ISSN 2219-9365

https://doi.org/10.31891/2219-9365-2024-78-32
УДК 004.49

СМІРНОВ Олексій
Хмельницький національний університет
https://orcid.org/0009-0008-2189-0096
a.smirnov3825@gmail.com
КЛЕЙН Олександр
Хмельницький національний університет
https://orcid.org/0000-0002-1896-943X
olexandrkleyn@gmail.com
ГЛУХЕНЬКИЙ Олександр
Хмельницький національний університет
https://orcid.org/0009-0003-1498-3423
goldbergoalexander@gmail.com
ЛУТЮК Лев
Хмельницький національний університет
https://orcid.org/0009-0004-7497-3427
Lev_Lutiuk@ukr.net
СЕМЕНЮК Богдан
Хмельницький національний університет
https://orcid.org/0009-0001-8831-8835
faludore@mail.com

ВИЯВЛЕННЯ ВРАЗЛИВОСТЕЙ В ПРОГРАМНИХ МОДЕЛЯХ АПАРАТНИХ ЗАСОБІВ

В роботі розглянуто проблему виявлення сторонніх знаків в програмних моделях апаратних засобів. Для її вирішення запропонована стратегія, що базується на використанні генетичного алгоритму. Після проведення аналізу було встановлено, що традиційна модель генетичного алгоритму не підходить через її низьку ефективність. Це пов'язано з проблемами складності, які за своєю суттю виникають як при перетворенні гешу в функції, так і при відображенні запиту на цільовий параметр. Зокрема, було проблемно створити задачі гамільтонового завершення в графі. Ця задача, заснована на рішеннях для визначення мінімальної кількості ребер у графі, яку необхідно додати, щоб забезпечити існування гамільтонового циклу. Таким чином, було запропоновано модифікації, внесені в традиційну модель генетичного алгоритму на додаток до нових функцій, операторів тощо, які були використані. Внесені модифікації в традиційну модель генетичного алгоритму дозволили розробити стратегію та підхід до виявлення сторонніх знаків внесених в програмні моделі апаратних засобів.

Для проведення експериментів з встановлення ефективності розробленого підходу було розроблено застосунок, який реалізує генетичний алгоритм з модифікаціями. На вхід цього застосунку подається два файли. Перший файл містить програмну модель апаратного пристрою, а другий файл містить еталонну модель цього ж апаратного пристрою. В базі моделей сторонніх знаків зберігаються типові моделі внесення сторонніх знаків. Загалом розроблений застосунок виступає як класифікатор. В результаті його використання в експериментальних дослідженнях на штучних наборах вхідних моделей, він продемонстрував результат класифікації, який оцінено метрикою F1, що дорівнює 82%. Таке значення є допустим. Для його покращення потрібно наповнювати базу моделей сторонніх знаків.

Напрямами подальших досліджень є удосконалення в підході, що базується на модифікаціях в генетичному алгоритмі. Ці удосконалення першочергово стосуватимуться врахування більшої кількості моделей внесення сторонніх знаків в програмні моделі апаратних засобів.

Ключові слова: апаратне забезпечення, вразливості, генетичний алгоритм.

SMIRNOV Oleksii, KLEIN Olexandr, HLUKHENKYI Oleksandr,
LUTIUK Lev, SEMENIUK Bohdan
Khmelnitskyi National University

DETECTION OF VULNERABILITIES IN SOFTWARE MODELS OF HARDWARE

The work deals with the problem of detecting extraneous signs in software models of hardware. To solve it, a strategy based on the use of a genetic algorithm is proposed. After the analysis, it was found that the traditional genetic algorithm model is not suitable due to its low efficiency. This is due to complexity issues that inherently arise both when converting a hash to a function and when mapping a query to a target parameter. In particular, it was problematic to create Hamiltonian completion problems in the graph. This problem is based on solutions to determine the minimum number of edges in a graph that must be added to ensure the existence of a Hamiltonian cycle. Thus, the modifications made to the traditional genetic algorithm model were proposed in addition to the new functions, operators, etc. that were used. Modifications made to the traditional model of the genetic algorithm made it possible to develop a strategy and approach to the detection of extraneous signs included in software models of hardware.

To conduct experiments to establish the effectiveness of the developed approach, an application was developed that implements a genetic algorithm with modifications. Two files are submitted to the input of this application. The first file contains a

software model of a hardware device, and the second file contains a reference model of the same hardware device. The database of third-party mark models contains typical models of entering third-party marks. In general, the developed application acts as a classifier. As a result of its use in experimental studies on artificial sets of input models, it demonstrated a classification result evaluated by the F1 metric equal to 82%. Such a value is permissible. To improve it, you need to fill the database of models of third-party signs.

Directions for further research are improvements in the approach based on modifications in the genetic algorithm. These improvements will primarily concern the consideration of a greater number of models for entering third-party characters into software models of hardware.

Keywords: hardware, vulnerabilities, genetic algorithm.

ПОСТАНОВКА ПРОБЛЕМИ У ЗАГАЛЬНОМУ ВИГЛЯДІ ТА ЇЇ ЗВ'ЯЗОК ІЗ ВАЖЛИВИМИ НАУКОВИМИ ЧИ ПРАКТИЧНИМИ ЗАВДАННЯМИ

Апаратне забезпечення піддається складним атакам. Не тільки програмна частина комп'ютерної системи зазнає впливів та інфікувань, але і певні складові частини апаратного забезпечення. Виробники апаратних засобів і пристроїв можуть виробляти їх з вкладеними в архітектуру частинами, які потім можуть використовувати із певною метою. В апаратному забезпеченні можуть бути вставки, які можуть бути націлені на певні криптографічні алгоритми. Тому, потребують аналізу та розроблення методи протидії таким атакам із використанням нових стратегій.

