УДК [004.056](https://teacode.com/online/udc/00/004.056.html).5; 004.738.5

Группа специальностей ВАК: 2.3.6.

**РЕАЛИЗАЦИЯ КЛИЕНТСКОГО ПРИЛОЖЕНИЯ ДЛЯ ВЫЯВЛЕНИЯ АТАК ТИПА "ЧЕЛОВЕК ПОСЕРЕДИНЕ" В КАЧЕСТВЕ СИСТЕМНОЙ СЛУЖБЫ WINDOWS.**

***Степанов П.П.[[1]](#footnote-1), Никонова Г.В.[[2]](#footnote-2)***

***Цель исследования*** *заключается в разработке клиент-серверного приложения для выявления и оперативного реагирования на атаки типа "человек по середине". Клиентское приложение предназначено для тонкой настройки политики безопасности для разных узлов сети.*

***Методом проведения*** *исследования является разработка сетевой утилиты средствами объектно-ориентированного программирования на языке C#, с реализацией функционала программного обеспечения на уровне классов и методов. Предложен метод разработки функционала на максимально абстрактном уровне – уровне контрактов или интерфейсов (interface), с описанием сигнатуры методов, которые обязан реализовывать класс. Предложен метод использования DI (Dependency injection), что уменьшает связность кода и делает его более гибким для изменений и более легким для написание модульных тестов (Unit*).

***В результате исследования*** *была разработана утилита, реализованная как системная служба Windows, которая анализирует текущую информацию узла сети. Данная утилита собирает такие параметры как адрес DHCP сервера, адреса DNS серверов, адрес шлюза, IP адрес текущей машины, имя текущей машины ARP таблицу и таблицу маршрутизации. Параметры сети передаются для дальнейшего анализа и реагирования на инциденты информационной безопасности.*

***Практическая ценность –*** *приведена практическая разработка клиентского приложения утилиты. Описана реализация системной службы Windows на языке C#, которая передает данные для выявления атак “человек по середине", что в свою очередь позволит выявлять и анализировать состояние всех узлов сети (например, компьютеров одной организации) на предмет атак “человек по середине", таких как: MAC spoofing, ARP spoofing, Evil Twin, DHCP spoofing, ICMP redirect, DNS spoofing, конфигурирование роутера в результате взлома. Также приложение пишет системные логи в Windows.*

*Ключевые слова: Windows service, C#, утилита, ARP протокол, DHCP сервер, DNS сервер, ARP-spoofing, перехват трафика.*

# **ВВЕДЕНИЕ**

В настоящее время распределенные технологии и реализующее их программное обеспечение приобретают все большее значение для повышения надежности пересылки и сохранение конфиденциальности пересылаемых сообщений. Современные реалии требуют постоянного повышения безопасности и эффективности этих систем и диктуют задачи, которые невозможно решить без использования научного подхода. В связи с тем, что современные линии связи могут иметь огромную протяженность, сейчас они тянутся на сотни тысяч километров, в результате можно реализовать физическое подключение к каналу, воздействовать на каналы, нарушая функционирование системы [1-5].

В целом можно сказать, что проблема повышения эффективности и надежности передачи данных в настоящее время очень актуальна и важность проблемы возрастает прямо пропорционально развитию сетевых технологий [6]. Это в первую очередь необходимость решать проблемы, связанные с уязвимостью компьютерных сетей, таких как получение несанкционированного доступ к системе на логическом и физическом уровне, а также несанкционированный доступ в пределах локальной сети или за пределами локальной сети.

**Теория**

# Серьезной угрозой при передаче данных являются сетевые атаки [7-14].

В целом алгоритм сетевой атаки можно разделить на следующие этапы:

1) Анализ структуры сети, топологии сети, определение открытых портов, версии программного обеспечения и операционной системы.

2) Определение возможности проведения сетевых атак на основе обнаруженных уязвимостей и наличия определенных сервисов на атакуемом хосте.

3) На основе обнаруженных уязвимостей, выполнение самой атаки.

4) Анализ полученных результатов.

