Научная статья
УДК 004.6
DOI:10.31854/1813-324X-2022-8-2-91-99

Оценка свойств объектов средств вычислительной техники для обеспечения постинцидентного аудита

Игорь Сергеевич Пантюхин, zevall@ya.ru

Национальный исследовательский университета ИТМО,
Санкт-Петербург, 197101, Российская Федерация

Аннотация: *Исследование компьютерных инцидентов является важным направлением деятельности в области информационной безопасности. В работе рассматривается метод описания свойств объектов средств вычислительной техники для обеспечения постинцидентного аудита. Исследование инцидентов рассматривается с помощью анализа свойств объектов энергозависимой памяти и сетевого трафика. Данные свойства представлены в виде совокупности атрибутов и анализируются путем применения теории графов. Для решения конечной задачи определения и формализации компьютерного инцидента могут применяться различные алгоритмы на графах и совокупности свойств. В работе представлен вычислительный эксперимент постинцидентного аудита средств вычислительной техники на примере определения компьютерного инцидента. Представленный метод минимизирует объемы обрабатываемой информации путем использования для анализа только атрибутов.*

Ключевые слова: *аудит, теория графов, информационная безопасность, вычислительная техника, файловая система, сетевой трафик*

Ссылка для цитирования: Пантюхин И.С. Оценка свойств объектов средств вычислительной техники для обеспечения постинцидентного аудита // Труды учебных заведений связи. 2022. Т. 8. № 2. С. 91–99. DOI:10.31854/1813-324X-2022-8-2-91-99

The Properties of Computer Equipment Objects Evaluation to Ensure Post-Incident Audit

Igor Pantiukhin, zevall@ya.ru

ITMO University,
St. Petersburg, 197101, Russian Federation

Abstract: *The study of computer incidents is an important area of activity in the field of information security. The paper considers a method for describing the properties of objects of computer equipment to ensure post-incident audit. The investigation of incidents is considered by analyzing the properties of objects of volatile memory, non-volatile memory, and network traffic. These properties are presented as a set of attributes and are analyzed by applying graph theory. To solve the final problem of determining and formalizing a computer incident, various algorithms on graphs and sets of properties can be used. The paper presents a computational experiment of post-incident audit of computer equipment by the example of determining a computer incident. The presented method minimizes the amount of information processed by using only attributes for analysis.*

Keywords: *audit, graph, information security, computer technology, file system, network traffic*

For citation: Pantiukhin I. The Properties of Computer Equipment Objects Evaluation to Ensure Post-Incident Audit. *Proc. of Telecom. Universities.* 2022;8(2):91–99. (in Russ.) DOI:10.31854/1813-324X-2022-8-2-9-99

Введение

Стремительное развитие информационных сетей способствует появлению все новых и более сложных компьютерных инцидентов и, следовательно, разработке методов их исследования с целью повышения информационной безопасности [1].

Постинцидентный аудит проводится с помощью анализа свойств средств вычислительной техники. Для обеспечения восстановления событий инцидентов информационной безопасности применяется метод на основе графов, который заключается в построении графа, поиске подграфа из множества данных, относящихся к какому-либо вирусу (прочитанные файлы, отправленные сетевые пакеты, запускаемые процессы и т. д.).

По сравнению с существующими подходами, такими, как построение взаимосвязей между данными аудита [2] и применение поискового алгоритма [3], рассматриваемый подход, основанный на применении теории графов, минимизирует объемы обрабатываемой информации путем использования для анализа только атрибутов [4].

Структура графа

Граф строится от 3 узлов, каждый из которых является корнем дерева-подграфа для соответствующего типа данных: диска, памяти и сети (рисунок 1) [5]. Ребра, связывающие узлы, являются условными, то есть не представляют некоторую связь в анализируемой системе, и используются только для связности построенного графа. Эти ребра являются единственными ребрами в графе без направления, так как в данном случае оно не имеет смысловой основы. В подграфах ребра направлены от некоторого объекта в сторону его составных частей. Например, метаданные файла являются частью файла, а файл является частью папки. Подобная иерархия и формирует структуру подграфов.

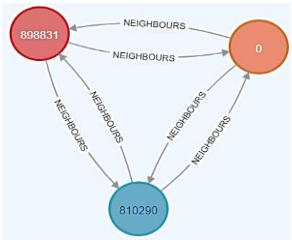

Рис. 1. Структура графа

Fig. 1. Graph Structure

Каждый из узлов подграфов имеет определенный тип (в СУБД Neo4j называемые “Label”) и единственное ассоциированное с ним значение. Типы узлов подграфов включают, например, файл, папку, время создания файла, сетевой пакет, адрес сетевого пакета, процесс в оперативной памяти и т. д. Каждое ребро подграфа имеет тип, отражающий смысл связи в анализируемой системе. Например, файл находится внутри папки, и эта связь обозначена соответствующим ребром.

Подграф файловой системы

Структура подграфа файловой системы иерархическая и состоит из узлов папок (коричневый цвет) и узлов файлов (розовый цвет) (рисунок 2).