Перспективним для розроблення є підхід з використанням генетичного алгоритму, який може бути застосовано для ефективного розв'язання складної задачі зіставлення підграфів, які можуть бути використані для подання архітектури апаратних засобів. Криптографічний алгоритм може бути зламаний за допомогою диференціальної атаки з відкритим текстом зі значно обмеженими ресурсами. Щоб запобігти таким атакам, потрібно враховувати контрзаходи не тільки для криптоалгоритму, але і для всіх мережевих шифрів. Також, крім подання нових примітивів мереж взаємозв'язку, динамічних мереж маршрутизації, потрібно модифікувати елементи, які вимагають прийняття рішень для зловмисника, щоб створити йому проблеми із виконанням зловмисних дій.

Актуальність роботи полягає в розробці методу криптографічного захисту від вразливостей в апаратному забезпеченні на основі генетичного алгоритму (ГА).

Аналіз відомих рішень щодо криптографічного захисту від вразливостей в апаратному забезпеченні

Проектування апаратних засобів здійснюється із застосуванням сучасних інформаційних технологій. По суті таке проектування розробляється програмно. І лише повністю відтестовані і промодельовані програмні моделі передаються на реалізацію з метою їх створення як апаратних пристроїв та засобів.

Автоматизація проектування до певної міри є самовизначальною. Вона включає інструменти та процеси, пов'язані з автоматизацією високорівневої архітектури аж до макета плати, готової до виготовлення. На найвищому рівні як правило розпочинають з графа переходу станів (ГПС) і за допомогою певного інструменту переводять його в апаратну модель (АМ). У зв'язку з цим, АМ може бути оброблена за допомогою таких інструментів, що дозволяє транслювати структури полів для перекладу в примітиви на рівні макета плати, де архітектура може бути спланована, маршрутизована і згенерована готовою до виготовлення як макет плати, враховуючи, що він формально перевірений за допомогою симуляції та тестування або інших програмних засобів. Довіра до такої моделі апаратних пристроїв та засобів на сьогодні може бути нівельована через ризик внесення в неї програмним чином фрагменту, який в подальшому на апаратному рівні надаватиме доступ стороннім особам да цього пристрою чи засобу. До початку активного використання зловмисниками кіберсередовища замовники плат не дуже зосереджувались на такій проблемі. Але в поточний момент часу кіберризики зросли в усіх сферах використання інформаційних технологій і на усіх етапах їх розроблення. Довіра протягом усього процесу створення апаратного пристрою до будь-якої конкретної системи або архітектури була високою, бо компанії припускали, що ліцензовані ядра інтелектуальної власності і готові компоненти за своєю суттю безпечні і не поставлять під загрозу довіру до даного архітектури. І навпаки, не тільки тому, що процес проектування зазнає багато різних форматів і переходів з рук в руки для даної архітектури, але може включати зовнішні, здавалося б, надійні джерела, яких не вистачає загальній довірі.

Довіра до виробництва плат набуває актуальності. Особливо з розвитком систем з IoT, від яких залежатиме надійне функціонування, наприклад, систем в розумному будинку тощо. На сьогодні немає беззастережної довіри до виробничого процесу. Компанії вирішують перевести процес на аутсорсинг, щоб належним чином обробляти макети архітектури або іншу конфіденційну інтелектуальну власність, не зловживаючи і не використовуючи їх поза узгодженими умовами виробництва. На основі таких проблем вже сформувалась стратегія «нульової довіри» [1, 2]. І навпаки, при розгляді цієї стратегії виявляється, що її метою є зміщення парадигми від одноразової перевірки до парадигми постійної [3]. Передумовою цієї ініціативи є як усунення довірених моделей, так і створення довіреного середовища. Тим не менш, за останні кілька місяців спостерігаємо перехід з метою кращого забезпечення виробництва та довіри в

ланцюжку поставок мікросхем до забезпечення загальної зміни парадигми довіри до компонентів специфікацій плат.

Етика або практика виробників не буде ставитись під сумнів. Наприклад, одноразове працевлаштування, яке передає розробки, може поставити під загрозу всю довіру і призвести до того, що проекти стануть сумнівними з точки зору власності та порушення внутрішньої архітектури порівняно з еталонною. Концепція зрізу віддалено нагадує обчислювальні одиниці [4]. Можна побачити паралелі між різною архітектурою, які в кінцевому підсумку будемо використовувати в якості розгляду для експерименту. Оскільки кореляція не є причинно-наслідковим зв'язком, припустимо, що розробка з використанням ядер і додаткові продажі склали значну частину доходів, яку отримає компанія виробник. Але саме в частині ядра архітектури пристрою чи системи компанії виробники можуть також розпочати суперечку щодо інтелектуальної власності. Це буде друга причина через яку потрібно розробити метод, який би убезпечив витік інформації про внутрішню будову апаратного пристрою на основі його програмної моделі [5-9]. На сьогодні не існує жодного методу чи практики, для яких компанія могла б довести право власності. Таким чином, їй доведеться витратити як час, зусилля, так і ресурси на використання методів з реверс-інжинірингу архітектури апаратних пристроїв і визначення того, чи була включена унікальна частина її розробки в архітектуру пристроїв іншої компанії [10-12]. Не має методу або практики, за допомогою яких компанія могла б довести своє право власності, і при цьому не витративши значні витрати часу і ресурсів. Відсутність методів перевірки права власності, які були б включені в проекти, може бути пов'язана з різними причинами, наприклад, не існує простого методу, який би не впливав негативно на схему під час процесу проектування та життєвого циклу його розробки або метод не може бути легко інтегрований чи використаний з існуючими стандартними інструментами автоматизованого проектування.