5) Защита от несанкционированного доступа (создание учетной записи, открытие порта, открытие соединения).

6) Скрытие информации об атаке (очистка журналов, закрытие портов).

Если атака будет успешной, злоумышленник нанесет ущерб, похитив данные или выведя из строя атакованный хост [9-16].

Класс атак на протоколы сетевого обмена с использованием логического доступа может быть реализован путем получения доступа к персональным компьютерам и серверам локальной сети, а также к телекоммуникационному оборудованию.

Помимо такого вида есть сетевые атаки, направленные на информационный обмен данными, поскольку информационные протоколы могут иметь ряд недостатков, а также из-за неправильной конфигурации аппаратного и программного обеспечения злоумышленник может получить несанкционированный доступ, а затем осуществить сетевую атаку [12-16].

После успешного получения доступа, необходимого для проведения сетевой атаки, злоумышленник имеет возможность для перехвата данных и служебной информации, передаваемых по сетям, которые впоследствии могут быть использованы для их модификации, замены исходных данных ложными, перенаправления пакетов [17-19] и т.д. Более того, можно манипулировать не только пользовательскими данными, но и служебной информацией, передаваемой сетевыми узлами, например, таблицами маршрутизации ARP-таблицами [9-16].

Лучшим методом предотвращения сетевых атак является использование протоколов, поддерживающих шифрование данных, таких как SSL.

В качестве противодействия атакам на протоколы сетевого обмена применяются логические способы защиты информации это: проверка подлинности обеих сторон; шифрование передаваемой информации как симметричными, так и асимметричными криптографическими системами [11]; проверка целостности принимаемых данных; защита от удаления и изменения данных [7, 11].

Другим способом защиты данных, передаваемых по открытым каналам связи, является создание защищенных виртуальных сетей (VPN).

Самый простой способ осуществить атаку "человек посередине" — это метод, основанный на изменении IP-адреса на компьютере жертвы. Это изменение может быть внесено локально, с помощью программного обеспечения для удаленного аминистрирования или с помощью вирусного программного обеспечения. Чтобы изменить IP-адрес в Unix, вам нужно удалить старый адрес шлюза, затем добавить новый адрес, в системах Windows вам нужно открыть окно свойств протокола TCP/IP и заполнить требуемое поле [4-6]. По сути это процедура изменения шлюза.

**Решение**

Сетевые атаки типа «человек посередине» [MITM](https://ru.wikipedia.org/wiki/MITM) [8-16] (англ. Man in the middle) являются серьезно проблемой информационной безопасности. Основная идея этих атак заключается в том, что трафик не идет напрямую между двумя узлами сети, а проходит через злоумышленника. Злоумышленник может просматривать и модифицировать проходящий через него трафик. Возникла идея в реализации клиент-серверной утилиты для выявления и оперативного реагирования на подобные атаки (рисунок 1). Данная утилита должная состоять из клиентского приложения (которое будет устанавливаться на все компьютеры), серверного приложения которая будет анализировать поступающие данные и сервера базы данных где будут храниться валидные настройки узлов сети, такие как адреса DHCP серверов, адреса DNS серверов, адреса шлюзов. В статье приведено описание разработанного клиентского приложения под операционную систему Windows. Которое будет работать как системная служба и отправлять данные о настройках сети, которые могут быть компрометированы про проведение [MITM](https://ru.wikipedia.org/wiki/MITM) атак [19-25].