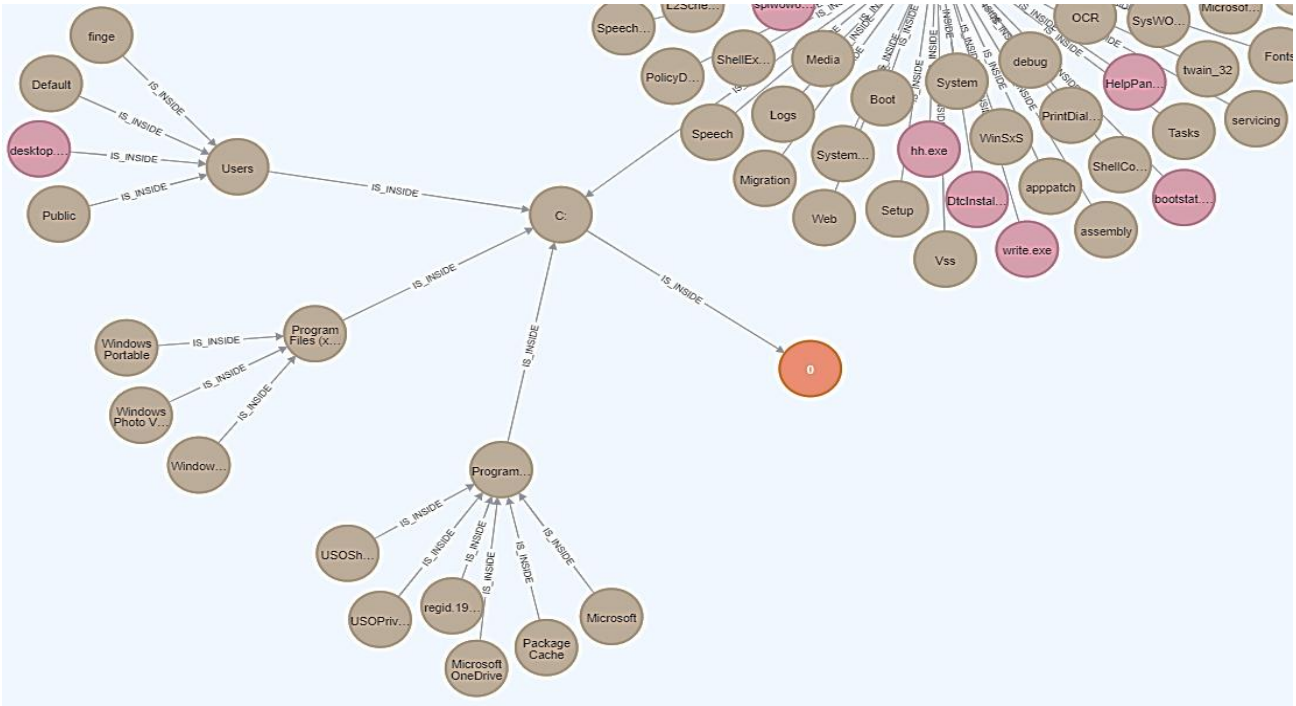

Рис. 2. Структура подграфа

Fig. 2. Subgraph Structure

Путь к конкретному файлу от корня подграфа соответствует этому пути в файловой системе. К каждому узлу файла в графе привязан некоторый набор метаданных: размер, контрольная сумма, время изменения и другие (рисунок 3).

Подграф оперативной памяти

Подграф оперативной памяти состоит из узлов процессов, связанных с корневым узлом (рисунок 4). С каждым узлом процесса связано некоторое количество узлов метаданных: ID родительского процесса, название процесса и другие (рисунок 5). К узлу процесса также могут быть привязаны узлы обработчиков объектов (handles) (рисунок 6), а к последним – метаданные (рисунок 7). Другим типом узлов, привязанных к процессу, являются узлы идентификаторов безопасности (SID, *аббр. от англ.* Security Identifier) (рисунок 8), к которым также привязаны метаданные (рисунок 9).

