КІБЕРБЕЗПЕКА: освіта, наука, техніка
CYBERSECURITY: EDUCATION, SCIENCE, TECHNIQUE
№ 4 (12), 2021
ISSN 2663 - 4023
DOI 10.28925/2663-4023.2021.12.108116
УДК 654.071

**Журавчак Даниїл Юрійович**
аспірант, асистент кафедри захисту інформації
Національний Університет "Львівська Політехніка", Львів, Україна
ORCID ID: 0000-0002-8461-8996
*danyil.y.zhuravchak@lpnu.ua*

# СТВОРЕННЯ СИСТЕМИ ЗАПОБІГАННЯ ПОШИРЕННЯ ВІРУСІВ ВИМАГАЧІВ ЗА ДОПОМОГОЮ МОВИ ПРОГРАМУВАННЯ PYTHON ТА УТИЛІТИ AUDITD НА БАЗІ ОПЕРАЦІЙНОЇ СИСТЕМИ LINUX

**Анотація.** Орієнтований на дані період створює все більше проблем, пов'язаних із безпекою, з якими навіть експерти навряд чи можуть впоратися. Однією з найскладніших загроз є віруси-вимагачі/віруси-шифрувальники, яких дуже важко виявити і ще важче вчасно заблокувати. Для успішного виявлення вірусу-вимагача у мережі, чи на кінцевій станції необхідно обробляти дуже велику кількість даних. Успішність кореляції полягає у різноманітності джерел даних. Під час дослідження методів дії вірусів- вимагачів було виявлено, що основна мета є вимагання викупу за розшифрування даних, які знаходилися на файловій системі та під час проникнення на систему вірус-вимагач успішно зашифрував. На теренах України перша глобальна атака віруса-вимагача(NotPetya) була 27 червня 2017 року. На думку Адміністрації Президента США Дональда Трампа атака із використанням вірусу NotPetya в червні 2017 року стала найбільшою хакерською атакою в історії. У спільній заяві країни альянсу Five Eyes поклали відповідальність за дану атаку на російську владу. Відповідальність за атаку покладають на Росію також уряди Данії та України. Багато аналітик назвали ці дії не просто політичного характеру, а саме воєною агресію. Досліджуючи методи виявлення та протидії вірусам-вимагачам було знайдено метод-пастку honeypot. Було заплановано розробити самотужки систему honeypot на базі файлової системи. Методи розроблені під час дослідження показали суттєві результати у виявленні вірусів-вимагачів із використанням концепції програмних приманок за допомогою символічними/легкими посиланнями операційної системи Linux, зокрема Ubuntu, для зменшення пошкодження файлової системи. Під час дослідження дотримувались показників CIA (конфіденційність, цілісність, доступність), як показників моніторингу безпеки комп'ютерної мережі. Запропонований метод пропонує оптимізувати процедуру реагування та прискорити процедуру знищення зловмисного програмного забезпечення та запровадити співпрацю між штучним інтелектом та людиною для покращення класифікації та виявлення програм-вимагачів.

**Ключові слова:** програмні приманки; безпека; віруси-вимагачі; шифрування; python; auditD; linux.

## ВСТУП

Постійний розвиток цифрових інновацій дозволяє використовувати нові способи зберігання та передачі інформації. Цифрові дані стали одним із найважливіших надбань різних державних установ та комерційних організацій. Факт високої цінності даних, особливо в корпоративному середовищі, зробив віруси-вимагачі однією з найбільших загроз бізнесу, перш за все, якщо цей бізнес керується принципами "прийняття рішень на основі даних", "бізнес-аналітика", видобуток даних , великі дані [1, 2]. Ось чому багато установ стають вразливими до атак вірусів-вимагачів в середовищі

інформаційних технологій та цифрових комунікацій. Відповідно, як розробка, так і вдосконалення методів та процедур їх попередження, виявлення та знищення в режимі реального часу стають критично важливими.

Виявити та, крім того, запобігти атаці віруса-вимагача складно через складність використовуваних алгоритмів шифрування. Як тільки пристрій стає «зараженим», інші системи та компоненти, підключені до мережі, також стають вразливими. Однак виявлення та / або запобігання загрозі в мережі або кінцевій точці можливе різними способами: антивірусні та мережеві системи захисту від шкідливих програм; моніторинг; використовуючи принцип honeypot або "приманка". Наприклад, Sysmon може збирати дані про підозрілі дзвінки WinAPI(У екосистемах компанії Microsoft) або системні дзвінки у операційних системах на базі Unix/Linux та використовувати файли системних журналів для повідомлення про шифрування даних.

