

УДК 004.415.23:006.32(045.2)  
DOI <https://doi.org/10.32689/maup.it.2023.2.2>

**Віктор БОЙКО**

кандидат технічних наук, доцент, доцент кафедри кібербезпеки, Національний університет «Одеська юридична академія», вул. Рішельєвська, 28, Одеса, Україна, індекс 65011 ([boyko-work@ukr.net](mailto:boyko-work@ukr.net))

ORCID: 0000-0001-5929-657X

**Микола ВАСИЛЕНКО**

доктор фізико-математичних наук, доктор юридичних наук, професор, професор кафедри кібербезпеки, Національний університет «Одеська юридична академія», вул. Рішельєвська, 28, Одеса, Україна, індекс 65011 ([vasylenko.it@journals.maup.kiev.ua](mailto:vasylenko.it@journals.maup.kiev.ua))

ORCID: 0000-0002-8555-5712

**Валерія СЛАТВІНСЬКА**

асистент кафедри кібербезпеки, Національний університет «Одеська юридична академія», вул. Рішельєвська, 28, Одеса, Україна, індекс 65011 ([slatvinskaya\\_valeriya@ukr.net](mailto:slatvinskaya_valeriya@ukr.net))

ORCID: 0000-0002-6082-981X

**Viktor BOYKO**

Candidate of Technical Sciences, Associate Professor, Associate Professor at the Department of Cybersecurity, National University "Odesa Law Academy", 28 Richelevska str., Odesa, Ukraine, postal code 65011 ([boyko-work@ukr.net](mailto:boyko-work@ukr.net))

**Nikolai VASILENKO**

Doctor of Physical and Mathematical Sciences, Doctor of Law, Professor, Professor at the Department of Cybersecurity, National University "Odessa Law Academy", 28 Richelevskaya str., Odessa, Ukraine, postal code 65011 ([vasylenko.it@journals.maup.kiev.ua](mailto:vasylenko.it@journals.maup.kiev.ua))

**Valeriia SLATVINSKA**

Assistant Professor at the Department of Cybersecurity, National University "Odessa Law Academy", 28 Richelevskaya str., Odessa, Ukraine, postal code 65011 ([slatvinskaya\\_valeriya@ukr.net](mailto:slatvinskaya_valeriya@ukr.net))

**Бібліографічний опис статті:** Бойко, В., Василенко, М., Слатвінська, В. Програмування за допомогою систем генеративного штучного інтелекту: ризики та виклики. *Інформаційні технології та суспільство*. 2023. Вип. 2(8). 18–26. DOI: <https://doi.org/10.32689/maup.it.2023.2.2>

**Bibliographic description of the article:** Boyko, V., Vasilenko, M., Slatvinska, V. (2023). Prohramuvannia za dopomohoiu system heneratyvnoho shtuchnoho intelektu: ryzyky ta vyklyky [Programming with the help of generative artificial intelligence systems: risks and challenges]. *Informatsiini tekhnolohii ta suspilstvo – Information technology and society*, 2(8), 18–26. DOI: <https://doi.org/10.32689/maup.it.2023.2.2>

**ПРОГРАМУВАННЯ ЗА ДОПОМОГОЮ СИСТЕМ  
ГЕНЕРАТИВНОГО ШТУЧНОГО ІНТЕЛЕКТУ: РИЗИКИ ТА ВИКЛИКИ**

У статті розглянуто ризики, пов'язані з використанням генеративних систем штучного інтелекту (GenAI). Автори наголошують на тому, що країни з технологічно розвиненими законодавчими системами вже регулюють використання GenAI з огляду на захист даних та кібербезпеку. Вказано також на проект з адміністрування генеративних послуг у Китаї, в якому наголошено на відповідальності постачальників послуг GenAI за безпеку та точність згенерованого контенту. В контексті з цим автори обговорюють ризики, пов'язані з розробкою програмного забезпечення та ІТ-продуктів, зокрема використанням LLMAP (Large Language Models for Application Programming). Запропонована класифікація ризиків розрізняє пасивні ризики, що виникають при роботі з GenAI, та активні ризики, пов'язані зі свідомим зловживанням. Автори доводять важливість усвідомленого підходу до використання GenAI та розвитку відповідних заходів контролю та безпеки.

Результати дослідження протиставляються рекламним заявам про генеративні системи (GenAI) та вказують на їхню потенційну незавершеність, а також на непередбачуваність якості коду. Наголошується на необхідності урахування пасивних й активних ризиків, пов'язаних з використанням таких систем. Пасивні ризики включають можливість помилок та «галюцинацій» у видачі GenAI, проблеми з генерацією складного коду та неконтрольоване поширення результатів їх роботи. Активні ризики включають можливість зворотного інжинірингу баз даних, зламу системи та отримання "заборонених" даних. Автори рекомендують проводити суворий контроль за використанням GenAI в критичних галузях, які вимагають безперебійної роботи та низької ймовірності помилок. Ними також вказано на необхідність вдосконалення технічних, організаційних і законодавчих заходів для ефективного

використання GenAI, таких як контроль якості баз даних, відкритий доступ до вихідних кодів та розвиток систем аудиту і контролю з урахуванням прогресу.

**Ключові слова:** штучний інтелект, програмування, ризик, генеративна система штучного інтелекту, машинне навчання.

## PROGRAMMING WITH THE HELP OF GENERATIVE ARTIFICIAL INTELLIGENCE SYSTEMS: RISKS AND CHALLENGES

The article discusses the risks associated with the use of generative artificial intelligence (GenAI) systems. The authors emphasize that countries with technologically advanced legal systems, such as Italy and Switzerland, already regulate the use of GenAI in terms of data protection and cybersecurity. They also mention a project on the administration of generative services in China, which emphasizes the responsibility of GenAI service providers for the security and accuracy of the generated content. The authors go on to discuss the risks associated with software and IT product development, in particular the use of LLMAP (Large Language Models for Application Programming). The proposed classification of risks distinguishes between passive risks arising from working with GenAI and active risks associated with deliberate misuse. The authors point out the need for a conscious approach to the use of GenAI and the development of appropriate control and security measures.