Розглянемо опис моделі атак на апаратне забезпечення, Будемо класифікуючи ці атаки за трьома основними категоріями: час; впровадження несправностей; бічний канал. Також, описуємо стандартну модель атаки проти криптографічних алгоритмів. Для аналізу важливості і наслідків атак розглянемо ключові концепції інформаційної безпеки та криптографії [13-19]. Наявний стандарт визначає набір фундаментальних послуг, необхідних для інформаційної безпеки, які називаються тріадою CIA [13]: конфіденційність; цілісність; доступність. І навпаки, ця тріада полягала лише в тому, щоб вважати фундаментальною структурою і для повної інкапсуляції інформаційної безпеки необхідні додаткові послуги: автентичність; незаперечення. В цілому ці сервіси складають основи інформаційної безпеки і є основними принципами, які повинні підтримуватися криптографічними алгоритмами. Однак це не єдині концепції, необхідні алгоритмам, що використовуються для криптографії. Так що основними методами, що використовуються для побудови будь-якого криптографічного алгоритму, є принципи Шеннона [15]: заплутування; змішування. Ці концепції ґрунтуються на оригінальному відкритому тексті секретного ключа і результуючого шифротексту. Плутанина полягає в тому, що текст повинен залежати від декількох частин, в той час як змішування використовується для забезпечення лавинного ефекту. Тобто, невеликі зміни в одному початковому тексті повинні істотно впливати на результуючий текст [16]. Найбільш поширеними способами виконання плутанини і змішування є використання транспозиції і заміщення. Тобто, обфускації тексту(ОТ) і заміщення в ньому (ЗТ). Ідея перестановки, або мережі, може полягати в тому, щоб просто зашифрувати початковий відкритий текст з полем підстановки, що замінює вихідні символи простого тексту деякими іншими значеннями [17].

Основна мета будь-якого криптографічного алгоритму полягає в тому, щоб використовувати певний задіяний процес, так що спроба атакувати або вплинути на цілісність даних була неможлива. У той час як ОТ і ЗТ є лише кількома способами досягти цього, а інші алгоритми покладаються на математичні процеси для забезпечення інтенсивності обчислень і складності.

Таким чином, в архітектурі апаратних пристроїв і засобів може бути присутня зайва функція, яку було реалізовано виробниками навмисно з подальшим її використанням із зловмисною метою. Також, різні виробники можуть недоброчесно використовувати готові апаратні рішення, які отримують з програмних моделей. Їх перевірка забирає багато часу. Тому, ці дві проблеми вимагають розроблення методів для забезпечення швидкої перевірки програмних моделей апаратних засобів або розробки методів, які б базувались на шифруванні інтелектуальних частин програмних моделей, які б забезпечували захист від такий додаткових функцій або втрати інформації про архітектуру нових апаратних пристроїв.

Стратегія використання генетичних алгоритмів як засібу виявлення стороннього коду в програмних моделях апаратних пристроїв

Параметри, які зазвичай використовуються в ГА: розмір популяції (простір розв'язків ГА або кількість окремих розв'язків); генерація (єдина ітерація, в якій використовуються біологічні оператори для природного відокремлення та поповнення популяції); коефіцієнт елітарності (частка особин з популяції, яка гарантовано виживе і перейде в наступне покоління, тобто не замінюється потомством); частота мутацій (ймовірність появи потомства, яке міститиме гени, які не походять ні від батьків А, ні від В); частота кросинговеру (ймовірність того, що обрані батьки схрещують гени і виробляють потомство); придатність

(якісне представлення рішення, засноване на кількісному значенні, що випливає з цільової функції); ціль (бажане значення придатності (відстані до цілі) або рішення відносно задачі). Після цього функція ініціалізації популяції стохастично генерує (розмір популяції) індивідуальні рішення, які складуть весь набір членів генеральної сукупності, що використовуються в алгоритмі. Пристосованість - це кількісна величина, отримана з математичної функції, що має відношення до простору задач ГА. Функція сполучення складається з трьох етапів: елітарність; розмноження; мутація. Під час виконання функції елітарності частина популяції, визначена коефіцієнтом елітарності, буде гарантовано виживати в наступному поколінні, а решта членів популяції замінюються народженим потомством. Дочірня функція, як правило, є процесом до двох перемог, в якому двоє батьків народжують двох дітей, де кожна дитина має потенціал замінити кожного з батьків у наступному поколінні. При застосуванні кросовера діти створюються за допомогою кросоверної функції, і в найпростішій формі можуть бути представлені або в одноточковій, або в двоточковій, або в якійсь її варіації, де кожна з них відноситься до числа точок, в яких батьківська хромосома зрощується і гени отримані від кожного з батьків. У тому випадку, якщо батьки А і В мають ймовірність кросовера меншу, ніж ймовірність кросовера, то потомство не народжується, і батьки просто доживають до наступного покоління. Для правильного підбору батьків використовуються додаткові схеми, які будуть використовуватися в процесі спарювання, отримання потомства, де їх шанси на спаровування пропорційні пристосованості. Слідом за створенням потомства кожен піддається випадковим мутації, заснованої на частоті мутацій, де одним з простих прийомів, є метод виконання випадкового обміну. Один ген вибирається випадковим чином і замінюється випадковим геном, відсутнім в хромосомі.