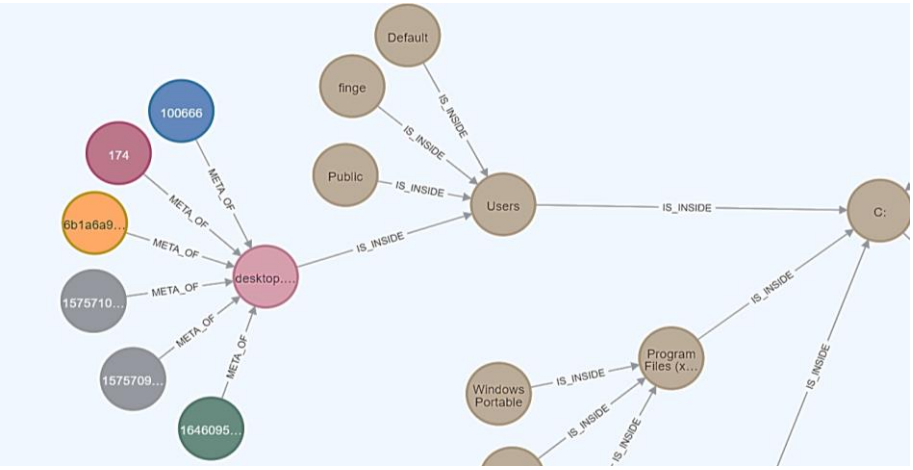

Рис. 3. Набор метаданных

Fig. 3. Metadata Set

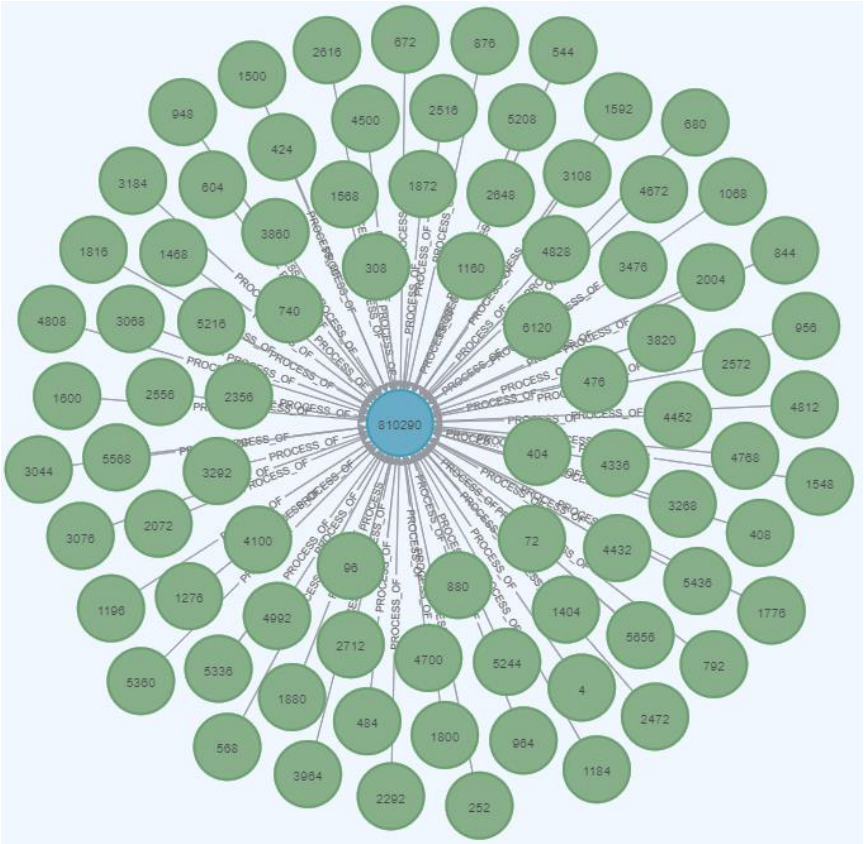

Рис. 4. Подграф оперативной памяти

Fig. 4. RAM Subgraph

Рис. 5. Связь узлов процесса с узлами метаданных

Fig. 5. Linking Process Nodes to Metadata Nodes

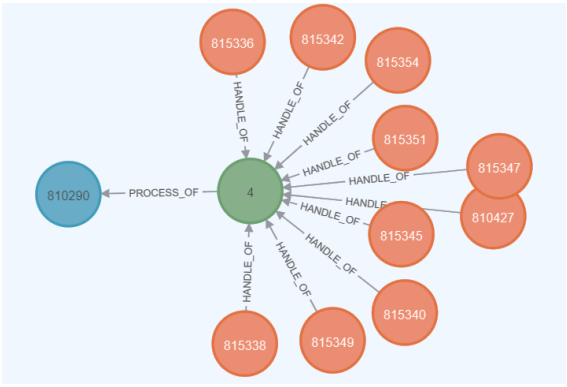

Рис. 6. Связь узлов обработчиков объектов с узлом процесса

Fig. 6. Connection of Object Handler Nodes to the Process Node

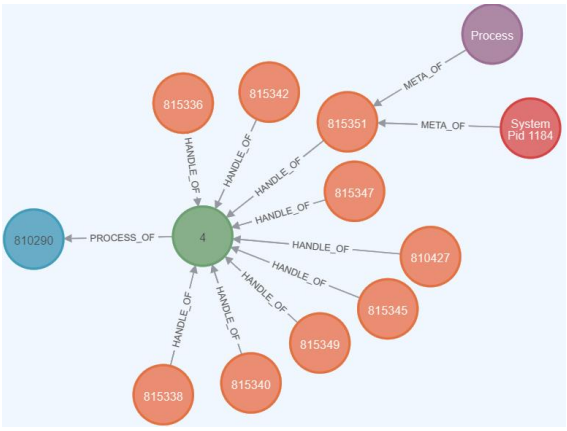

Рис 7. Связь метаданных с узлом обработчика объекта

Fig. 7. Metadata Association with the Object Handler Node

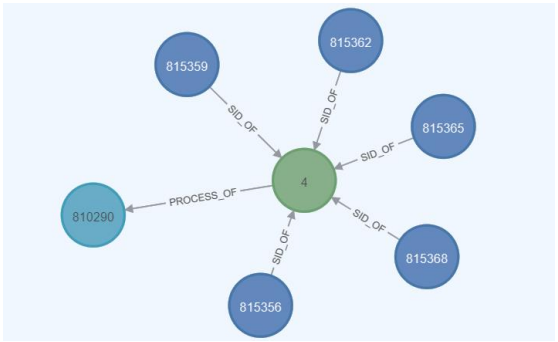

Рис. 8. Узел идентификаторов безопасности

Fig. 8. Security Identifiers Node

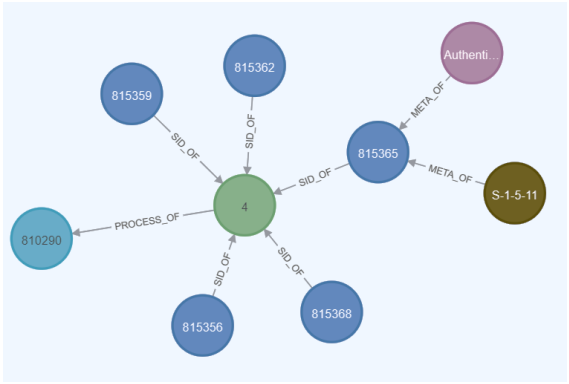

Рис. 9. Связь метаданных с узлом идентификаторов

Fig. 9. Linking Metadata to the Identifiers Node

Подграф сети

Подграф сети состоит из узлов пакетов, связанных с корневым узлом (рисунок 10). К каждому пакету привязаны узлы метаданных: IP отправления, IP назначения, размер пакета и др. (рисунок 11).