Однак таке рішення, як моніторинг мережі / файлової системи, може бути недостатньо ефективним і недостатньо швидким для виявлення та запобігання пошкодженню в результаті атак вірусів-вимагачів. Наприклад, виробник обладнання та програмного забезпечення Acer зазнав нападу REvil у березні 2021 року, коли зловмисники змусили компанію заплатити 50 мільйонів доларів [3]. Незважаючи на те, що Acer постійно контролює свої IT-системи, така загроза сталася, тому доцільно використовувати конкретні механізми запобігання. У липні 2020 року на іншу технологічну компанію Garmin напали віруси-вимагачі, що призвело до повної та часткової недоступності веб-сайту, підтримки клієнтів та програм користувачів [4].

У цій статті ми намагаємось вирішити проблему низької ефективності протидії та відсутності виявлення програм-вимагачів та оцінюємо методи запобігання зараженню системи вимогами кібербезпеки. Атака віруса-вимагача складається з декількох послідовних кроків, протягом яких виявлення загрози може зайняти від кількох годин до місяця або більше. Навіть фахівці з безпеки не завжди можуть розшифрувати файли за допомогою складних ключів дешифрування [5]. Зрештою, незважаючи на існування різноманітних засобів виявлення загрози атаки вірусів-вимагачів, трапляються численні інциденти, і навіть у найбільш захищених кіберсистемах компаній, які зазнають цієї загрози. Ось чому для протидії загрозам даного типу необхідна ефективна та інтегрована система, здатна аналізувати та захищати надійний контроль файлових / мережевих ресурсів. Ця проблема особливо актуальна під час Індустрії 4.0/ Industry 4.0, оскільки програма вимагання може потенційно спричинити вихід з ладу декількох пристроїв IoT, знизити продуктивність пристроїв та незаконно отримати конфіденційну інформацію.

Вирішення проблеми буде цінним для таких типів цільових аудиторій, як антивірусні компанії, дослідницькі установи та великі корпорації, які мають важливе значення для забезпечення захисту даних та забезпечення дотримання політики конфіденційності та безпеки. Це дослідження дозволить їм застосовувати та вдосконалювати захист від шкідливого програмного забезпечення, використовуючи запропонований нами спосіб створення файлів-приманок та виконувати виявлення шкідливого програмного забезпечення на основі нового підходу в системному моніторингу та аналізі поведінки.

Мета цієї роботи – створити систему виявлення та протидії вірусам-вимагачам за допомогою мови програмування python та утиліти auditD на базі операційної системи Linux.

**Постановка проблеми.** Попри існування різноманітних засобів для ідентифікації вірусів-вимагачів, численні інциденти трапляються і навіть найбільш інформаційно-захищені компаніях, які піддаються цій загрозі та мають певний набір вразливостей. У

цій статті ми намагаємося покращити ефективність виявлення, а також запобігання зараження систем вірусами-вимагачам та суміжним загрозам. Саме тому наявність ефективної та цілісної системи, здатної аналізувати та забезпечувати надійний захист та контроль файлових/мережевих ресурсів є необхідним для протистояння небезпекам у кіберпросторі.

**Аналіз останніх досліджень і публікацій.**

Віруси-вимагачі - це тип шкідливого програмного забезпечення, яке блокує доступ до об'єкта чи його даних або блокує різні критичні процеси. Метою цієї акції є зловживання грошима жертв-вимагачів [6]. Цей тип загрози діє за наступним алгоритмом:

1. Вірус-вимагач сканує файлову систему.
2. Вірус-вимагач зашифровує критичні файли, використовуючи надійні алгоритми шифрування.
3. Ключі шифрування відправляються на сервер зловмисника.

Також необхідно зауважити, що віруси-вимагачі досить часто видаляють ключі шифрування. Даний тип атаки можна оцінювати не як вимагацький, чи терористичний, а саме, деструктивний. Саме так працював вірус-вимагач NotPetya на просторах України у 2014 році. [9]

Сучасні атаки вірусів-вимагачів шифрують внутрішні файли системи та поширюють їх по мережі до інших ресурсів. Цей підхід вірусу означає, що інші активи в тій же внутрішній мережі інфіковані, а більш критичні, чи конфіденційні дані шифруються для подальшого вимагання викупу.