У той час як ГА базується на теорієї еволюції Дарвіна, то традиційна структура ГА не враховує спостереження в природі. Таким чином, запропоновані удосконалення методу значною мірою ґрунтуються на цьому традиційному підході, реалізуючи різноманітні нові дарвінівські концепції та параметри, які визначимо наступним чином: неострівна багатопопуляція (множинні незалежні популяції споріднених видів отримують ту саму проблему); кросовер популяції (швидкість з якою дві популяції схрещуються, причому батьки належать до обох популяцій); плаваюча елітарність (поточний рівень елітизму, розрахований на кожне покоління, таким чином, що він базується на загальній пристосованості населення та адитивному рівні смертності); мікроскопічна елітарність (придатність розчину обчислюється на основі пристосованості окремих генів у хромосомі, що дозволяє домінувати та рецесивно розглядати гени); багатоцільовий фітнес (для оцінки рішень використовуються множинні фітнес-функції і їх не слід плутати з концепцією мікроскопічної або плаваючої елітарності); рівень смертності (адитивне значення до плаваючого елітизму для визначення кількості непридатних членів для повторного заселення); коефіцієнт народження дітей (верхня межа кількості дітей, яких двоє батьків можуть народити); розмір мутаційного гена (верхня межа кількості генів, які можуть мутувати під час однієї мутації); компенсація популяції (кількість поколінь до введення мігруючої популяції); тривалість життя (кількість поколінь, які використовується популяцією до її повного відновлення); популяційна мутація (коли всі члени популяції перевищують плаваюче значення елітарності, тоді використовується фіксований коефіцієнт елітарності, спаровування пропускається, і вся популяція мутує до тих пір, поки не з'являться нові члени); максимальні мутації (верхня межа кількості мутацій, яких може зазнати один член популяції під час популяційної мутації); стохастичний дарвінівський підхід (кількість дітей, кількість мутацій, мутовані гени, батьківський відбір, успадкування рецесивних генів і популяційний кросовер визначаються кожним поколінням; це зроблено для того, щоб змоделювати теорію Дарвіна про мінливість, зумовлену природою, а не одомашненням, тобто впливом людини або деяким контролем над процесом).

Визначаємо кодування проблеми як відображення станів у сторонньому знаку з станами в оригінальному алгоритмі. Де одна пара стану вважається геном, що утворює хромосому, членом популяції або потенційним рішенням. Практикою підвищення ефективності ГА є використання так званої острівної моделі, методу розпаралелювання для обробки субпопуляцій у межах ГА. Замість того, щоб використовувати цей існуючий підхід, ГА включає підходи щодо споріднених видів того ж роду та міграції нових мешканців. Таким чином, замість того, щоб використовувати острівну модель для обробки однієї популяції, концептуалізуємо і використовуємо модель, в якій початкова популяція сама по собі представляє цей острів, де через певний проміжок часу або поколінь унікальна популяція споріднених видів одного роду мігрує на нього. Такий підхід збільшує зусилля по боротьбі з конвергенцією популяцій, коли члени популяції з часом тяжіють до єдиного рішення. при цьому дозволяючи генерувати потенційно нові та унікальні рішення шляхом схрещування популяцій між мігруючими та острівними мешканцями. Завдяки використанню кроспопуляції можемо опосередковано включати концепції як варіацій через природу, так і врахування географічного розташування, тобто варіація від острівного до мігруючого виду шляхом спарювання і, хоча природня варіація залежить від наявності нового географічного місця.

Оцінка окремих генів дозволяє розглядати їх домінантним і рецесивним чином, забезпечуючи можливість використовувати квадратну методологію. Домінантним геном вважається ген, який відповідає цільовим критеріям проблеми, у цьому сценарії це картування, яке не потребує додавання країв, а рецесивними генами вважаються ті гени, які не відповідають цільовим критеріям. Таким чином, це за своєю

суттю створює елітарність і пристосованість у мікроскопічному масштабі та слідує з медичних міркувань.

Оскільки проблема стосується конкретно алгоритму ГА, то використовуємо багатоцільову придатність, так що, подібно до того, як це було помічено з недоліками та методів сторонніх знаків на основі країв, то використовуваний засіб все ще може мати ті ж недоліки. Є сценарії, в яких додавання бітів неминуче. Таким чином, придатність оцінюється як з точки зору кількості як ребер, так і бітів, які потрібно додати, щоб вмістити сторонній знак.

Використання плаваючого елітизму має на меті пом'якшити заміну змішувань нижче середньої, так що з часом пристосованість усіх членів поступово зростатиме і падатиме, але врешті-решт зближуватиметься до оптимального рішення. Він керується значенням смертності, що є додатковим значенням до середньої пристосованості популяції, і прогнозує врахування швидкості навчання в нейронних мережах. І навпаки, популяційна мутація використовує фіксований елітизм і концептуально моделює соціально-економічний статус альфа-членів в анімалістичних ієрархіях, згідно з теорією дарвінівської концепції мінливості в природі. Таким чином, лише членам альфа-верств найвищого рівня будуть гарантовані права на все при змінах, зумовлених природою (зміни в екологічній стійкості).

Розглянемо кросовери. Процес схрещування популяцій досить простий при розгляді декількох незалежних популяцій. Кожна популяція випадковим чином визначає, чи буде вона виконувати схрещування популяцій. При цьому, якщо розглянути сценарій, коли популяція А визначає, що вона повинна перехресно заселятися, то кожен член, повторно заселений в процесі спарювання, буде складатися з батька А з популяції А і батька В з популяції В, таким чином, що кожен з батьків вибирається випадковим чином з кожної популяції. Аналогічно, під час процесу спарювання двох батьків випадковим чином вибираються гени, які обидва є рецесивними. У процесі спарювання народжується кілька дітей на основі: батьківських домінантних генів А і рецесивних В; батьківських домінантних генів В і рецесивних А; домінантних А з випадковими рецесивними; домінантних В з випадковими рецесивними. При цьому повертається дитина з найкращою фізичною підготовкою.