Построение графа

Оценка свойств объектов осуществляется путем формирования подграфа на основе совокупности атрибутов объектов. Данный подграф содержит в себе информацию об инциденте и может быть автоматизирован различными графовыми алгоритмами и алгоритмами машинного обучения. В работе представляется принцип поиска и обнаружения прецедента на основе теории графов.

Каждый из трех подграфов строится независимо друг от друга в следующем порядке: диск, память, сеть. Существует возможность строить граф с любыми одним или двумя подграфами.

Для построения графа используются 3 источника данных: запись информации о файловой системе (пути и метаданные всех файлов) в свободном формате, дамп оперативной памяти и PCAP-файл с записью сетевого трафика. Используемое для получения данных ПО не имеет значения.

Поскольку исходные данные не представлены в формате графа (данные файловой системы и данные сетевого трафика представляют собой список объектов, а дамп оперативной памяти не имеет никакой четкой структуры), необходима реструктуризация этих данных. В связи с этим построение графа происходит в два этапа: обработка исходных данных (представление в формате, легко конвертируемом в граф) и запись в базу данных.

Реализация этапа обработки данных

Данные всех трех типов читаются из файлов и преобразовываются в промежуточный формат для сохранения. Из-за специфики каждого типа данных этот формат несколько разнится между типами данных.

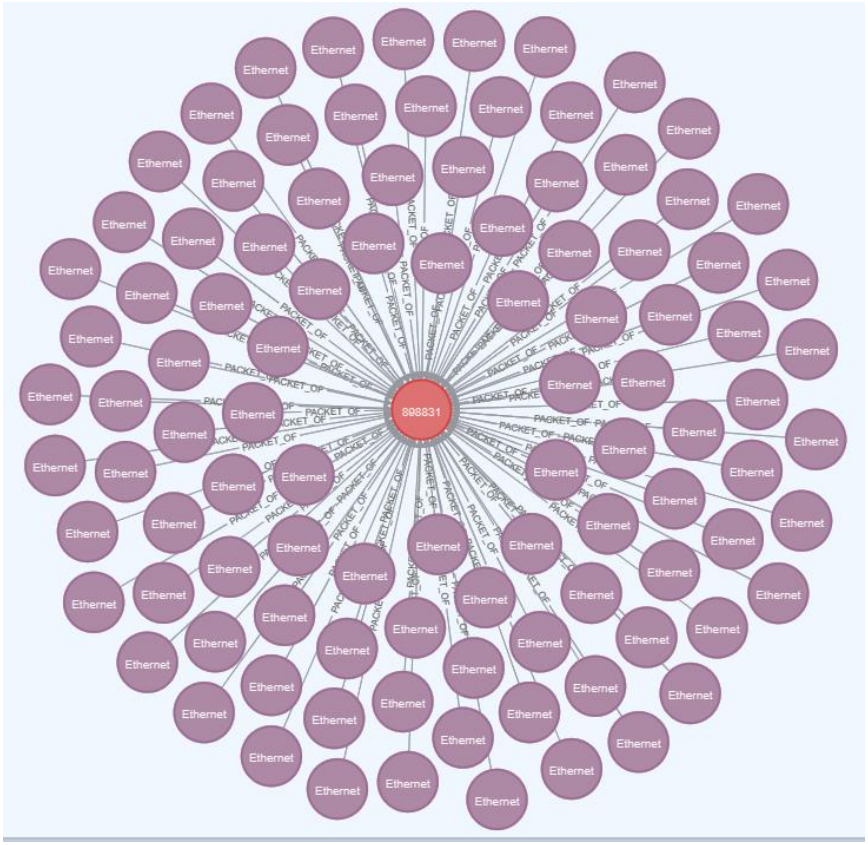

Рис. 10. Подграф сети

Fig. 10. Network Subgraph

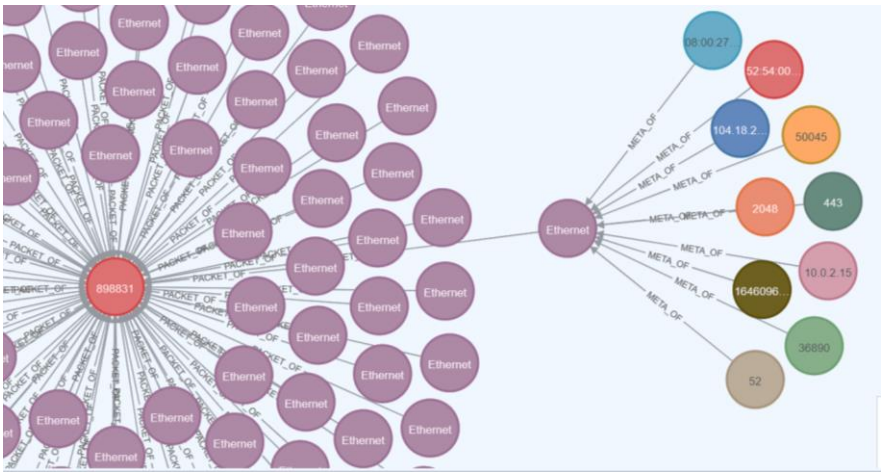

Рис. 11. Связь узла метаданных с каждым пакетом подграфа сети

Fig. 11. The Association of the Metadata Node with Each Packet of the Network Subgraph

Данные файловой системы не проходят никакой специальной предобработки, так как изначально генерируются специально разработанным модулем в нужном формате. Их обработка ограничивается прочтением JSON-файла.