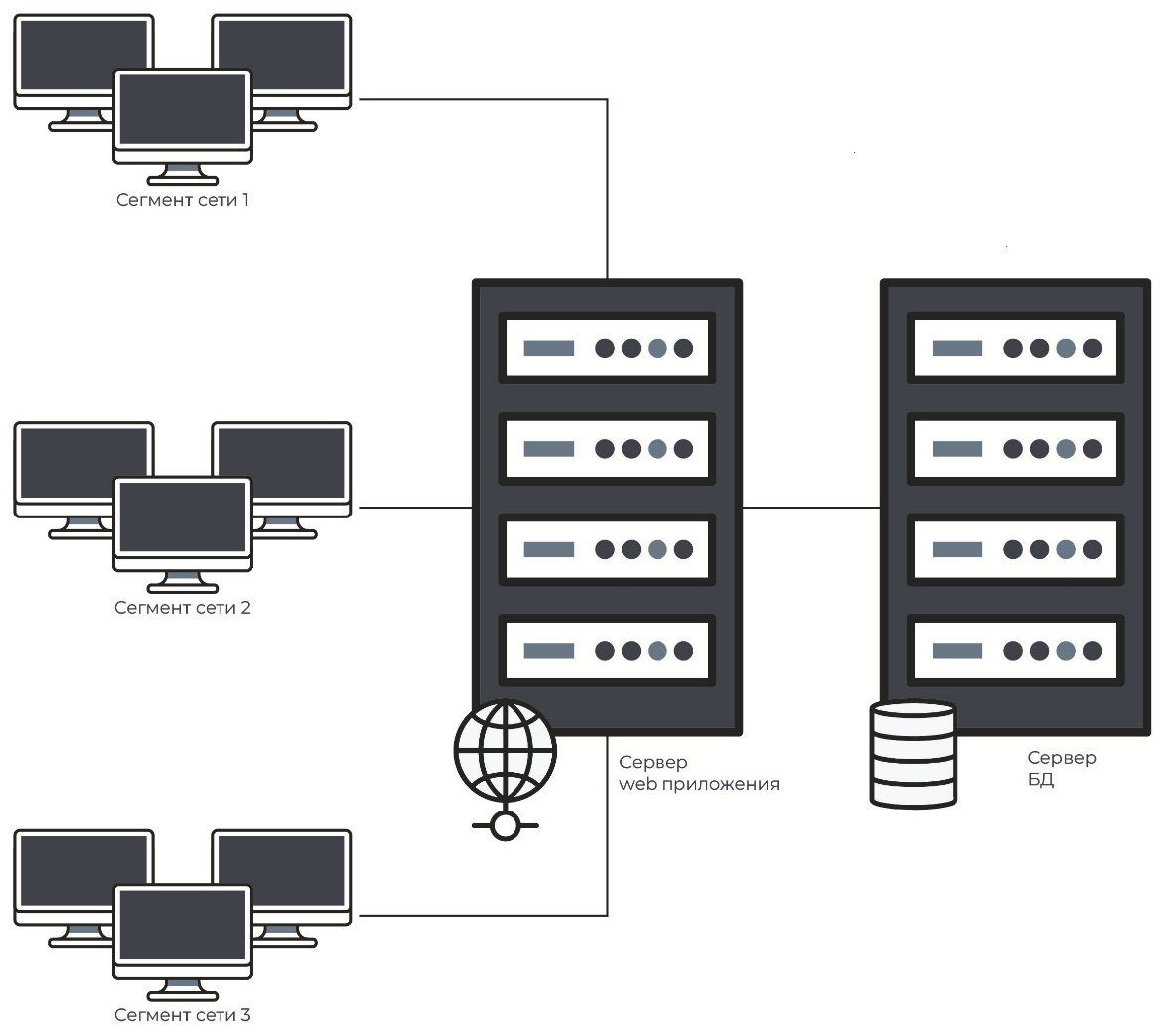


Рисунок 1 – Схема работы приложения

## **Описание решения (Solution) NetworkMonitor**

На данный момент (Solution) NetworkMonitor состит из 4 проектов:

1. NetworkMonitor.Common – Сборка содержит в себе базовые вещи, которые используются во всех других сборках (рисунок 2):

Constants - константы такие как сообщения оба ошибках, сообщения для логирования, значения по умолчанию.

Dto – (Data Transfer Object) - данные для передачи от клиента серверу.

Enums - содержит перечисления.

Exceptions – содержит исключения, которые могут возникать в ходе работы.

ExtensionMethods – содержит методы расширения для базовый классов .net, например для NetworkInterface.