Процес мутації - це просто обмін, коли розміри їх однакові, або заміна, коли один менший за другого. Спосіб заміщення є рандомізованим, таким чином, гени ітеративно обробляються, і випадковим чином вирішується, мутувати кожному гену чи ні.

Розглянемо поняття про сторонні знаки на основі кодування стану. Хоча цей метод детально описаний, але він має особливість, в якій він перетворює хеш-сигнатуру стороннього знаку в граф переходу стану, який потім відображається за допомогою ізоморфізму підграфу і прийняття рішення про завершення стану, де коди станів синтезу будуть застосовуватися до кодів хеш-функції. Геш-функція накладається на вихідний стан оптимальним чином, а додавання необхідних бітів / ребер та часткове застосування кодів станів для бієктивного відображення станів з хешу та оригінального ГА є достатнім. Розмір і складність цього ГА залежить від обох значень, обраних для змінних. Загальна кількість бітів, оброблених з гешу для генерації ГА, наприклад, розмір даних буде обробляти тільки перші біти будь-якої геш-сигнатури. Кількість бітів, що складають стан у геш-функції, наприклад, довжина кодування в бітах розглядатиме всі біти як символьний стан. Після цього результуючий ГА обробляється за допомогою ізоморфізму підграфу та прийняття рішення про завершення циклу, щоб вставити ГА та забезпечити дотримання кодів станів для оцінювання. Це досягається шляхом розгляду гешу як запиту обчислення, в той час як алгоритм також виробляє необхідні місця, для яких ребра повинні бути додані, щоб уможливити циклічність кодів станів, тобто, одночасно вирішуючи як ізоморфізм підграфу, так і проблеми завершення.

Метод сторонніх знаків на основі країв працює так, що геш використовувався для побудови ГА. Послідовність перетравлюється на основі довжини введення-виведення, а потім розкладається на ребра в певній послідовності, щоб її можна було реконструювати. Замість побудови графа, метод має на меті побудувати серію ребер, придатних для повторного використання, які можуть бути відображені/вставлені в оціночний критерій, де сигнатура може бути відтворена за допомогою комбінацій вводу/виводу. Таким чином, цей метод значною мірою покладається на відображення та вставку ребер/бітів для розміщення сторонніх знаків.

Потрібно забезпечити певний підхід до оцінювання результатів виконання методу. Щоб дослідити вплив методів на синтез, використаємо підмножину файлів для ГА, що перекриваються. Вони мають периферійну мобільність і існують комбінації введення-виведення, для яких поведінка не була явно заявлена. Такі ребра можуть бути додані до системи. І навпаки, решта з них обмежені сценаріями, в яких буде отримана будь-яка з бажаних переваг. При цьому уточнюється введення-виведення або знаходять ізоморфний розв'язок. Коли ці умови не спрацьовують, вся система повинна бути розширена принаймні на біт, щоб забезпечити необхідний простір для вставки країв і сторонніх знаків. Необхідна кількість бітів інтуїтивно визначається характеристиками і загальною кількістю ребер або комбінацій введення-виведення, які необхідно вставити.

Сценарій синтезу, який використовували при генерації нової базової лінії, буде включати фактори, а саме: використання алгебраїчного скрипту; відображення з використанням бібліотек; використання інструменту кодування, оскільки він не дозволяє використовувати задані користувачем значення кодування

стану. Конкретний перелік операцій, що виконуються в цьому сценарії враховує результуючі значення синтезу, отримані для кожної оцінки в термінах літералів, площі і затримки.

Таким чином, відображення, отримані для кожного з розмірів, були різними, в тому сенсі, що, хоча ізоморфні, кожен з них був зіставлений по-різному зі станами. Вільні і примусові стани кодів в ГА з стороннім знаком не були абсолютно однаковими в процесі синтезу. Оскільки метод є недетермінованим і в кінцевому підсумку залишений стохастичним процесом генерації рішень відображення, проте, після окремих запусків алгоритму сторонніх знаків, з метою створення невеликого діапазону потенційних стохастичних сторонніх знаків, єдиними рішеннями, що генеруються, є ті, що призводять до більш високих додаткових витрат на синтез. Однак, за допомогою транзитивних властивостей і теорії графів, які є ізоморфними, також матимемо ізоморфне відображення. Видалення кратних ребер не вплине на розв'язок, так що відображення, може бути безпечно змінено, що в кінцевому підсумку зменшить додаткові витрати обох екземплярів сторонніх знаків.

Розглянемо сценарій додавання стороннього знака нової базової лінії з графіками запитів, таким чином, що замість станів, буде становий оцінювальний ГА. Істотна відмінність полягає в тому, що істотний стан для включення графіків запитів більше відсутній, тому повинні повторно запустити процес для отримання нових результатів додаткових витрат синтезу, щоб провести порівняння впливу на базовий рівень, оскільки ізоморфізм більше не існує. Незважаючи на те, що додаткові витрати в даний час є значущими для обох випадків, повинні навіть якщо вони можуть бути, то базовий рівень в порівнянні з показником ГА, як правило, збільшив додаткові витрати. Додавання ребер і бітів збільшує додаткові витрати. Однак важливим фактором для розгляду є раніше згадані значення кодування вільного стану. Хоча вони, як правило, призначаються за принципом стратегії «першим прийшов першим обслуговується», можемо дослідити їх вплив на отримані додаткові витрати, що утворюються в результаті синтезу. Враховуючи, що базовий рядок має стани, а графіки запитів містять їх, тоді може дослідити справжній вплив вільного призначення на окремий стан. Таким чином, при наявності простору кодування, можливого для них і підмножини непримусових кодів станів, виконано синтез для кожного з цих значень стану. Сценарії синтезу, що стосуються інструменту синтезу, цільового ГА та значень призначення стану з сторонніми знаками, генеруються автоматично. Інструмент синтезу використовує скрипти та пов'язані з ними файли з сторонніми знаками, створені для виконання фізичного синтезу та генерації результатів постсинтезу сторонніх знаків.