Захист від вірусів-вимагачів - це постійний процес, який вимагає щоденних досліджень та розробок у сфері виявлення та запобігання, оскільки сторона, що атакує, постійно розробляє методи вступного проникнення з використанням тактики обходу антивірусного забезпечення.

Останні наукові праці відображають нові методи забезпечення максимальної безпеки інформаційних систем, ефективні методи системного аналізу та виявлення шкідливого програмного забезпечення. Однак у цій галузі є місце для вдосконалення. Незважаючи на популярність виявлення поведінкових програм, що базується на поведінковому аналізі, все ще бракує чітких конструкцій та методів вирішення, щоб швидко ідентифікувати загрози та усунути загрозу з мінімальними або відсутніми втратами. [7-9]

У своєму дослідженні Харраз А. і Кірда Е розробили нову систему запобігання вірусам-вимагачам, яка забезпечує високу стійкість. Їхній підхід заснований на аналізі поведінки програмного забезпечення та вимагає мінімальних змін операційної системи. Їх системи контролюють програми вводу-виводу I/O, які вимагають патернів для кожного процесу, щоб виявити ознаки поведінки, подібної до вірусів-вимагачів. Процес припиняється, коли спостерігається ненормальна поведінка у комп'ютерній мережі, чи на кінцевій станції. [10]

Ця стаття спрямована на підвищення ефективності виявлення та запобігання вірусам-вимагачам за допомогою моніторингу honeypots та здійснення ідентифікації шкідливих програм на основі взаємодії з файловими системами. Існує попит на безперервне полювання на загрозливі програми-вимагачі, оскільки це все ще серйозна проблема. З цього випливає план, який виявлятиме атаки вірусів-вимагачів на основі взаємодії з файловими системами.

Таблиця 1 містить порівняння різних систем виявлення та запобігання вірусам-вимагачам. CryptoLock має власну метрику втрати даних, показує високий рівень

виявлення та має невеликий відсоток помилкових спрацьовувань, які можуть бути покращені. Він використовує моніторинг у реальному часі для виявлення даних про зміни користувачів [11]. При роботі з вірусами-вимагачами допустимий невеликий відсоток втрати даних, тому що досить часто алгоритми виявлення починають працювати після того, як вірус-вимагач активно шифрував дані.

Ентропія Шеннона та fuzzy hash техніка є найменш точними методами у нашому порівнянні. Однак він має 0% хибнопозитивних результатів [12].

Інструмент AIRad повністю заснований на різних статистичних та машинних методах навчання [13]. Швидкість виявлення дуже висока, враховуючи цю інтелектуальну систему.

DIMAQS (база даних вірусів-вимагачів) - це база даних із відомими вірусами-вимагачами [14]. Таке рішення є найменш ефективним при роботі з новими зразками шкідливого програмного забезпечення та їх виявленні, оскільки їх може не бути в базі даних.

*Таблиця 1*

**Порівняльний аналіз виявлення вірусів-вимагачів методами: CryptoLock, Shannon entropy, AIRad, DIMAQS**

| **Вірус-вимагач** | **Втрата даних** | **Швидкість виявлення** | **Процент хибного виявлення** |
|---|---|---|---|
| CryptoLock | 0.2% (10/5099) | 100% | 3.8% |
| Shannon’s entropy & fuzzy hash technique | - | 95.7% | 0% |
| AIRad | - | 99.54% | 0.0005% |
| DIMAQS | - | 100% | 0% |

**Мета статті.** Мета цієї роботи – створити систему виявлення та протидії вірусам-вимагачам за допомогою мови програмування python та утиліти auditD на базі операційної системи Linux.

## РЕЗУЛЬТАТИ ДОСЛІДЖЕНЬ

У цьому дослідженні ми використовували файловий метод honeypot для виявлення та запобігання атакам-вимогам [15]. Ця концепція успішно використовувалась раніше і показала приблизно добрі результати. [16] Крім того, в дослідженнях інших авторів застосовується використання машинного навчання, що вдосконалює механізм виявлення загроз. Особливістю цього підходу є використання медових банок як пасток для підозрілих програм та пакетів, тоді як алгоритми машинного навчання аналізують поведінку та класифікують віруси / шкідливі програми [17].