Для парсинга дампов оперативной памяти используется Python-библиотека Volatility 3. С ее помощью из дампа оперативной памяти извлекается список процессов и связанные с ними метаданные. Используются следующие плагины: windows.pslist, windows.psscan, windows.handles, windows.getsids. Каждый плагин обернут в класс, который предоставляет интерфейс для запуска плагина и записи возвращаемых данных в единую для всех плагинов структуру. При запуске каждого плагина данные в структуре дополняются, что позволяет получить более полный их массив, чем при использовании любого из плагинов отдельно. Названная выше структура с математической точки зрения является деревом, что позволяет легко создать из нее граф на этапе записи данных.

Для парсинга записей сетевого трафика (файлы pcap) используется Python-библиотека Scapy. С ее помощью из записи сетевого трафика извлекается список сетевых пакетов. Подробные метаданные собираются из пакетов, использующих протоколы IP, TCP, UDP. Все данные сохраняются в промежуточную структуру для записи.

Реализация этапа записи данных

Для создания графа в СУБД Neo4j использована Python-библиотека neomodel. Перед записью в СУБД данных подграфов создаются 3 корневых узла для подграфов и 3 ненаправленных ребра между ними. После создания корневых узлов в СУБД поочередно записываются каждый из подграфов. Порядок записи отдельных узлов в подграфах отличается из-за их разной структуры. В подграфе файловой системы из-за произвольной глубины дерева для каждого файла реализован поиск пути в уже созданном подграфе. В случае, если этот путь найден, узел файла связывается с найденным узлом родительской папки, в противном случае весь путь или недостающая его часть создаются, начиная с папок, ближайших к корню файловой системы. Для подграфов оперативной памяти и сети процесс записи с СУБД происходит поочередно для каждой ветви дерева из-за его заранее известной глубины.

Вычислительный эксперимент

Для проведения эксперимента использовалась виртуальная машина, запущенная в Oracle VirtualBox, с предустановленной операционной системой Microsoft Windows 10 Home на ПК с одноядерным процессором и 2 Гб оперативной памяти. Данная виртуальная машина была заражена вирусом из семейства RedLine путем запуска вредоносного исполняемого файла. Для записи сетевого трафика было использовано ПО WireShark [6–8], запись велась в течение 3 минут с момента запуска вредоносного исполняемого файла, результат сохранен в файл формата pcap. Для создания дампа оперативной памяти использовалась утилита VirtualBox debugvm, дамп создан через одну минуту после запуска вредоносного файла, результат сохранен в файл формата elf. Для записи состояния файловой системы использован Python-скрипт, описанный выше, результат сохранен в файл формата json. На основе данных трех файлов был составлен граф в базе данных Neo4j [9–11] с 999553 узлами 36 типов и 999556 ребрами 7 типов. Для поиска подграфов использовано ПО Neo4j Browser и запросы на языке Cypher.

Согласно существующему поведенческому анализу вируса [12–15], он производит, среди прочих, доступ к системному файлу AppLaunch.exe и выполняется в течение промежутка времени около 2 секунд. Благодаря такому короткому времени выполнения можно предположить, что значительная часть событий, произошедших на компьютере в этот промежуток времени, имеет непосредственное отношение к действию вируса. Это позволяет произвести анализ графа на основании конкретного момента работы вируса. Для определения этого времени было решено найти файл AppLaunch.exe в графе и извлечь время доступа к нему из метаданных (рисунок 12).

На основе известного времени работы вируса, извлеченного из метаданных файла AppLaunch.exe, и информации о том, что суммарное выполнение программы вируса занимает около 2 секунд, был выбран промежуток времени 3 секунды до и после известного времени, который должен гарантированно покрыть промежуток работы вируса независимо от того, в какой момент этой работы был произведен доступ к файлу AppLaunch.exe. На этом промежутке был проведен поиск всех файлов, к которым производится доступ (рисунок 13) и всех отправленных и принятых сетевых пакетов (рисунок 14).

