the critical points of influence: agent input data, configuration state, agent intents and goals, inter-agent communication channels, the execution process, authorization mechanisms, and system availability. Based on this model, a taxonomy of 40 attack vectors grouped into 7 categories is developed: prompt injection, configuration state poisoning, agent intent manipulation, attacks on inter-agent communication, exploitation of execution failures, attacks on authorization and access control, and attacks on integrity and availability. A multi-layered (compositional) defense architecture is proposed that combines input validation, agent intent verification, coordination and consensus mechanisms, and self-healing capabilities. The research methodology comprises three stages: threat modeling based on literature synthesis and a preliminary expert review (a formal Delphi study with an external panel is left to future work); design of the compositional defense architecture; and empirical validation in a reproducible software package on 2,157 scenarios – 1,441 synthetic domain-specific scenarios covering all 7 taxonomy categories (each represented by more than 100 scenarios) and 716 scenarios from public datasets (deepset/prompt-injections and labelled Terraform fixtures from the KICS project, both Apache-2.0) – under 1-, 3-, and 5-agent cluster configurations. The defense mechanisms are implemented as deterministic checks (rules, policies, cryptographic integrity), ensuring full reproducibility. The aggregate detection rate on the synthetic scenarios is 79.9% (95% CI 77.3-82.3%) with a false positive rate of 0.7%; effectiveness differs across categories, with the lowest results for agent intent manipulation (A3, 64.7%) and inter-agent communication (M4, 66.7%). On an independent real-world injection set (deepset) the domain-tuned rules detect only 14.4% of attacks, and non-English injections are not detected at all, which demonstrates the limited cross-domain transferability of static rules. Conversely, in a live LLM-agent loop (AgentDojo) on a domain-specific IaC benchmark the defense reduces the attack success rate from 50.0% to 16.7% and raises utility under attack from 66.7% to 91.7%, confirming its effectiveness within the target domain. All 40 attack vectors are additionally mapped to the MITRE ATT&CK, MITRE ATLAS, CWE, CVSS 4.0, NIST CSF 2.0, and OWASP frameworks; ATT&CK and ATLAS prove complementary (92.5% combined coverage), as the LLM-agent-specific vectors are covered by ATLAS. We formally prove composition properties (detection monotonicity, a union bound on false positives) and consensus correctness restoration under up to $f<n/2$ compromised agents. Future work includes a fully machine-checked verification of these properties and expansion of the attack scenario set.

Keywords: infrastructure as code; infrastructure as tested code; agentic systems; threat model; attack taxonomy; multi-layered defense; Terraform; cloud security; multi-agent systems.

REFERENCES (TRANSLATED AND TRANSLITERATED)

1. Rahman, A., Parnin, C., & Williams, L. (2019). The seven sins: Security smells in infrastructure as code scripts. In *Proceedings of the 41st International Conference on Software Engineering (ICSE '19)* (pp. 164-175). IEEE. https://doi.org/10.1109/ICSE.2019.00033
2. Rahman, A., Mahdavi-Hezaveh, R., & Williams, L. (2019). A systematic mapping study of infrastructure as code research. *Information and Software Technology, 108*, 65-77.https://doi.org/10.1016/j.infsof.2018.12.004
3. Verdet, A., Hamdaqa, M., Da Silva, L., & Khomh, F. (2023). Exploring security practices of infrastructure as code: An empirical study [Preprint]. *arXiv*. https://doi.org/10.48550/arXiv.2308.03952
4. Parkhomenko, I. I., & Savonik, M. V. (2025). Preventing AWS infrastructure-as-code misconfiguration threats based on testing. *Cybersecurity: Education, Science, Technique, 1*(29), 236-251.https://doi.org/10.28925/2663-4023.2025.29.887
5. HashiCorp. (2026). *HCP Terraform agents*. https://developer.hashicorp.com/terraform/cloud-docs/agents
6. He, Y., Wang, E., Rong, Y., Cheng, Z., & Chen, H. (2024). Security of AI agents [Preprint]. *arXiv*. https://doi.org/10.48550/arXiv.2406.08689
7. Greshake, K., Abdelnabi, S., Mishra, S., Endres, C., Holz, T., & Fritz, M. (2023). Not what you've signed up for: Compromising real-world LLM-integrated applications with indirect prompt injection. In *Proceedings of the 16th ACM Workshop on Artificial Intelligence and Security (AISec '23)* (pp. 79-90). ACM. https://doi.org/10.1145/3605764.3623985
8. Debenedetti, E., Zhang, J., Balunović, M., Beurer-Kellner, L., Fischer, M., & Tramèr, F. (2024). AgentDojo: A dynamic environment to evaluate prompt injection attacks and defenses for LLM agents. In *Advances in Neural Information Processing Systems, 37*. https://doi.org/10.48550/arXiv.2406.13352
9. OWASP Foundation. (2025). *OWASP Top 10 for large language model applications*. https://owasp.org/www-project-top-10-for-large-language-model-applications/
10. Lamport, L. (1994). The temporal logic of actions. *ACM Transactions on Programming Languages and Systems, 16*(3), 872-923. https://doi.org/10.1145/177492.177726
11. Ongaro, D., & Ousterhout, J. (2014). In search of an understandable consensus algorithm. In *Proceedings of the 2014 USENIX Annual Technical Conference (USENIX ATC '14)* (pp. 305-319). USENIX Association.
12. Castro, M., & Liskov, B. (1999). Practical Byzantine fault tolerance. In *Proceedings of the Third Symposium on Operating Systems Design and Implementation (OSDI '99)* (pp. 173-186). USENIX Association.
13. MITRE Corporation. (2024). *MITRE ATT&CK*. https://attack.mitre.org/
14. MITRE Corporation. (2025). *Common Weakness Enumeration (CWE)*. https://cwe.mitre.org/
15. FIRST.org. (2023). *Common Vulnerability Scoring System version 4.0: Specification document*. https://www.first.org/cvss/v4.0/specification-document
16. National Institute of Standards and Technology. (2024). *The NIST Cybersecurity Framework (CSF) 2.0* (NIST CSWP 29). https://doi.org/10.6028/NIST.CSWP.29
17. MITRE Corporation. (2024). *MITRE ATLAS (Adversarial Threat Landscape for Artificial Intelligence Systems)*. https://atlas.mitre.org/

Отримано редакцією журналу / Received: 18.01.26
Прорецензовано / Revised: 02.02.26
Схвалено до друку / Accepted: 24.09.26