Основною ідеєю концепції honeypot є створення синтетичних файлів та папок для відстеження змін на файловій системі операційної системи. Цей підхід до виявлення програм вимагання повинен мати найнижчий показник false positive(хибне спрацювання) першого виду і є метою для більшості вірусів-вимагачів, оскільки він базується на аналізі поведінки програмного забезпечення. Однак вище вказаний спосіб має певні недоліки: наявність більше 1000 і більше цільових приманок значно збільшує навантаження на центральний процесор. Ось чому ми вважаємо, що доцільно застосовувати суттєве і чітко визначене значення файлових приманок для виявлення програм-вимагачів.

ISSN 2663 - 4023

Для підтвердження запропонованого нами підходу було використано кілька зразків вірусів-вимагачів, які перераховують файли на файлові системі та запускають процес шифрування.

Курс дослідження складається з декількох послідовних етапів, починаючи з підготовки середовища. Робочий процес системи виявлення та запобігання вимогам у файловій системі пристрою показаний на рис.1. Файли Honeypot потрібно спочатку створити з функцією, яка приймає два аргументи 1) шлях, де слід зберігати файли, і 2) розмір файлу. Функція повертає абсолютний шлях до файлової приманки, яка була створена нашою системою.

Потім виконується створення символічного посилання, яке вказує на каструлю. Цей процес автоматизований за допомогою функції, яка також приймає два аргументи - 1) шлях до приманки та 2) шлях, де відбуватиметься рекурсивне створення символічних посилань на медовий банк. Назва файлів приманки повинна відповідати наступним вимогам:

1. бути прихованим для користувачів, щоб не відбулося випадкового дії видалення.
2. мати комбінацію цифр та спеціальних символів, щоб файл міг знаходитися зверху в каталозі файлової системи.

Після того, як середовище та файли honeypot готові, налаштовується моніторинг окремих файлів. Для моніторингу можна використовувати такі опції:

1. Watchdogs - це бібліотека Python, яка використовує у ролі «сторожові собаки» для моніторингу файлів і каталогів, діяльність яких генерує події. Однак він не реєструє інформацію, пов'язану з процесом на рівні операційної системи. Отже, потрібне інше рішення.
2. fuser - це утиліта в Linux, яка ідентифікує процеси за допомогою файлів або сокетів. Всі процеси запущені з прапором “-kw” він знайде всі процеси запису в нашу систему і вб’є їх. Недоліком використання цього інструменту є тривалість перерахування всіх процесів та їх відкритих файлів, тому нам потрібно знайти більш надійний інструмент моніторингу.
3. auditd - найкращий варіант для наших потреб моніторингу досліджень. Утиліта AuditD - це демон системного аудиту подій в Linux, який створює операції та інциденти на основі системних викликів.

Використовуючи інструмент auditd, ми успішно реєструємо доступ до записів до файлів honeypot та автоматизуємо виконання моніторингу. Правила аудиту налаштовані на відстеження доступу для читання / запису до файлової системи. Наступним кроком є контроль за доступом до файлів honeypot, що є постійним процесом. Кульмінація відбувається, коли надходить повідомлення про виявлення загрози та усувається шкідливий процес, що дійшов до нашої системи. Процеси вірусу-вимагача успішно знищені; виявлення триває у режимі реального часу.

Коли інцидент буде вирішено, експерти з кібербезпеки аналізують системні події та журнальні файли, щоб визначити джерело загрози. Для подальшого дослідження для автоматизації цього процесу можуть бути розроблені додатки та програми для автоматизації та, можливо, штучного інтелекту.

Для перевірки результатів дослідження були використані три зразки програм-вимагачів зі сховища GitHub:

1. GonnaCry [18];
2. JavaRansomware [19];
3. RAASNet [20].

У таблиці 2 описані результати перевірки. Ключовим фактором, який впливав на швидкість виявлення та відповіді, є реалізація алгоритму криптографії та швидкість його шифрування. Наприклад, JavaRansomware зашифрував 30 файлів, 27 з яких були нашими спеціальними медовими банками. Сигнал вбивства JavaRansomware був надісланий під час процесу шифрування, перш ніж ми змогли видалити шкідливий файл. Оптимальним і бажаним процесом виявлення та реакції для нашої системи є такий - нам потрібно виявляти та видаляти програм-вимагачів одночасно, коли воно звертається до нашого файлу та торкається його. У випадку з JavaRansomware цього не вдалося досягти. Відповідно доцільно працювати над оптимізацією системи для швидкого виявлення та усунення процесів шифрування.