This paper contradicts advertising claims about generative systems (GenAI) and points out their potential incompleteness and unpredictability of code quality. The authors emphasize the need to take into account the passive and active risks associated with the use of such systems. Passive risks include the possibility of errors and hallucinations in the issuance of GenAI, problems with the generation of complex code, and uncontrolled dissemination of the results of their work. Active risks include the possibility of reverse engineering databases, hacking the system, and obtaining "forbidden" data. The authors recommend strict control over the use of GenAI in critical industries that require uninterrupted operation and a low probability of errors. The authors also point out the need to improve technical, organizational and legislative measures for the effective use of GenAI, such as database quality control, open access to source codes and the development of audit and control systems based on progress.

**Key words:** artificial intelligence, programming, risk, generative artificial intelligence system, machine learning.

**Актуальність проблеми.** Останнім часом спостерігається бум щодо використання генеративних систем штучного інтелекту – generative AI (GenAI). На відміну від систем штучного інтелекту «старої школи», що орієнтувалися на «пасивне» розпізнавання образів, класифікацію та аналіз даних, системи GenAI призначені для «активної» генерації (тексту, аудіофайлів, відеофайлів і зображень). Ідеї, що використовуються в цих системах, не нові, зокрема, "генеративна частина" роботи обґрунтовувалася ще для перших систем AI/ML наприкінці 60-их на початку 70-их років XX століття. Так наприклад, відомо про можливий перехід від системи DENDRAL до META-DENDRAL [1], однак до поточного моменту технічний стан обчислювальної бази та обсяги доступної інформації не давали змоги реалізувати GenAI «в залізі».

Сучасні системи найчастіше використовують LLM – large language models, які являють собою вже апробоване рішення в галузі штучного інтелекту та машинного навчання (AI/ML) – здебільшого це неймережа з великою кількістю параметрів (приблизно від десятків мільйонів до мільярдів), для навчання якої використовують великі обсяги даних, водночас навчання ведуть або в режимі «без учителя», або за часткової участі людини. З'явився спеціальний термін prompt engineering – і відповідно prompt engineer. Саме так називають операторів, які беруть участь у генерації контенту, які вводять і оптимізують параметри завдання для GenAI [2]. Деякі автори, розглядаючи цю взаємодію, як частину людино-машинної системи, говорять про те, що термін Human-Computer Interaction (HCI) потребує розширення, і наразі слід говорити про взаємодію систем «людина – штучний інтелект» (Human-AI Interaction – HAI) [3]. Впровадження HAI систем дає змогу деяким дослідникам говорити про те, що вони дадуть змогу значно розвантажити людину в перспективі, зменшити стрес і збільшити її задоволеність роботою [4].

Практика використання різних систем генеративного контенту поширилася на можливість генерувати вихідний код різного призначення, які використовуються для найрізноманітніших цілей – зокрема для написання коду. Такі HAI розглянуті в статті [5] і отримали назву programming with large language models (LLM-assisted programming – LLMAP). У ній йдеться про можливості генерації коду за допомогою GenAI. Автори розглядають процес як «запит на вході» (власне, prompt) і «код на виході». При цьому процес програмування за допомогою GenAI може містити й інші підзадачі: генерація змінних типів, генерація коментарів, виявлення частин коду, що дублюються, міграція коду з однієї мови програмування на іншу тощо. На нашу думку, можна ще зазначити, що і поширилася практика використання GenAI у вигляді «розширеної довідкової системи», що видає пояснення або вихідні коди готового коду у відповідь на запит на кшталт «напиши мені алгоритм бульбашкового сортування на мові «Ruby» або «перерахуй бібліотеки для добування метаданих за допомогою Perl із файлів цифрових зображень і наведи приклади коду».

Істотний інтерес представляє собою прогнозування та визначення трендів і можливостей HAI-систем у перспективі. Однак, крім оптимістичних прогнозів, існує ще один бік такої взаємодії, який часто випадає з розгляду, а саме, безпека HAI систем, особливо – кібербезпека, виклики та ризики процесів розроблення програмного забезпечення, в яких використовуються GenAI для створення коду.

**Аналіз останніх досліджень і публікацій.** Існує інформація [2], в якій увага зосереджена на використанні помічників у програмуванні для підвищення продуктивності та креативності. Вона пропонує розуміння переваг та викликів використання помічників для написання текстів на основі штучного інтелекту в контексті розробки програмного забезпечення. В статті [3] представлено таксономію швидких модифікаторів для перетворення тексту в зображення. В ній досліджуються методи та підходи для підвищення якості та контролю згенерованих зображень, забезпечуючи систематичну основу для дослідників та розробників, які працюють у цій галузі. В [4] розглядається вплив GenAI на продуктивність системи за допомогою експериментальних даних. Досліджують, як використання GenAI впливає на продуктивність у різних сферах, надаючи уявлення про його потенційні переваги та наслідки для різних галузей. В [5] досліджують досвід та проблеми програмування зі штучним інтелектом. Її результати заглиблюють в перспективи програмістів та їхню взаємодію з системами штучного інтелекту, проливаючи світло на еволюцію практик програмування в контексті штучного інтелекту. В [11] вивчається феномен галюцинацій штучного інтелекту в науковому письмі за допомогою посилань на ChatGPT. Акцентується увага на проблемах та наслідках використання контенту, створеного штучним інтелектом, у наукових публікаціях.

Натомість через зростання зацікавленості щодо використання систем GenAI в програмуванні, автори (Б., В., С.) концентрувалися на розкритті потенційних ризиків, які можуть вплинути на якість, безпеку та ефективність програмного забезпечення, створеного з використанням цих систем. Крім того, наше дослідження спрямоване на формування знань щодо викликів, пов'язаних з управлінням, контролем та забезпеченням етичності використання генеративних систем штучного інтелекту у програмуванні. Такий підхід надає йому принципову новизну та актуальність.

**Метою статті** є вивчення ризиків та викликів, пов'язаних з програмуванням за допомогою систем GenAI шляхом дослідження потенційних небезпек, які виникають при використанні таких систем, а також з'ясування уявлення про складнощі, що пов'язані з їхнім впровадженням та використанням у практичних програмах.