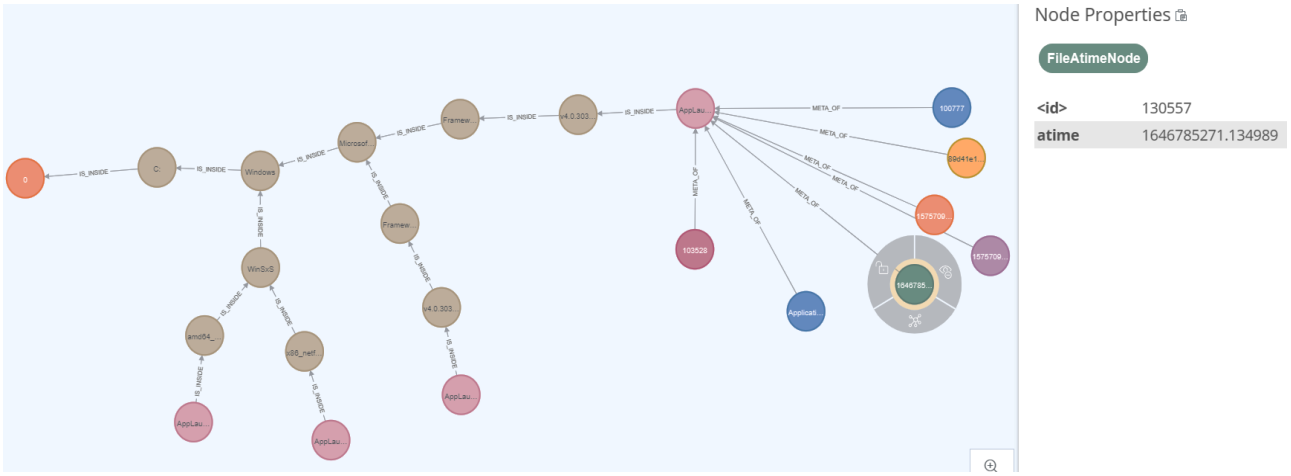

Рис 12. Нахождение времени запуска вируса

Fig. 12. Finding the Virus Launch Time

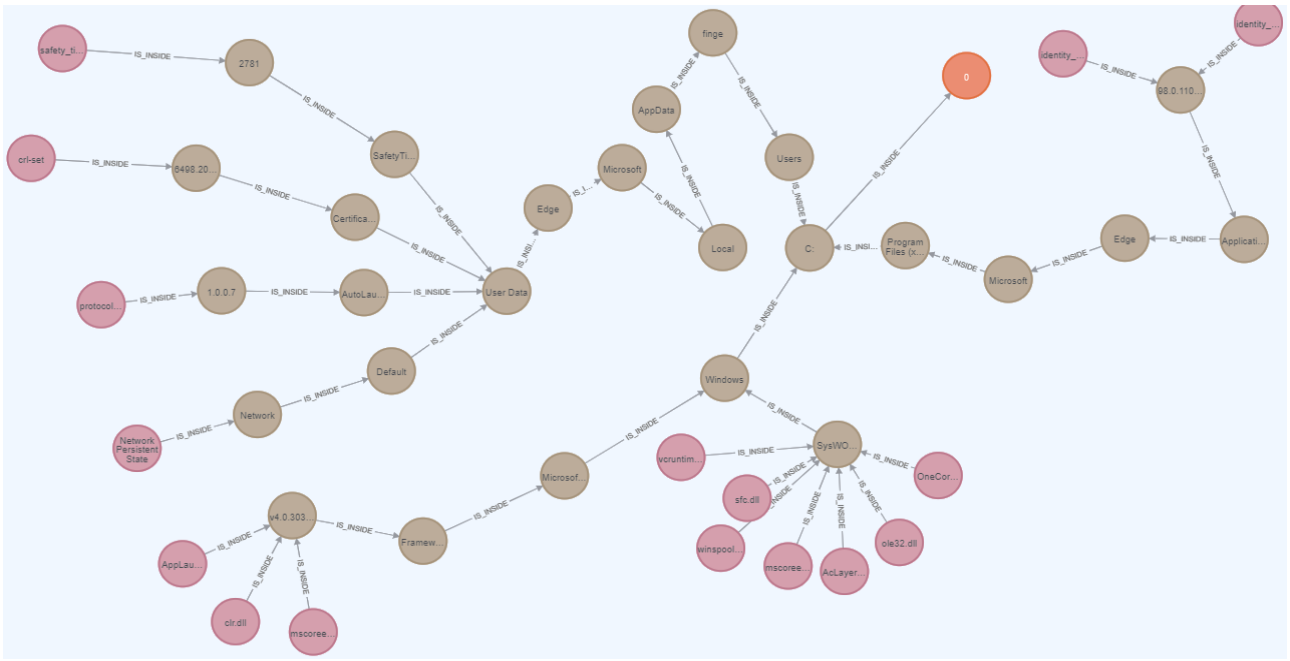

Рис. 13. Найденный подграф файловой системы

Fig. 13. The Found File System Subgraph

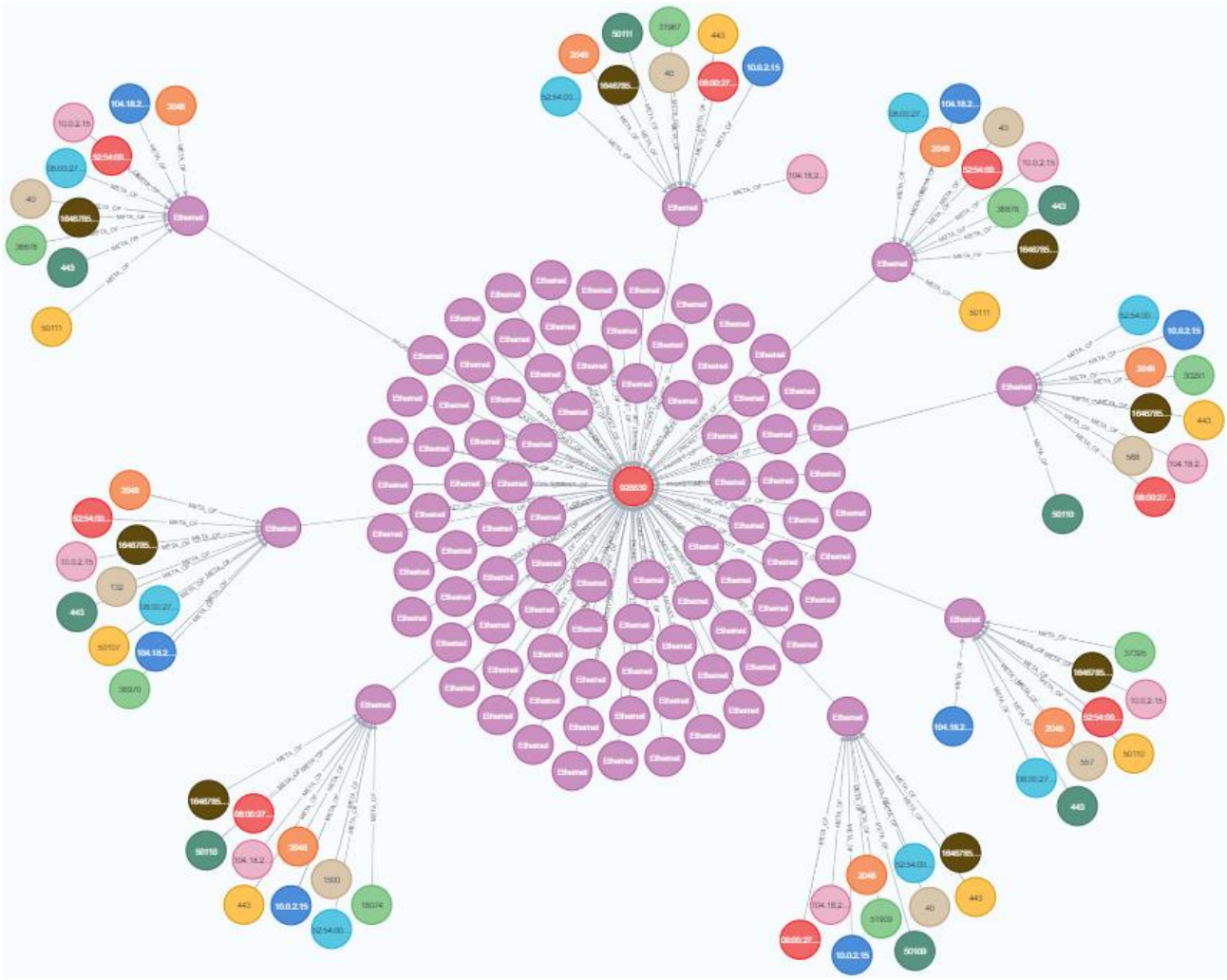

Рис 14. Найденный подграф сетевого трафика

Fig. 14. The Found Subgraph of Network Traffic

На основе найденного подграфа файловой системы обнаружен список файлов с высокой вероятностью затронутых действием вируса, а на основе подграфа сетевого трафика – MAC- и IP-адреса, на которые вирусом были отправлены данные.

Заключение

Использование метода постинцидентного аудита на основе графов позволяет увеличить скорость и эффективность проведения анализа средств вычислительной техники.

В работе показа возможность описания взаимосвязи между атрибутами объектов за счет применения основных положений теории графов. Теория графов позволяет установить взаимосвязи между состояниями объектов в формате атрибутов и их значений согласно алгоритму. Визуализация, в свою очередь, позволяет повысить информативность обнаруженного инцидента.

Метод может применяться в решении задач определения взаимоотношений между объектами данных различных устройств, на которых произошел компьютерный инцидент.

Экспериментальным путем было установлено, что за счет анализа исключительно атрибутов наблюдается повышение информативности его результатов, а также минимизация ошибки аудита. Также стоит отметить тот факт, что при использовании данного метода реализуется возможность визуализации всего процесса анализа сетевого инцидента.

По результатам выполненного эксперимента можно утверждать, что использование метода на основе графов имеет достаточно высокий потенциал применения и дальнейшего развития в сфере постинцидентного аудита для анализа различных событий с целью обеспечения кибербезопасности.

Список источников

1. Деров Е. Учитывая быстрое развитие и рост популярности технологий Big Data, есть причина задуматься о целесообразности их применения при расследовании инцидентов ИБ // IT-компания КАБЕСТ. 2014. URL: http://kabest.ru/press/news/754/index.php?print=Y (дата обращения 15.04.2016)
2. Бессонова Е.Е., Зикратов И.А., Росков В.Ю. Анализ способов идентификации пользователя в сети интернет // Научно-технический вестник информационных технологий, механики и оптики. 2012. № 6(82). С. 128–129.
3. Бессонова Е.Е., Зикратов И.А., Колесников Ю.Л., Росков В.Ю. Способ идентификации пользователя в сети интернет // Научно-технический вестник информационных технологий, механики и оптики. 2012. № 3(79). С. 133–137.