У таблиці нижче ми відстежували середній час виявлення загрози, а також середній час її усунення. Час ліквідації визначається з моменту встановлення загрози до її знищення. Сума часу виявлення та часу усунення відображатиме тривалість дії загрози в системі від створення до припинення.

*Таблиця 2*

**Результати виявлення і протидії вірусам-вимагачам: GonnaCry, JavaRansomware, RAASNET**

| **Вірус-вимагач** | **Зашифровані файли перед вбивством процесу (кількість)** | **Зашифровані файли honeypot перед вбивством (кількість)** | **Середній час виявлення (секунди)** | **Середній час на вбивство (секунди)** | **Загальний час (секунди)** |
|---|---|---|---|---|---|
| GonnaCry | 2 | 0 | 0.98 | 1.75 | 2.73 |
| JavaRansomware | 30 | 27 | 1.98 | 3.46 | 5.44 |
| RAASNet | 5 | 0 | 1.19 | 1.12 | 2.31 |

## ВИСНОВКИ ТА ПЕРСПЕКТИВИ ПОДАЛЬШИХ ДОСЛІДЖЕНЬ

Мета дослідження досягнута шляхом аналізу різних атак вірусів-вимагачів, їх наслідків та створення профілактичного методу, заснованого на файлових програмах-приманках із використанням символічних посилань операційної системи Linux. Ми прийшли до висновку, що за допомогою нашого підходу можна дуже швидко виявити віруси-вимагачі за дуже короткий проміжок часу. Однак є ще можливості вдосконалити це рішення та зробити його готовим до роботи в корпоративних мережах. Зокрема, деякі вдосконалення середнього часу для виявлення / припинення підозрілого процесу були б корисними. Я вважаю, що створення достатньої кількості файлів приманок на основі різних розрахунків значно допоможе поліпшити / підтримати хороший стан інфраструктури та захистити систему. Ми розглядаємо можливість використання систем управління журналами та SIEM як доповнення до нашого методу, а також створення процесного демона для операційної системи, а також додаємо більше правил для політики auditD для покращення точності ідентифікації програм-вимагачів.

## СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ


1. Shakhovska, N., Fedushko, S., Melnykova, N., Shvorob, I., & Syerov, Y. (2019). Big Data analysis in development of personalized medical system. *Procedia Computer Science*, 160, 229-234.
2. Fedushko, S, Ustyianovych, T, Gregus, M. (2020). Real-Time High-Load Infrastructure Transaction Status Output Prediction Using Operational Intelligence and Big Data Technologies. *Electronics*; *9*(4), 668.
3. Kwan, C. (2021). Acer reportedly targeted with $50 million ransomware attack. https://www.zdnet.com/
4. Adler, S. (2020). Incident Of The Week: Garmin Pays $10 Million To Ransomware Hackers Who Rendered Systems Useless. https://www.cshub.com/
5. Tailor, Jinal P., and Ashish D. Patel. (2017). A comprehensive survey: ransomware attacks prevention, monitoring and damage control. *International Journal of Scientific Research 4(VIS)*, 15, 116-121.
6. Ross, B. (2016). Ransomware attacks: detection, prevention and cure. *Network Security*, 9, 5-9.
7. Dudykevych, V., Prokopyshyn, I., Chekurin, V., Opirskyy, I., Lakh, Y., Kret, T., Ivanchenko, Y., Ivanchenko, I. (2019). A multicriterial analysis of the efficiency of conservative information security systems. *Eastern-european journal of enterprise technologies. Information and controlling system*, *3*(9(99)), 6-13.
8. Vasylyshyn, S., Opirskyy, I., Susukailo, V. Analysis of the use of software baits as a means of ensuring information security // 2020 *IEEE 15th International Scientific and Technical Conference on Computer Sciences and Information Technologies*, CSIT 2020 - Proceedings, 2020, 2, pp. 242–245, 9321925
9. Hu, Z., Khokhlachova, Y., Sydorenko, V., Opirskyy, I. (2017). Method for Optimization of Information Security Systems Behavior under Conditions of Influences. I*nternational Journal of Intelligent Systems and Applications (IJISA)*, *9*(12), 46-58.
10. Kharraz, A., Kirda, E. (2017) Redemption: Real-Time Protection Against Ransomware at End-Hosts. In: Dacier M., Bailey M., Polychronakis M., Antonakakis M. (eds) Research in Attacks, Intrusions, and Defenses. RAID 2017. Lecture Notes in Computer Science, 10453. Springer, Cham. https://doi.org/10.1007/978-3-319-66332-6_5
11. Scaife, N., Carter, H., Traynor, P., & Butler, K. R. (2016, June). Cryptolock (and drop it): stopping ransomware attacks on user data. In *2016 IEEE 36th International Conference on Distributed Computing Systems (ICDCS)* (pp. 303-312). IEEE.
12. Joshi, Y.S., Mahajan, H., Joshi, S.N. et al. (2021). Signature-less ransomware detection and mitigation. *J Comput Virol Hack Tech.* https://doi.org/10.1007/s11416-021-00384-0
13. Poudyal, S., Dasgupta, D. (2020). AI-Powered Ransomware Detection Framework, *In IEEE Symposium Series on Computational Intelligence (SSCI)*, (pp. 1154-1161). https://doi: 10.1109/SSCI47803.2020.9308387.
14. Hagen, C., Dmitrienko, A., Iffländer, L., Jobst, M., & Kounev, S. (2018). Efficient and effective ransomware detection in databases. In *Annu. Comput. Secur. Appl. Conf.(ACSAC)*.
15. Moore, C. (2016). Detecting Ransomware with Honeypot Techniques. In *Cybersecurity and Cyberforensics Conference (CCC)*, 77-81. https://doi: 10.1109/CCC.2016.14.
16. Sethia, V., Jeyasekar, A. (2019). Malware Capturing and Analysis using Dionaea Honeypot. *International Carnahan Conference on Security Technology (ICCST)*, Chennai, India, (pp. 1-4). https://doi: 10.1109/CCST.2019.8888409.
17. Matin, I. M. M., & Rahardjo, B. (2019). Malware detection using honeypot and machine learning. У *2019 7th international conference on cyber and IT service management (CITSM)*. IEEE. https://doi.org/10.1109/citsm47753.2019.8965419
18. tarcisio-marinho (2020). https://github.com/tarcisio-marinho/GonnaCry
19. Panagiotis Drakatos (2017). JavaRansomware. https://github.com/PanagiotisDrakatos/JavaRansomware
20. Leon Voerman (leonv024) (2020). RAASNet. https://github.com/leonv024/RAASNet