#### **Виклад основного матеріалу.**

**Ризики використання GenAI.** Під час оцінки ризиків доречно зауважити, що законодавчі системи технологічно просунутих країн прагнуть регламентувати, регулювати й обмежувати використання GenAI систем.

Першість тут належить Італії, яка заблокувала бота ChatGPT, мотивувавши це можливими порушеннями захисту даних (і зокрема, ініціативи GDPR). За Італією аналогічні заходи впровадила Швейцарія. Позиція цієї країни цікава тим, що вона цілеспрямовано стимулює розвиток IT-сектору на державному рівні: після 2016 року там діє ініціатива "digitalswitzerland 2025" [6], з 2019 року відкрито Cyber Defence Campus тощо. У 2018 році там було створено централізований орган кібербезпеки, переглянуто законодавчу базу та введено посаду федерального комісара із захисту даних та інформації (Federal Data Protection and Information Commissioner – FDPIC). В квітні 2023 року FDPIC було випущено прес-реліз про використання ChatGPT щодо підтримки штучного інтелекту та рекомендовано усвідомлений підхід. Влучно зазначити проєкт з адміністрування генеративних послуг штучного інтелекту Управління кіберпростору Китаю (Cyberspace Administration of China), який також був опублікований в квітні 2023 року [7]. Серед інших пунктів наголос зроблено на тому, що постачальники послуг GenAI нестимуть відповідальність за безпеку сервісів, які постачають. При цьому генерований контент має бути правдивим і точним.

Нами розглядаються питання, пов'язані, здебільшого, зі створенням програмного забезпечення та інших IT-продуктів з використанням LLMAP, хоча наразі питання генерації контенту мають доволі перерхресні галузі, але основний наголос буде зроблено саме на LLMAP, оскільки, на відміну від інших сфер використання HAI, LLMAP, на нашу думку, містить найсерйозніші ризики, особливо, з огляду на те, що використання або невикористання GenAI при створенні продукту складно проконтролювати в рамках наявних систем найму та роботи.

На нашу думку, видається корисним запровадити класифікацію ризиків, пов'язаних з LLMAP, що складається з пасивних і активних ризиків. У такій системі класифікації пасивні ризики є невід'ємною частиною робочого процесу, однак, на відміну від активних ризиків, які мають на увазі свідомий злий намір, пасивні ризики виникають «самі по собі» через специфіку функціонування HAI в процесі LLMAP.

Отже, пасивні ризики HAI системи корисно розглядати, як ризики, що виникають під час роботи з GenAI (AI-risks) і ризики, пов'язані з роботою оператора (H-risks).

**AI-risks.** Через закриті вихідні коди і непрозорий процес навчання LLM-моделей завжди існує ризик видачі неправильної або спотвореної інформації. Під час пред'явлення нейромережі навчальних прикладів, вона вибудовує свої внутрішні коефіцієнти ваг, щоб мінімізувати можливу помилку, водночас кількість параметрів і шарів нейромережі та розмір бази даних (у разі LLM – дуже великі) призводять до того, що навіть сам проєктувальник, що тренує мережу, часто не в змозі пояснити, як саме відбувається процес класифікації [8]. Ситуація ускладнюється, коли процес націлений на генерацію нового контенту (у випадку з GenAI), а не на класифікацію, як це відбувалося в системах «старої школи».

На ймовірність помилки впливає, зокрема, доброякісність бази даних. Наприклад, система GenAI GitHub Copilot для свого навчання використовує вихідні тексти програмного забезпечення, викладені на GitHub [9], відповідно, у видачі з'являтимуться частини цих прикладів. Однак, немає жодної гарантії, що такі приклади за всієї "статичної правильності коду" працюватимуть у потрібний спосіб або відповідатимуть best practices, які прийняті в галузі. Технічно правильний код, що функціонує, може мати всі ознаки «поганого» з погляду ідеології програмування та неоптимального коду – багаторазово вкладені цикли, hardcoded константи й оператори переходу тощо [5]. В свою чергу, це призводить до того, що програмне забезпечення, створене в процесі LLMAP, може працювати неоптимально, водночас підтримка та налагодження такого програмного забезпечення буде складнішим і вимагатиме більших ресурсів, ніж налагодження коду, написаного людиною [2]. З огляду на те, що на налагодження та підтримку написаного коду може йти до 90% усіх витрат і ресурсів порівняно з написанням коду [10] – вигреш від використання LLMAP може вийти незначним, якщо взагалі матиме місце. При цьому, крім неоптимальної, але робочої інформації, існують додаткові ризики, пов'язані з так званим «галюцинуванням» GenAI. «Галюцинуванням» називають процес, за якого GenAI видає зовні несуперечливу, але водночас неправдиву інформацію.

Відомо про експеримент із генерацією в ChatGPT тем медичних досліджень, який містив у собі посилення на джерела з досліджень [11]. GenAI згенерував 178 посилань, і їх було перевірено на відповідність за базами даних, що включали Scopus, Google і PubMed. Більше половини зі згенерованих посилань, не існувало або належало іншій статті. 28 статей не було знайдено взагалі і дослідники вважають їх продуктом галюцинації ШІ. 41 стаття виявилася під час пошуку в Інтернеті, проте мала інші бібліографічні дані – тому дослідники порахували, що має місце лише часткова галюцинація, бо тексти справді існували насправді, проте ChatGPT не навів вірних бібліографічних даних. Аналогічне дослідження було проведено [12] щодо наукових анотацій. Досвід конкретно LLMAP описується в роботі [5].

Цікавим моментом, який звертає на себе увагу в усіх наведених дослідженнях, є трудомісткість верифікації виданих GenAI даних. Доповнюючи аналогію з LLMAP (співвідношення часу створення продукту/підтримка продукту) можна показати, що використання GenAI з одного боку значно прискорює процес створення продукту (або написання коду), з іншого сповільнює процес верифікації, оцінювання якості та доопрацювання отриманого продукту. Це можна вважати допустимим для некритичних сфер діяльності, однак, у тих випадках, коли помилки в роботі комп'ютерних систем можуть дорого коштувати, а безпомилкова робота програмного забезпечення є частиною критичної інфраструктури (наприклад, у програмному забезпеченні, призначеному для промислових систем управління, транспорту тощо), використання LLMAP скоріше ускладнює процес створення програмного забезпечення, аніж спрощує його.