Геш-функція спочатку намагатиметься обробити цифрові підписи в їх необробленому двійковому представленні, доводячи підхід до основного результату, оскільки отриманий сторонній знак, побудований ГА, завжди був повністю пов'язаним. Оптимальним знайденим рішенням було використання стандартної геш-функції для переправлення цифрового підпису та перетворення отриманого геш-рядка у формат, придатний для алгоритму зіставлення. Важливо, щоб для використання була обрана відповідна геш-функція, вільна від атак, але не містила надлишку бітів. Таким чином, після розгляду функцій, що стосуються вразливостей геш-функцій, що містяться в бібліотеці, можна стандартизувати використання гешу на цьому етапі.

При розгляді атак на геш-функції є дві основні проблеми при їх використанні для додавання сторонніх знаків за допомогою кодування стану: попереднє зображення; колізія. Для даного гешу та функції ці атаки можуть бути визначені як з'ясування, який цифровий підпис зробив геш, і що виробляє даний геш. Крім того, враховуючи, що стандартизація геш-функції є спільною і тому доступною всім, то атака не повинна мати можливості відтворити оригінальний підпис зі значень коду даного алгоритму, створити інший підпис зі значень коду стану, реконструювати цифровий підпис, який використовується у сторонніх знаках, щоб підтвердити право власності.

Розроблений алгоритм є гібридизованим ГА і частиною класу еволюційних алгоритмів (ЕА). в той час як типовий ГА заснований на дарвінівських принципах та включає концепції Дарвіна, а також додає багато нових концепцій і параметрів для кращого моделювання природного відбору, спарювання, мутації та процесу кросинговеру. Це основний алгоритм, що лежить в основі процесу зіставлення та доповнення між запитом і цільовою функцією. Незважаючи на те, що відомо, що схеми кодування станів впливають на синтез заданим чином, значення кодування одного стану має здатність повністю вилучити синтезовану систему з діапазону допустимих додаткових витрат. Проблема сторонніх знаків за допомогою коду тепер ще більше заглиблюється в багатометричну проблему оптимізації на багатьох фронтах що полягає в тому, який розмір, довжину кодування, вільні значення кодування та підпис вибрати. Кожен з них може негативно вплинути не тільки на складність графіку сторонніх знаків/запитів, але й на коди станів, що застосовуються, і на їх вільний вибір.

ГА — це еволюційні алгоритми, які моделюють природний відбір та еволюцію видів за допомогою спрощеної моделі, яка використовує оператори кросинговеру, спарювання та мутації, які використовуються на окремих членах популяції. Ці типи алгоритмів, як правило, сконструйовані таким чином, щоб краще забезпечувати більш ефективні рішення складних оптимізаційних задач. Типова структура, якої дотримуються ці ГА, така, що придатність є цільовою функцією для оцінки членів генеральної сукупності, а

ген позначає ітерацію алгоритму. Модель ГА зображена на рис. 1.

Параметри ГА, які зазвичай використовуються в ГА, і їх описи означають, що включення їх залишається за розробником програмного забезпечення, і хоча є загальним, та не завжди включаються.

Пристосованість обчислюється за допомогою деякої математичної функції, специфічної для задачі оптимізації, призначеної для якісної оцінки членів популяції; по суті, це «відстань від цілі» для члена популяції і, інтуїтивно, чим більша відстань від цілі, тим нижча якість індивіду. Наприклад, якщо хочемо реалізувати ГА для визначення змінних, то відомо, що ціль як індивід у популяції матиме «гени», і результуюче значення математичної функції для цих генів визначається як відстань від бажаної цілі, так і загальна якість особистості; це також означає, що придатність відповідає введеному рівнянню.

Під час цього процесу ініціалізації створюються та випадковим чином ініціалізуються загальні члени, визначені параметром розміру генеральної сукупності. Процес випадкової ініціалізації знову визначається проблемою, що розв'язується, і додатковими перевірками або правилами, необхідними для забезпечення генерації лише дійсних рішень. Наприклад, якщо потрібно випадковим чином побудувати графові структури, то всі вузли повинні бути унікальними і пов'язаними.

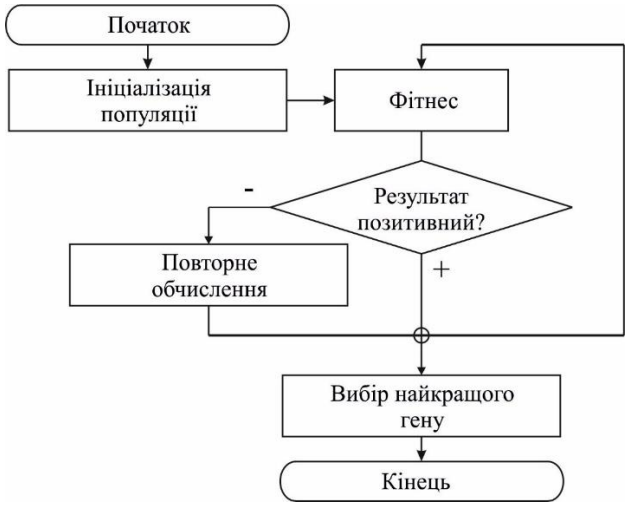

Рис. 1. Модель ГА

У процесі спарювання відбувається основна частина операцій в ГА, до поточної популяції застосовується фіксована елітарність, використовуються оператори батьківського відбору, кросинговер батьківських генів для створення нових членів популяції і мутація. Процес спаровування може бути розширений і краще проілюстрований. Батьківський відбір може здійснюватися різними способами.