4. Пантюхин И.С., Зикратов И.А., Левина А.Б. Метод проведения постинцидентного внутреннего аудита средств вычислительной техники на основе графов // Научно-технический вестник информационных технологий, механики и оптики. 2016. Т. 16. № 3. С. 506–512. DOI:10.17586/2226-1494-2016-16-3-506-512
5. Кристофидес Н. Теория графов. Алгоритмический подход. М.: Мир, 1978. 432 с.
6. Orebaugh A., Ramirez G., Beale J. Wireshark & Ethereal Network Protocol Analyzer Toolkit. Elsevier, 2006.
7. Wang S., Xu D.S., Y.S. Analysis and application of Wireshark in TCP/IP protocol teaching // IEEE International Conference on E-Health Networking, Digital Ecosystems and Technologies (EDT, Shenzhen, China, 17–18 April 2010). IEEE: 2010. PP. 269–272. DOI:10.1109/EDT.2010.5496372
8. Ndatinya V., Xiao Z., Manepalli V.R., Meng K., Xiao Y. Network forensics analysis using Wireshark // International Journal of Security and Networks. 2015. Vol. 10. No. 2. PP. 91–106. DOI:10.1504/IJSN.2015.070421
9. Miller J.J. Graph Database Applications and Concepts with Neo4j // Proceedings of the Southern Association for Information Systems Conference (Atlanta, USA, 23rd–24th March 2013). Association for Information Systems, 2013. PP. 141–147.
10. Bruggen R.V. Learning Neo4j. 2014. P. 222.
11. Guia J., Soares V.G., Bernardino J. Graph Databases: Neo4j Analysis // Proceedings of the 19th International Conference on Enterprise Information Systems (ICEIS, Porto, Portugal). SciTePress: 2017. Vol. 1. PP. 351–356. DOI:10.5220/0006356003510356
12. Збицкий П.В. Функциональная сигнатура компьютерных вирусов // Доклады ТУСУРа. 2009. № 1(16). С. 75–76.
13. Татаринов А.А., Болдырихин Н.В. Анализ методов обнаружения вредоносного программного обеспечения на основе поведенческих признаков // Сборник избранных статей Всероссийской научно-практической конференции «Национальная безопасность России: актуальные аспекты» (Санкт-Петербург, Россия, 29 марта 2020). СПб: Частное научно-образовательное учреждение дополнительного профессионального образования Гуманитарный национальный исследовательский институт «НАЦРАЗВИТИЕ», 2020. С. 18–22.
14. Назаров А.В., Марьенков А.Н., Калиев А.Б. Выявление поведенческих признаков работы вируса-шифровальщика на основе анализа изменений значений параметров компьютерной системы // Прикаспийский журнал: управление и высокие технологии. 2018. № 1(41). С. 196–204.
15. Назаров А.В., Марьенков А.Н. Проблема выявления признаков вируса-шифровальщика в работе компьютерной системы // VII Всероссийская заочная Интернет-конференция «Проблемы информационной безопасности» (Ростов-на-Дону, Россия, 20–21 февраля 2018). Ростов: Федеральное государственное образовательное учреждение высшего образования «Ростовский государственный экономический университет (РИНХ)» Ростовское региональное отделение вольного экономического общества России, 2018. С. 10–14.