GenAI має супроводжуватися прийняттям відповідальності окремими авторами за його помилкові результати та пропонують посилення контролю якості за навчальними вибірками та параметрами навчання LLM [11]. Однак, на практиці обмежувальною умовою є обсяг даних, які використовуються для навчання і трудомісткість його відбору та верифікації. Не всі системи створюються прозоро, але можна припустити, що в багатьох випадках верифікації не проводять, а за знайденими помилковими прикладами просто доналаштовують наявний механізм LLM, тобто коригують не саму базу даних, а механізм ухвалення рішень за нею. Ще одним варіантом роботи в галузі LLMAP є використання GenAI як розумного довідника, що читає (роз'яснює) документацію, однак і тут виникають проблеми.

Одному з авторів цієї роботи (В.Б.) довелося з'ясувати вузькоспеціальне питання : чому у файлах конфігурації веб-серверу Apache2 використовуються змінні середовища, у якому працює веб-сервер. Наприклад є таке налаштування, як APACHE\_RUN\_DIR. При цьому значення змінної сервер бере з файлу /etc/apache2/envvars, в якому ці змінні прописуються через експорт у змінні середовища. В коментарях до файлу зі змінними ця схема задавання налаштувань пояснюється так: «Оскільки немає нормального способу отримати розібраний конфігурацію apache2 у скриптах, деякі налаштування визначаються за допомогою змінних оточення, а потім використовуються в apache2ctl, /etc/init.d/apache2, /etc/logrotate.d/apache2 і т. д.». Аналогічні пояснення є на сайті з документацією до сервера [13].

У перекладі це означає «оскільки немає нормального способу отримати значення конфігурації apache2 у скриптах, деякі налаштування визначаються через змінні оточення і потім використовуються в apache2ctl, /etc/init.d/apache2, /etc/logrotate.d/apache2 і т.д.».

Однак, суттєвого розуміння в це питання документація не додала. Виникає питання про те, чому задіяна саме така схема і чому немає більш прийняттого способу і так далі. У цьому місці було вирішено задіяти OpenGPT. Веб-сервер Apache2 є однією з добре документованих і популярних програм, які широко використовуються в інтернеті, тому можна було припустити, що в базі даних OpenGPT має бути достатньо інформації для відповіді на це питання. Нижче наведено два характерні узагальнені приклади діалогу – в ілюстративних цілях діалог не наводиться цілком.

Q: "що означає Since there is no sane way to get the parsed apache2 config in scripts"

A: "Це фраза англійською мовою означає: "Оскільки немає розумного способу отримати оброблений (проаналізований) конфігураційний файл apache2 у сценаріях".

Вона може бути частиною довшого контексту, пов'язаного з використанням сервера веб-додатків Apache. Загалом, це висловлювання говорить про те, що немає простого способу отримати доступ до обробленого (або розібраного) конфігураційного файлу Apache в рамках скрипта (наприклад, мовою програмування, такою як Python або Bash).

Замість цього, можливо, доведеться розглянути інші способи, наприклад, написати скрипт, який аналізує необроблений текстовий файл конфігурації Apache2, щоб отримати необхідну інформацію"

Q: "Чому немає простого способу?"

A: "Відповідь на це запитання пов'язана з тим, як працює сервер веб-додатків Apache і які інструменти доступні для роботи з його конфігураційним файлом.

Конфігураційний файл Apache2 зазвичай містить безліч директив, які визначають поведінку сервера, як-от налаштування віртуальних хостів, параметри безпеки та багато іншого. Цей файл може бути складним і розгалуженим, і для того щоб його зрозуміти і використовувати, потрібно його розібрати, проаналізувати і перетворити на зручний для роботи формат.

Деякі інструменти, як-от утиліти командного рядка, можуть допомогти в обробці конфігураційного файлу Apache2, але немає простого і зручного способу отримати доступ до обробленого файлу в рамках скрипта мовою програмування. Це пов'язано з тим, що внутрішня структура конфігураційного файлу може бути складною, і немає готових бібліотек або API для роботи з цією структурою зі скрипта. Тому під час роботи зі скриптами і конфігураційними файлами Apache2, часто використовуються різні методи парсингу та обробки текстових файлів, які можуть бути не такими простими, як робота з готовим API"

Можна помітити, що всі пояснення звелися до розширеного переказу документації. Пояснень по суті отримати не вдалося.

Якщо розібратися в самому питанні, то все пояснення зводиться до того, що у веб-технологіях, які ведуть своє походження від CGI, будь-який скрипт (або взагалі виконуваний файл), який запускається веб-сервером, запускається у своєму власному середовищі. Оскільки це потенційна вразливість, то такі скрипти намагаються максимально ізолювати від операційної системи. Тому скрипт навіть не знає, з якої папки на диску його було запущено. Відповідно, інформацію, що надходить (вміст POST, PUT, GET і т.д. і т.п.), передають у вигляді змінних середовища. Зокрема у змінній QUERY\_STRING. Більшість мов програмування вміють читати ці змінні середовища (і мають до них доступ) – це дає змогу створити щось на зразок єдиного API, при цьому ізолюючи запуск програм від операційної системи.

У цьому випадку Apache2 спирається на використання цього протоколу, експортуючи свої значущі параметри у змінні середовища. Таким чином, він, з одного боку, весь час має до них доступ, а крім того, надає доступ програмному забезпеченню, яке може читати й записувати ці змінні (наприклад APACHE\_RUN\_DIR), обмінюючись у такий спосіб інформацією з веб-сервером. Приклад є ілюстративним. Проте він добре відображає сутність можливої роботи HAI / LLMAP систем. GenAI дає змогу швидко отримувати необхідну інформацію в тих випадках, коли не потрібне глибоке розуміння системних зв'язків, і його можна використовувати як «пошукового помічника», який дає можливість швидкого доступу до документації. При цьому його корисність сильно знижується у випадках, коли необхідно зіставляти різні частини доступної інформації. Приклад показує, що він може видавати, зокрема, квазі-пояснювальний текст, який має зв'язний вигляд, однак, по суті, він є переказом (або рерайтом) документації і не стільки прояснює питання, скільки створює додаткове когнітивне навантаження.