КІБЕРБЕЗПЕКА: освіта, наука, техніка
CYBERSECURITY: EDUCATION, SCIENCE, TECHNIQUE
№ 4 (12), 2021
ISSN 2663 - 4023

**Danyil Y. Zhuravchak**
Postgraduate student of Information Security Department
Lviv Polytechnic National University, Lviv, Ukraine
ORCID ID: 0000-0002-8461-8996
*danyil.y.zhuravchak@lpnu.ua*

# RANSOMWARE SPREAD PREVENTION SYSTEM USING PYTHON, AUDITD AND LINUX

**Abstract.** The data-driven period produces more and more security-related challenges that even experts can hardly deal with. One of the most complex threats is ransomware, which is very taxing and devastating to detect and mainly prevent. The success of correlation lies in the variety of data sources. During the study of the methods of action of ransomware viruses, it was found that the main purpose is to demand ransom for decryption of data that were on the file system and during the penetration of the system, the ransomware virus successfully encrypted. The first global attack of the ransowmare (NotPetya) on the territory of Ukraine was on June 27, 2017. According to the Administration of US President Donald Trump, the attack using the NotPetya virus in June 2017 became the largest hacker attack in history. In a joint statement, the Five Eyes claimed responsibility for the attack on Russian authorities. The governments of Denmark and Ukraine are also blaming Russia for the attack. Many analysts have called these actions not just political in nature, but military aggression. A honeypot trap method was found while researching methods for detecting and counteracting ransomware. It was planned to develop a honeypot system on its own based on the Linux file system. Our research methods showed significant results in identifying ransomware processes using the honeypot concept augmented with symbolic linking to reduce damage made to the file system. The CIA (confidentiality, integrity, availability) metrics have been adhered to. We propose to optimize the malware process termination procedure and introduce an artificial intelligence-human collaboration to enhance ransomware classification and detection.

**Keywords:** cybersecurity; ransomware; encryption; decryption; ransomware preventive measures; threat; information security; incident response.