Параметри дарвінівського генетичного алгоритму є достатніми для реалізації. На відміну від традиційної моделі ГА, націленої на проблему з однією ціллю в техніці сторонніх знаків на основі кодування стану, немає простої задачі з одним випадком, тому мета зіставлення запиту з цільовим значенням включає розгляд як ребер, так і бітів. У випадку, якщо значення є повністю визначеним, тоді повинні додати біт до системи, щоб врахувати необхідні додаткові переваги. І навпаки, коли значення є неповністю визначеним, то треба мінімізувати кількість ребер доданих до системи, зазначаючи, що ця стратегія також застосовна до наступного кроку.

Таким чином, в модель традиційного ГА внесено модифікації, які враховують особливості моделей сторонніх знаків, які вносяться в програмні моделі апаратних засобів.

Для проведення експериментів з встановлення ефективності розробленого підходу було розроблено застосунок, який реалізує ГА з модифікаціями. На вхід цього застосунку подається два файли. Перший файл містить програмну модель апаратного пристрою, а другий файл містить еталонну модель цього ж апаратного пристрою. В базі моделей сторонніх знаків зберігаються типові моделі внесення сторонніх знаків. Загалом розроблений застосунок виступає як класифікатор. В результаті його використання в експериментальних дослідженнях на штучних наборах вхідних моделей, він продемонстрував результат класифікації, який оцінено метрикою F_1, що дорівнює 82%. Таке значення є допустим. Для його покращення потрібно наповнювати базу моделей сторонніх знаків.

ВИСНОВКИ З ДАНОГО ДОСЛІДЖЕННЯ І ПЕРСПЕКТИВИ ПОДАЛЬШИХ РОЗВІДОК У ДАНОМУ НАПРЯМІ

Розглядаючи проблему сторонніх знаків, традиційна модель ГА не підходила. Це пов'язано з проблемами складності, які за своєю суттю виникають як при перетворенні гешу в функції, так і при

відображенні запиту на цільовий параметр. Зокрема, було проблемно створити задачі гамільтонового завершення в графі. Ця задача, заснована на рішеннях для визначення мінімальної кількості ребер у графі, яку необхідно додати, щоб забезпечити існування гамільтонового циклу. Відомо, що ця задача має компутаційну складність недетермінованого многочлена повного, крім того, знаходження самого гамільтонового циклу має складність NP-повного. Таким чином, було запропоновано модифікації, внесені в традиційну модель ГА на додаток до нових функцій, операторів тощо, які були використані.

Внесені модифікації в традиційну модель ГА дозволили розробити стратегію та підхід до виявлення сторонніх знаків внесених в програмні моделі апаратних засобів. Експериментальні дослідження щодо ефективності розробленого підходу показали результати класифікації з значенням F_1, що дорівнює 82%. Таке значення є прийнятним для класифікатору.

Напрямами подальших досліджень є удосконалення в підході, що базується на модифікаціях в генетичному алгоритмі. Ці удосконалення першочергово стосуватимуться врахування більшої кількості моделей внесення сторонніх знаків в програмні моделі апаратних засобів.

Література

1. Kumar A. Machine learning-based early detection of IoT botnets using network-edge traffic. *Computers & Securityю.* 2022.

2. Brooks R. Distributed denial of service (DDoS): a history. *IEEE Annals of the History of Computing.* 2021. V. 44. No. P. 44-54.

3. Janiesch C., Patrick Z., Kai H. Machine learning and deep learning. *Electronic Markets.* 2021. V. 31. No. 3. P. 685-695.

4. Han Y. Dynamic neural networks: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence.* 2021. V. 44. No. 11. P. 7436-7456.

5. Swamy S. N., Solomon R. K. An empirical study on system level aspects of Internet of Things (IoT). *IEEE Access* 8. 2020. P. 188082-188134.

6. Denysiuk D., Savenko O., Lysenko S., Savenko B., Kashtalian A. Method for Detecting Steganographic Changes in Images Using Machine Learning. In: *2023 13th International Conference on Dependable Systems, Services and Technologies (DESSERT)*, Athens, Greece. 2023. P. 1-6. doi:10.1109/DESSERT61349.2023.10416453.

6. Moshkovitz M. Explainable k-means and k-medians clustering. In: *International Conference on Machine Learning*. PMLR. 2020. P. 7055-7065.

7. Zhang Y., Jiang T., Wang H. Challenges and Solutions in Botnet Detection Using Clustering Algorithms. *International Journal of Network Security.* 2022. V. 24. No. 2. P. 112-124.

8. Aslan Ö., Refik S. A comprehensive review on malware detection approaches. *IEEE Access 8*. 2020. P.6249-6271.

9. Letteri I., Antonio D. C., Giuseppe D. P. New optimization approaches in malware traffic analysis. In: *International Conference on Machine Learning, Optimization, and Data Science*. Cham: Springer International Publishing. 2021. P. 57-68.

10. Markowsky G., Savenko O., Lysenko S., Nicheporuk A. The technique for metamorphic viruses' detection based on its obfuscation features analysis. *CEUR-WS* 2104. 2018. P. 680-687.

11. Lysenko S., Savenko Ol., Bobrovnikova K. DDoS Botnet Detection Technique Based on the Use of the Semi-Supervised Fuzzy c-Means Clustering. *CEUR-WS* 2104. 2018. P. 688-695.

12. Savenko B., Lysenko S., Bobrovnikova K., Savenko O., Markowsky G.. Detection DNS Tunneling Botnets // *Proceedings of the 2021 IEEE 11th International Conference on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications (IDAACS), IDAACS'2021*, Cracow, Poland, September 22-25, 2021.