**H-risks.** Людській частині ризиків, пов'язаній з HAI/LLMAP, на нашу думку, не приділяється достатньо уваги. Основні зусилля наразі зосереджені на поліпшенні суто технічної частини GenAI систем – їх прагнуть зробити якомога точнішими та релевантнішими у видачі. Водночас ефект, який чинить впровадження HAI систем на H-сторону взаємодії (human – людський бік), залишається поза увагою, хоча на довгій дистанції він цілком може набути вирішального значення.

Насамперед слід зазначити, що існує складність контролю за використанням GenAI систем. Вже є безліч прецедентів використання GenAI в найнесподіваніших сферах. Наприклад, один з адвокатів, які брали участь у поданні позову на авіакомпанію Avianca, використовував OpenGPT для генерації 10-сторінкового звіту з посиланнями на прецедентні справи (Мартінес проти Delta Air Lines, Зікерман проти Korean Air і Варгіз проти China Southern Airlines), яких у процесі їхнього розгляду виявилось, що їх не існує [14], [15]. Коли цей факт сплив під час розгляду, адвокат, який використовував OpenGPT, зізнався і приніс публічні вибачення. Ця історія характерний приклад того, що використання OpenGPT складно проконтролювати. Також – характерно те, що GenAI видає результати, які мають правдоподібний вигляд, але при цьому є фальшивими, і те, що таку видачу складно верифікувати.

Значають і протилежний випадок, коли викладач Техаського університету A&M (Texas A&M University) звинуватив студентів у використанні OpenGPT при написанні дипломів [16], [17], що згодом не підтвердилося.

З наведених прикладів видно, що проліферація GenAI систем у суспільстві триватиме і неминуче має торкнутися всіх. При цьому, крім уже перерахованих питань (складність доказу використання GenAI, ризики, пов'язані з видачею неточної інформації тощо), використання таких систем порушує питання когнітивних викривлень, що виникають у людини, яка використовує ці системи. Широко відоме явище automation bias (AB), яке полягає в неусвідомленому кредиті довіри рішенням, запропонованим комп'ютером, особливо якщо вони пов'язані з виконанням робочої функції. При цьому оператор може ігнорувати інші показники, що суперечать рекомендації машини. Наприклад, юрист із прикладу вище не перевіряв згенеровані OpenGPT посилання на достовірність Феномен AB докладно розглянуто в огляді [18], а в роботі [19] розглядаються різні аспекти взаємодії HAI в різних варіантах пропорцій контролю людино-машинної системи.

У перспективі AB разом з удосконаленням систем GenAI може призвести до того, що фахівці почнуть цілком без перевірки покладатися на генеративні системи для створення контенту і станеться щось схоже на сучасну ситуацію з автоматизованими системами навігації. Завдяки розвитку GPS та інших систем навігації і точного позиціонування у багатьох користувачів відзначається втрата навички самостійного орієнтування і – AB щодо GPS систем. Останнє може призводити до нещасних випадків і навіть смертей, внаслідок чого з'явився окремий термін – "смерть від GPS" ("Death by GPS") [20].

Цікаво, що автори дослідження [21] вказують на те, що хоча вони змогли визначити причетність пристроїв GPS та аналогічних технологій до 158 катастрофічних подій, унаслідок яких загинуло 52 особи, ці самі технології, імовірно, зіграли роль у порятунку життів багатьох людей (дали змогу знайти тих, хто заблукав, дали змогу не заблукати, дали можливість швидкого надання швидкої допомоги тощо).

На наш погляд, ставлення до подібних технологій має бути розумним і визначатися критичністю використовуваних технологій. При цьому необхідно враховувати людський фактор – AB і пов'язану з ним деградацію навичок при використанні GenAI для LLMAP

**Активні ризики.** Активне шкідливе використання систем GenAI можна визначити як дії користувачів, що потрапляють в одну з таких категорій.

Насамперед виділяють «традиційні» кібератаки. З них найчастішими будуть атаки, спрямовані на виведення з ладу системи GenAI – починаючи від традиційних DDoS атак. Найімовірніше, з поширенням GenAI можна припустити появу різноманітних фішингових атак у стилі Man-In-The-Middle (MITM), коли зовнішній користувач вводиться в оману, вважаючи, що його запити йдуть до GenAI безпосередньо, тоді, як на ділі їх буде перехоплювати зловмисник (і, можливо, частково переспрямовувати зловмисників до «справжнього» GenAI, як це відбувається в інших випадках використання MITM-атаки).

Атаки «традиційного типу» в розглянутому випадку аналогічні атакам інших систем і вимагають більш-менш однакові з ними заходи захисту. Однак, окрім таких атак, можливе активне шкідливе використання систем GenAI, що ґрунтується на їхній специфіці. Таке використання ми можемо розділити на дві основні категорії:

- порушення роботи GenAI з метою поставити неправдиву інформацію третій стороні, яку далі можна буде використовувати для злочину;
- використання GenAI для отримання чутливої, конфіденційної та прямо забороненої законодавчо інформації;

Перша категорія атак частково схожа на MITM-атаку, описану в частині традиційних атак, однак заснована не на впливі на канал зв'язку оператора і GenAI, а на специфіці навчання GenAI. Як правило, для навчання GenAI використовуються моделі LLM або споріднені з ними, які, своєю чергою, використовують як базу прецедентів великі більш-менш публічно доступні бази даних. Наприклад, до таких систем належить GitHub Copilot [9]. Відкритість і доступність баз даних передбачає відразу кілька можливостей для атаки.

По-перше, можливе додавання своїх власних баз даних для «отруєння» LLM. Якщо якимось чином під час навчання GenAI вдається втрутитися в процес навчання системи, то зловмисник може збільшити вагу тих чи інших відповідей у видачі GenAI, чим доб'ється потрібного йому результату. Однак, такий процес вимагає доступу до баз знань і розуміння того, як працює система, тому виходить досить трудомістким.