## REFERENCES

1. Shakhovska, N., Fedushko, S., Melnykova, N., Shvorob, I., & Syerov, Y. (2019). Big Data analysis in development of personalized medical system. *Procedia Computer Science*, 160, 229-234.
2. Fedushko, S, Ustyianovych, T, Gregus, M. (2020). Real-Time High-Load Infrastructure Transaction Status Output Prediction Using Operational Intelligence and Big Data Technologies. *Electronics*; *9*(4), 668.
3. Kwan, C. (2021). Acer reportedly targeted with $50 million ransomware attack. https://www.zdnet.com/
4. Adler, S. (2020). Incident Of The Week: Garmin Pays $10 Million To Ransomware Hackers Who Rendered Systems Useless. https://www.cshub.com/
5. Tailor, Jinal P., and Ashish D. Patel. (2017). A comprehensive survey: ransomware attacks prevention, monitoring and damage control. *International Journal of Scientific Research 4(VIS)*, 15, 116-121.
6. Ross, B. (2016). Ransomware attacks: detection, prevention and cure. *Network Security*, 9, 5-9.
7. Dudykevych, V., Prokopyshyn, I., Chekurin, V., Opirskyy, I., Lakh, Y., Kret, T., Ivanchenko, Y., Ivanchenko, I. (2019). A multicriterial analysis of the efficiency of conservative information security systems. *Eastern-european journal of enterprise technologies. Information and controlling system*, *3*(9(99)), 6-13.
8. Vasylyshyn, S., Opirskyy, I., Susukailo, V. Analysis of the use of software baits as a means of ensuring information security // 2020 *IEEE 15th International Scientific and Technical Conference on Computer Sciences and Information Technologies*, CSIT 2020 - Proceedings, 2020, 2, pp. 242–245, 9321925
9. Hu, Z., Khokhlachova, Y., Sydorenko, V., Opirskyy, I. (2017). Method for Optimization of Information Security Systems Behavior under Conditions of Influences. I*nternational Journal of Intelligent Systems and Applications (IJISA)*, *9*(12), 46-58.
10. Kharraz, A., Kirda, E. (2017) Redemption: Real-Time Protection Against Ransomware at End-Hosts. In: Dacier M., Bailey M., Polychronakis M., Antonakakis M. (eds) Research in Attacks, Intrusions, and

Defenses. RAID 2017. Lecture Notes in Computer Science, 10453. Springer, Cham. https://doi.org/10.1007/978-3-319-66332-6_5

11. Scaife, N., Carter, H., Traynor, P., & Butler, K. R. (2016, June). Cryptolock (and drop it): stopping ransomware attacks on user data. In *2016 IEEE 36th International Conference on Distributed Computing Systems (ICDCS)* (pp. 303-312). IEEE.
12. Joshi, Y.S., Mahajan, H., Joshi, S.N. et al. (2021). Signature-less ransomware detection and mitigation. *J Comput Virol Hack Tech.* https://doi.org/10.1007/s11416-021-00384-0
13. Poudyal, S., Dasgupta, D. (2020). AI-Powered Ransomware Detection Framework, *In IEEE Symposium Series on Computational Intelligence (SSCI)*, (pp. 1154-1161). https://doi: 10.1109/SSCI47803.2020.9308387.
14. Hagen, C., Dmitrienko, A., Iffländer, L., Jobst, M., & Kounev, S. (2018). Efficient and effective ransomware detection in databases. In *Annu. Comput. Secur. Appl. Conf.(ACSAC)*.
15. Moore, C. (2016). Detecting Ransomware with Honeypot Techniques. In *Cybersecurity and Cyberforensics Conference (CCC)*, 77-81. https://doi: 10.1109/CCC.2016.14.
16. Sethia, V., Jeyasekar, A. (2019). Malware Capturing and Analysis using Dionaea Honeypot. *International Carnahan Conference on Security Technology (ICCST)*, Chennai, India, (pp. 1-4). https://doi: 10.1109/CCST.2019.8888409.
17. Matin, I. M. M., & Rahardjo, B. (2019). Malware detection using honeypot and machine learning. У *2019 7th international conference on cyber and IT service management (CITSM)*. IEEE. https://doi.org/10.1109/citsm47753.2019.8965419
18. tarcisio-marinho (2020). https://github.com/tarcisio-marinho/GonnaCry
19. Panagiotis Drakatos (2017). JavaRansomware. https://github.com/PanagiotisDrakatos/JavaRansomware
20. Leon Voerman (leonv024) (2020). RAASNet. https://github.com/leonv024/RAASNet