References

1. Derov E. Considering the Rapid Development and Growing Popularity of Big Data Technologies, There is a reason to Think about the Expediency of Their Use in the Investigation of Information Security Incidents. *IT-kompaniia KABEST.* 2014. Available from: http://kabest.ru/press/news/754/index.php?print=Y [Accessed 15th April 2016]. (in Russ.)
2. Bessonova E., Zikratov I., Roskov V. Analysis of Internet User Identification Methods. *Scientific and Technical Journal of Information Technologies, Mechanics and Optics*. 2012;6(82):128–129. (in Russ.)
3. Bessonova E., Zikratov I., Kolesnikov Yu., Roskov V. Internet User Identification Method. *Scientific and Technical Bulletin of information Technologies, Mechanics and Optics*. 2012;3(79):133–137. (in Russ.)
4. Pantiukhin I.S., Zikratov I.A., Levina A.B. Method of conducting post-incident internal audit of computer equipment based on graphs. *Scientific and Technical Bulletin of Information Technologies, Mechanics and Optics*. 2016;16(3):506–512. (in Russ.) DOI:10.17586/2226-1494-2016-16-3-506-512
5. Christophides N. *Graph Theory. Algorithmic Approach*. Moscow: Mir Publ.; 1978. 432 p. (in Russ.)
6. Orebaugh A., Ramirez G., Beale J. *Wireshark & Ethereal Network Protocol Analyzer Toolkit*. Elsevier; 2006.
7. Wang S., Xu D.S., Y.S. Analysis and application of Wireshark in TCP/IP protocol teaching. *IEEE International Conference on E-Health Networking, Digital Ecosystems and Technologies, EDT, 17–18 April 2010, Shenzhen, China*. IEEE, 2010. p.269–272. DOI:10.1109/EDT.2010.5496372
8. Ndatinya V., Xiao Z., Manepalli V.R., Meng K., Xiao Y. Network forensics analysis using Wireshark. *International Journal of Security and Networks*. 2015;10(2):91–106. DOI:10.1504/IJSN.2015.070421
9. Miller J.J. Graph Database Applications and Concepts with Neo4j. *Proceedings of the Southern Association for Information Systems Conference, 23–24 March 2013, Atlanta, USA*. Association for Information Systems; 2013. p.141–147.
10. Bruggen R.V. *Learning Neo4j*. 2014. p.222.
11. Guia J., Soares V.G., Bernardino J. Graph Databases: Neo4j Analysis. *Proceedings of the 19th International Conference on Enterprise Information Systems, ICEIS, Porto, Portugal*. SciTePress: 2017. vol.1. p.351–356. DOI:10.5220/0006356 003510356
12. Zbitsky P.V. Functional Signature of Computer Viruses. *Reports of TUSUR*. 2009;1(16):75–76. (in Russ.)
13. Tatarinov A.A., Boldyrikhin N.V. Analysis of Methods for Detecting Malicious Software Based on Behavioral Signs. *Proceedings of the All-Russian Scientific and Practical Conference on National Security of Russia: Current Aspects, 29 March 2020, St. Petersburg, Russia*. St. Petersburg: Private Scientific and Educational Institution of Additional Professional Education Humanitarian National Research Institute "NATIONAL RAZVITIE" Publ.; 2020. p.18–22. (in Russ.)
14. Nazarov A.V., Marienkov A.N., Kaliev A.B. Identification of behavioral signs of the cipher virus based on the analysis of changes in the values of computer system parameters. *Caspian Journal: Management and High Technologies*. 2018;1(41): 196–204. (in Russ.)
15. Nazarov A.V., Marienkov A.N. The problem of detecting signs of a cryptographer virus in the operation of a computer system. *Proceedings of the VII All-Russian Correspondence Internet Conference on Problems of Information Security, 20–21 February 2018, Rostov-on-Don, Russia.* Rostov: Federal State Educational Institution of Higher Education "Rostov State Economic University (RINH)" Rostov Regional Branch of the Free Economic Society of Russia Publ.; 2018. p.10–14. (in Russ.)

Статья поступила в редакцию 11.03.2022; одобрена после рецензирования 11.05.2022; принята к публикации 23.05.2022.
The article was submitted 11.03.2022; approved after reviewing 11.05.2022; accepted for publication 23.05.2022.

Информация об авторе:

ПАНТЮХИН Игорь Сергеевич
ассистент факультета инфокоммуникационных технологий Национального исследовательского университета ИТМО
https://orcid.org/0000-0002-3946-6057