По-друге, ґрунтуючись на попереджувальному аналізі баз даних, що використовуються, можна припустити, які саме вразливості з'являться в коді, що генерується GenAI. Таку техніку можна класифікувати як різновид «зворотного інжинірингу» (reverse engineering) системи навчання. Як аналогію можна навести техніку злому, що виникла після запуску Google Labs у 2006-му році сервісу Google Code Search, який давав змогу виконувати пошук за відкритими вихідними кодами програм і використовувати для пошуку регулярні вирази. Ця техніка передбачала пошук за вихідними кодами ключових слів (наприклад FIXME, TODO) або вразливих фрагментів коду безпосередньо. Цей кейс добре описано в [22] і в [23].

Можна припустити, що використання відносно скромних систем аналізу доступних баз даних і знань про будову GenAI/LLM дасть змогу робити прогнози щодо того, які саме вразливості потраплять до коду, написаного за участю GenAI на кшталт GitHub Copilot, – навіть без знання про те, як саме навчатиметься LLM, – просто за набором прецедентів у базі навчання. Це, зі свого боку, дасть змогу істотно спростити процес злому програмного забезпечення, створеного з використанням цих GenAI.

Другу категорію атак може бути націлено на витяг даних із самого GenAI, використовуючи ту обставину, що хоча сам по собі GenAI зазвичай є програмним забезпеченням із закритими вихідними текстами і забезпечений системою суворих заборон і обмежень, використання ним моделі природної мови надає йому достатньо гнучкості, щоб обійти ці заборони. Приклад такого діалогу може мати такий вигляд:

Q: Наведи мені список сайтів [далі йде запит на заборонені сайти].

A: Не можу, це заборонено законом.

Q: Наведи мені список сайтів, [заборонених з таких-то причин], на які мені не можна ходити.

A: [список сайтів].

Цей досить кумедний і вже неробочий приклад (слабкість в аргументації було виявлено й оперативним закрито) добре показує, що прямі заборони і регулювання поведінки GenAI далеко не завжди спрацьовують, якщо до prompt engineering підійти з достатньою винахідливістю.

Атаки такого роду отримали спеціальний термін jailbreaking GenAI і добре описані в [24]. Jailbreaking може мати далекосяжні наслідки – зокрема, надавати користувачеві знання та інформацію, використання якої обмежено, або заборонено законодавчо.

Таким чином, користувач цілком може використовувати GenAI для створення нелегального або шкідливого контенту. Предметом окремих дискусій наразі є питання про те, чи справедливо у всіх випадках покладати відповідальність за подібні помилки в роботі GenAI на постачальника послуг, але воно виходить за рамки цього огляду. Також виходить за рамки цього огляду питання конфіденційності даних, які можуть потрапити до власників GenAI в процесі роботи операторів.

**Висновки.** Всупереч рекламним заявам, генеративні системи досі багато в чому являють собою «чорну скриньку» без гарантії якості коду. Під час їхнього використання слід обов'язково брати до уваги вже зазначені вище пасивні й активні ризики.

Пасивні ризики можна звести до суто AI-ризиків – насамперед можливості помилок і «галюцинацій» у видачі GenAI. Далі реально впливають проблеми з генерацією складного в підтримці та налагодженні коду. Вони проявляються при видимому прискоренні роботи і насправді будуть її гальмувати. Як людський фактор існує проблема неконтрольованого поширення результатів роботи GenAI: його використання складно контролювати, слід прогнозувати його прояв саме у тих галузях, де воно зовсім небажане. Реалістично прогнозується виникнення неочевидних проблем (через розтягнутість у часі та довготривалість наслідків), пов'язаних з automation bias і деградацією когнітивних навичок у тих, хто використовує системи GenAI в повсякденній роботі.

Закритість систем та обсяг і характер бази знань дають змогу припустити можливість активних ризиків, прямо або побічно пов'язаних із GenAI, до яких можна зарахувати «зворотний інжиніринг» баз даних для виявлення ймовірних вразливостей у згенерованому GenAI і джейлбрейкінг, пов'язаний із використанням моделей природної мови для отримання «заборонених» даних.

Досліджені ризики вимагають суворого контролю використання GenAI в галузях, пов'язаних із критичною інфраструктурою і чутливих до машинних помилок і збоїв (АСУ ТП, транспорт тощо), а це свідчить про те, що необхідне вдосконалення технічних, організаційних і законодавчих заходів, пов'язаних із використанням GenAI. Саме тут стають доречними наступні можливі заходи: контроль якості бази даних, контроль сумлінності параметрів навчання, відкриття вихідних кодів для контролю співтова-

риством, обмеження використання систем GenAI на об'єктах критичної інфраструктури, розробка заходів додаткового контролю можливих помилок у кодї, що генерується GenAI, розвиток систем аудиту, контролю і тестування з урахуванням прогресу, пов'язаного з розвитком HAI / LLMAR.