13. Suryati O. T., Avon B. Impact analysis of malware based on call network API with heuristic detection method. *International Journal of Advances in Data and Information Systems.* 2020. V.1. No. 1. P. 1-8.

14. Wurzinger P. Automatically generating models for botnet detection. In: *Computer Security–ESORICS 2009: 14th European Symposium on Research in Computer Security, Saint-Malo, France, September 21-23, Springer Berlin Heidelberg Proceedings.* 2009. 2009. V. 14. P. 232-249.

15. Haddadi F., Zincir-Heywood A. N. Botnet detection using network flow analysis and support vector machines. *Computer Networks.* 2020. V. 181.

16. Zhao J., Liu Y., Luo X. Deep learning for botnet detection: A survey. *IEEE Access* 9. 2021. P. 82771-82785.

17. Mirsky Y., Doitshman T., Elovici Y., Shabtai A. Kitsune: An ensemble of autoencoders for online network intrusion detection. *IEEE Transactions on Information Forensics and Security.* 2021. V. 16. P. 122-133.

18. Смірнов О.П., Поплавський С.Ю., Ковальчук В.К., Лутюк Л.І. Удосконалений метод та засоби криптографічного захисту від вразливостей в апаратному забезпеченні / *Збірник наукових праць за*

матеріалами XV Всеукраїнської науково-практичної конференції «Актуальні проблеми комп'ютерних наук АПКН-2023». Хмельницький, 2023, С. 278-279. https://kn.khmnu.edu.ua/wp-content/uploads/sites/18/apkn-2023-corpuspaper.pdf

References

1. Kumar A. Machine learning-based early detection of IoT botnets using network-edge traffic. *Computers & Securityю.* 2022.
2. Brooks R. Distributed denial of service (DDoS): a history. *IEEE Annals of the History of Computing.* 2021. V. 44. No. P. 44-54.
3. Janiesch C., Patrick Z., Kai H. Machine learning and deep learning. *Electronic Markets.* 2021. V. 31. No. 3. P. 685-695.
4. Han Y. Dynamic neural networks: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence.* 2021. V. 44. No. 11. P. 7436-7456.
5. Swamy S. N., Solomon R. K. An empirical study on system level aspects of Internet of Things (IoT). *IEEE Access* 8. 2020. P. 188082-188134.
6. Denysiuk D., Savenko O., Lysenko S., Savenko B., Kashtalian A. Method for Detecting Steganographic Changes in Images Using Machine Learning. In: *2023 13th International Conference on Dependable Systems, Services and Technologies (DESSERT)*, Athens, Greece. 2023. P. 1-6.
doi:10.1109/DESSERT61349.2023.10416453.
6. Moshkovitz M. Explainable k-means and k-medians clustering. In: *International Conference on Machine Learning*. PMLR. 2020. P. 7055-7065.
7. Zhang Y., Jiang T., Wang H. Challenges and Solutions in Botnet Detection Using Clustering Algorithms. *International Journal of Network Security.* 2022. V. 24. No. 2. P. 112-124.
8. Aslan Ö., Refik S. A comprehensive review on malware detection approaches. *IEEE Access 8*. 2020. P.6249-6271.
9. Letteri I., Antonio D. C., Giuseppe D. P. New optimization approaches in malware traffic analysis. In: *International Conference on Machine Learning, Optimization, and Data Science*. Cham: Springer International Publishing. 2021. P. 57-68.
10. Markowsky G., Savenko O., Lysenko S., Nicheporuk A. The technique for metamorphic viruses' detection based on its obfuscation features analysis. *CEUR-WS* 2104. 2018. P. 680-687.
11. Lysenko S., Savenko Ol., Bobrovnikova K. DDoS Botnet Detection Technique Based on the Use of the Semi-Supervised Fuzzy c-Means Clustering. *CEUR-WS* 2104. 2018. P. 688-695.
12. Savenko B., Lysenko S., Bobrovnikova K., Savenko O., Markowsky G.. Detection DNS Tunneling Botnets // *Proceedings of the 2021 IEEE 11th International Conference on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications (IDAACS), IDAACS'2021*, Cracow, Poland, September 22-25, 2021.
13. Suryati O. T., Avon B. Impact analysis of malware based on call network API with heuristic detection method. *International Journal of Advances in Data and Information Systems.* 2020. V.1. No. 1. P. 1-8.
14. Wurzinger P. Automatically generating models for botnet detection. In: *Computer Security–ESORICS 2009: 14th European Symposium on Research in Computer Security, Saint-Malo, France, September 21-23, Springer Berlin Heidelberg Proceedings.* 2009. 2009. V. 14. P. 232-249.
15. Haddadi F., Zincir-Heywood A. N. Botnet detection using network flow analysis and support vector machines. *Computer Networks.* 2020. V. 181.
16. Zhao J., Liu Y., Luo X. Deep learning for botnet detection: A survey. *IEEE Access* 9. 2021. P. 82771-82785.
17. Mirsky Y., Doitshman T., Elovici Y., Shabtai A. Kitsune: An ensemble of autoencoders for online network intrusion detection. *IEEE Transactions on Information Forensics and Security.* 2021. V. 16. P. 122-133.
18. Smirnov O.P., Poplavskyi S.Yu., Kovalchuk V.K., Lutyuk L.I. An improved method and means of cryptographic protection against vulnerabilities in hardware / *Collection of scientific papers based on the materials of the XV All-Ukrainian scientific and practical conference "Actual problems of computer science APKN-2023"*. Khmelnytskyi, 2023, pp. 278-279. https://kn.khmnu.edu.ua/wp-content/uploads/sites/18/apkn-2023-corpuspaper.pdf (In Ukrainen)