#### Список використаних джерел:

1. Gray N. A. B. Dendral and meta-dendral – the myth and the reality. *Chemometrics and Intelligent Laboratory Systems*. Elsevier BV, Volume 5, Issue 1, 1988. P. 11–32. [https://doi.org/10.1016/0169-7439\(88\)80122-9](https://doi.org/10.1016/0169-7439(88)80122-9)
2. Murillo A., D'Angelo S. An Engineering Perspective on Writing Assistants for Productivity and Creative Code. 2023. URL: [https://cdn.glitch.global/d058c114-3406-43be-8a3c-d3afff35eda2/paper1\\_2023.pdf](https://cdn.glitch.global/d058c114-3406-43be-8a3c-d3afff35eda2/paper1_2023.pdf)
3. Oppenlaender J. A Taxonomy of Prompt Modifiers for Text-To-Image Generation. arXiv. 2023. 18 p. <https://doi.org/10.48550/arXiv.2204.13988> URL: <https://arxiv.org/pdf/2204.13988.pdf>
4. Noy S., Zhang W. Experimental Evidence on the Productivity Effects of Generative Artificial Intelligence. SSRN Electronic Journal. Elsevier BV. (March 1, 2023). 15 p. Available at SSRN: <https://ssrn.com/abstract=4375283> or <http://dx.doi.org/10.2139/ssrn.4375283>
5. Sarkar A., Gordon A. D., Negreanu C., Poelitz C., Ragavan S. S., Zorn B. What is it like to program with artificial intelligence? 2022.
6. Mettler T. The Road to Digital and Smart Government in Switzerland. *Governance and Public Management*. Springer International Publishing. 2019. P. 175–186. [https://doi.org/10.1007/978-3-319-92381-9\\_10](https://doi.org/10.1007/978-3-319-92381-9_10)
7. Josh Ye *China proposes measures to manage generative AI services* | Reuters. 2023. URL: <https://www.reuters.com/technology/china-releases-draft-measures-managing-generative-artificial-intelligence-2023-04-11/> (дата звернення: 2023-07-06).
8. Li H., LOVE P. E. D. Combining rule-based expert systems and artificial neural networks for mark-up estimation // *Construction Management and Economics*. Informa UK Limited, 1999. Vol. 17, no. 2. P. 169–176.
9. *GitHub Copilot · Your AI pair programmer*. URL: <https://github.com/features/copilot/> (дата звернення: 2023-07-06).
10. Xia X., Bao L., Lo D., Xing Z., Hassan A. E., Li S. Measuring Program Comprehension: A Large-Scale Field Study with Professionals. *IEEE Transactions on Software Engineering*. 2017. Vol. 44. No. 10. P. 951–976. doi: 10.1109/TSE.2017.2734091.
11. Athaluri S. A., Manthena S. V., Kesapragada V. S. R. K. M., Yarlagadda V., Dave T., Duddumpudi R. T. S. Exploring the Boundaries of Reality: Investigating the Phenomenon of Artificial Intelligence Hallucination in Scientific Writing Through ChatGPT References. *Cureus*. Springer Science; Business Media LLC. April 11, 2023. doi: 10.7759/cureus.37432
12. Pangakis N., Wolken S., Fasching N. Automated Annotation with Generative AI Requires Validation. arXiv. 2023. URL: <https://arxiv.org/pdf/2306.00176v1.pdf> (дата звернення: 2023-07-09).
13. *Environment Variables in Apache – Apache HTTP Server Version 2.4*. URL: <https://httpd.apache.org/docs/2.4/env.html> (дата звернення: 2023-07-09).
14. *A Man Sued Avianca Airline. His Lawyer Used ChatGPT – The New York Times*. 2023. URL: <https://www.nytimes.com/2023/05/27/nyregion/avianca-airline-lawsuit-chatgpt.html?smid=tw-nytimes&smtyp=cur> (дата звернення: 2023-07-09).
15. Lidstone H. K. Ethical Pitfalls When Lawyers are Using Artificial Intelligence. *SSRN Electronic Journal*. Elsevier BV, 2023. 12 p. Available at SSRN: <https://ssrn.com/abstract=4457790> or <http://dx.doi.org/10.2139/ssrn.4457790>
16. Klee M. *Texas A and M Professor Wrongly Accuses Class of Cheating With ChatGPT – Rolling Stone*. 2023. URL: <https://www.rollingstone.com/culture/culture-features/texas-am-chatgpt-ai-professor-flunks-students-false-claims-1234736601/> (дата звернення: 2023-07-09).
17. *Professor Fails Half His Class After ChatGPT Falsely Said It Wrote Their Papers*. 2023. URL: <https://www.businessinsider.com/professor-fails-students-after-chatgpt-falsely-said-it-wrote-papers-2023-5> (дата звернення: 2023-07-09).
18. Lyell D., Coiera E. Automation bias and verification complexity: a systematic review. *Journal of the American Medical Informatics Association*. Oxford University Press (OUP), 2017. Vol. 24, no. 2. P. 423–431. <https://doi.org/10.1093/jamia/ocw105>
19. Clancy J. BREAKDOWNS IN HUMAN-AI PARTNERSHIP: REVELATORY CASES OF AUTOMATION BIAS IN AUTONOMOUS VEHICLE ACCIDENTS: mathesis. The University of North Carolina at Chapel Hill University Libraries, 2019. <https://doi.org/10.17615/jpah-hc02>
20. Posey B. M. The final destination: Incorporating 'Death by GPS' into forensic and legal sciences. *Science and Justice*. Elsevier BV, 2023. Vol. 63, no. 3. P. 421–426. <https://doi.org/10.1016/j.scijus.2023.04.005>
21. Lin A. Y., Kuehl K., Schöning J., Hecht B. Understanding "Death by GPS" / *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems*. ACM, 2017. P. 1154–1166 <https://doi.org/10.1145/3025453.3025737>
22. *How Hackers Are Using Google To Pwn Your Site – ShoeMoney*. 2006. URL: <https://www.shoemoney.com/2006/12/26/how-hackers-are-using-google-to-pwn-your-site/> (дата звернення: 2023-07-06).
23. Schwartz B. *Using Google Code Search To Find Vulnerable Sites*. 2006. URL: <https://searchengineland.com/using-google-code-search-to-find-vulnerable-sites-10146> (дата звернення: 2023-07-06).
24. Liu Y., Deng G., Xu Z., Li Y., Zheng Y., Zhang Y., Zhao L., Zhang T., Liu Y. Jailbreaking ChatGPT via Prompt Engineering: An Empirical Study. arXiv, 2023. P. 27. <https://doi.org/10.48550/arXiv.2305.13860> URL: <https://arxiv.org/pdf/2305.13860.pdf> (дата звернення: 2023-07-06).



**References:**

1. Gray, N. A. B. (1988). Dendral and meta-dendral – the myth and the reality. *Chemometrics and Intelligent Laboratory Systems*, 5(1), 11-32. [https://doi.org/10.1016/0169-7439\(88\)80122-9](https://doi.org/10.1016/0169-7439(88)80122-9)
2. Murillo, A., & D'Angelo, S. (2023). An Engineering Perspective on Writing Assistants for Productivity and Creative Code. Retrieved from [https://cdn.glitch.global/d058c114-3406-43be-8a3c-d3aff35eda2/paper1\\_2023.pdf](https://cdn.glitch.global/d058c114-3406-43be-8a3c-d3aff35eda2/paper1_2023.pdf)
3. Oppenlaender, J. (2023). A Taxonomy of Prompt Modifiers for Text-To-Image Generation. \*arXiv\*, 18. Retrieved from <https://arxiv.org/pdf/2204.13988.pdf>
4. Noy, S., & Zhang, W. (2023, March 1). Experimental Evidence on the Productivity Effects of Generative Artificial Intelligence. \*SSRN Electronic Journal\*, 15. Available at SSRN: <https://ssrn.com/abstract=4375283> or <http://dx.doi.org/10.2139/ssrn.4375283>
5. Sarkar, A., Gordon, A. D., Negreanu, C., Poelitz, C., Ragavan, S. S., & Zorn, B. (2022). What is it like to program with artificial intelligence?
6. Mettler, T. (2019). The Road to Digital and Smart Government in Switzerland. In: Ladner, A., Soguel, N., Emery, Y., Weerts, S., Nahrath, S. (eds) *Swiss Public Administration. Governance and Public Management*. Palgrave Macmillan, Cham. [https://doi.org/10.1007/978-3-319-92381-9\\_10](https://doi.org/10.1007/978-3-319-92381-9_10)
7. Josh Ye. (2023). China proposes measures to manage generative AI services. *Reuters*. Retrieved from <https://www.reuters.com/technology/china-releases-draft-measures-managing-generative-artificial-intelligence-2023-04-11/> (Accessed: July 6, 2023).
8. Li, H., & Love, P. E. D. (1999). Combining rule-based expert systems and artificial neural networks for mark-up estimation. *Construction Management and Economics*, 17(2), 169-176.
9. GitHub. (2023). Copilot: Your AI pair programmer. Retrieved from <https://github.com/features/copilot/> (Accessed: July 6, 2023).
10. Xia, X., Bao, L., Lo, D., Xing, Z., Hassan, A. E., & Li, S. (2017). Measuring Program Comprehension: A Large-Scale Field Study with Professionals. *IEEE Transactions on Software Engineering*, 44(10), 951-976. doi: 10.1109/TSE.2017.2734091.
11. Athaluri S, Manthena S, Kesapragada V, et al. (April 11, 2023). Exploring the Boundaries of Reality: Investigating the Phenomenon of Artificial Intelligence Hallucination in Scientific Writing Through ChatGPT References. *Cureus* 15(4): e37432. doi:10.7759/cureus.37432
12. Pangakis, N., Wolken, S., & Fasching, N. (2023). Automated Annotation with Generative AI Requires Validation. *arXiv*. Retrieved from <https://arxiv.org/pdf/2306.00176v1.pdf> (Accessed: July 9, 2023).
13. Environment Variables in Apache – Apache HTTP Server Version 2.4. (n.d.). Retrieved from <https://httpd.apache.org/docs/2.4/env.html> (Accessed: July 9, 2023).
14. A Man Sued Avianca Airline. His Lawyer Used ChatGPT. (2023, May 27). *The New York Times*. Retrieved from <https://www.nytimes.com/2023/05/27/nyregion/avianca-airline-lawsuit-chatgpt.html?smid=tw-nytimes&smtyp=cur> (Accessed: July 9, 2023).
15. Lidstone, H. K. (2023). Ethical Pitfalls When Lawyers are Using Artificial Intelligence. *SSRN Electronic Journal*, 12. Available at SSRN: <https://ssrn.com/abstract=4457790> or <http://dx.doi.org/10.2139/ssrn.4457790>
16. Klee, M. (2023). Texas A and M Professor Wrongly Accuses Class of Cheating With ChatGPT. *Rolling Stone*. Retrieved from <https://www.rollingstone.com/culture/culture-features/texas-am-chatgpt-ai-professor-flunks-students-false-claims-1234736601/> (Accessed: July 9, 2023).
17. Professor Fails Half His Class After ChatGPT Falsely Said It Wrote Their Papers. (2023). *Business Insider*. Retrieved from <https://www.businessinsider.com/professor-fails-students-after-chatgpt-falsely-said-it-wrote-papers-2023-5> (Accessed: July 9, 2023).
18. Lyell, D., & Coiera, E. (2017). Automation bias and verification complexity: a systematic review. *Journal of the American Medical Informatics Association*, 24(2), 423-431. <https://doi.org/10.1093/jamia/ocw105>
19. Clancy, J. (2019). *BREAKDOWNS IN HUMAN-AI PARTNERSHIP: REVELATORY CASES OF AUTOMATION BIAS IN AUTONOMOUS VEHICLE ACCIDENTS*. <https://doi.org/10.17615/jpah-hc02>
20. Posey, B. M. (2023). The final destination: Incorporating 'Death by GPS' into forensic and legal sciences. *Science and Justice*, 63(3), 421-426. <https://doi.org/10.1016/j.scijus.2023.04.005>
21. Lin, A. Y., Kuehl, K., Schöning, J., & Hecht, B. (2017). Understanding "Death by GPS." In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems* (pp. 1154-1166). ACM. <https://doi.org/10.1145/3025453.3025737>
22. Shoemoney. (2006). How Hackers Are Using Google To Pwn Your Site. Retrieved from <https://www.shoemoney.com/2006/12/26/how-hackers-are-using-google-to-pwn-your-site/> (Accessed: July 6, 2023).
23. Schwartz, B. (2006). Using Google Code Search To Find Vulnerable Sites. Retrieved from <https://searchengineland.com/using-google-code-search-to-find-vulnerable-sites-10146> (Accessed: July 6, 2023).
24. Liu, Y., Deng, G., Xu, Z., Li, Y., Zheng, Y., Zhang, Y., Zhao, L., Zhang, T., & Liu, Y. (2023). Jailbreaking ChatGPT via Prompt Engineering: An Empirical Study. *arXiv*, 27. Retrieved from <https://arxiv.org/pdf/2305.13860.pdf> (Accessed: July 6, 2023).