Interfaces – содержит описание контрактов, которые необходимо реализовать для каждой платформы (Windows, Linux).

Settings - содержит классы с настройками.

1. NetworkMonitor.Implementation – Сборка содержит в себе реализации интерфейсов для различный платформ.
2. NetworkMonitor.Tests – Сборка содержит в себе Unit тесты для проверки корректной работы различный модулей. Папка Builders – содержи в себе реализации GoF паттерна Builder для создания сущностей, которые будут использовать в Unit тестах.
3. NetworkMonitor.WindowsService – Сборка содержит в себе клиент для системы Windows, которая может быть установлена в систему в качестве системной службы (Windows Services). Также здесь настраивается внедрение нужный зависимостей DI (Dependency injection) необходимых для работы именно под операционной системой Windows.

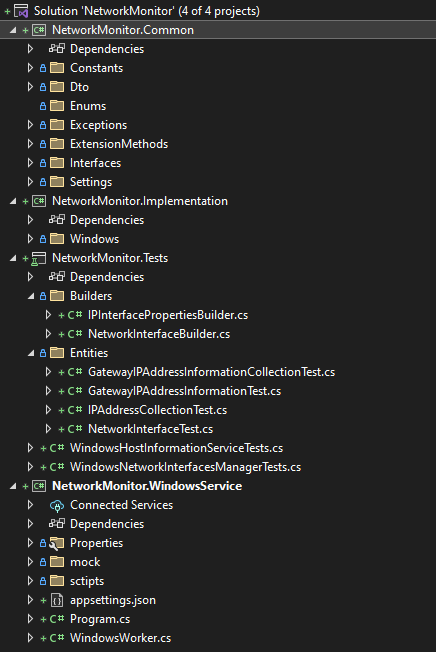


Рисунок 2 – Схема решения (Solution)

## **Описание интерфейсов и Dto.**

Прежде чем приступить к реализации функционала ПО на уровне классов и методов необходимо продумать функционал на максимально абстрактном уровне – уровне контрактов или интерфейсов (interface), прописав сигнатуры методов, которые обязан реализовывать класс. Это нужно для дальнейшего использования DI (Dependency injection) что уменьшает связность кода и делает его более гибким для изменений и более легким для написание модульных тестов (Unit).

* IHostInformationService – Интерфейс содержит в себе контракт на получения информации о узле сети (рисунок 3), и обязует класс содержать в себе следующие методы:
* GetDhcp() - получение IP адреса DHCP сервера. При проведении ряда [MITM](https://ru.wikipedia.org/wiki/MITM) таких как DHCP spoofing [1] или Evil Twin [2] он может быть изменен.
* GetGateway() - получение IP адреса шлюза по-умолчанию. Это тот узел сети, через который идет весь трафик по-умолчанию. Шлюз может быть изменен в результате взлома и переконфигурирования роутера, атак ICMP redirect [3] или DHCP spoofing[1].
* GetHostName() - получение имени машины.
* GetPv4Address() - получение IP адреса машины.
* GetDnsList() - получение списка IP адресов DNS. Если были изменены адреса DNS серверов злоумышленник может перенаправить пользователя на поддельный сайт, который выглядит как точная копия легитимного и запросить у пользования ввести логин и пароль тут не спасет для шифрование посредством SSL [4].
* GetTracertTable() - получение таблицы маршрутизации. Нужно для дальнейшего анализа и выявления атак ARP spoofing [5,6] и MAC spoofing. При выполнение этих атак на атакуемой машине будет аномалия, а именно трафик вместо того, чтобы сразу пойти на шлюз полмолчания будет проходить еще через один узел сети [7-9].
* GetArpTable() - получение ARP таблицы содержащий коллекцию Dto Host(рисунок 3). Нужно для дальнейшего анализа и выявления атак ARP spoofing и mac spoofing. При выполнение этих атак на атакуемой машине будет аномалия, а именно у двух узлов с разными IP адресами будут одинаковые MAC адреса [7-9].
* GetHostInformation() - Получение информации об узле сети в виде Dto HostInformation(рисунок 5).

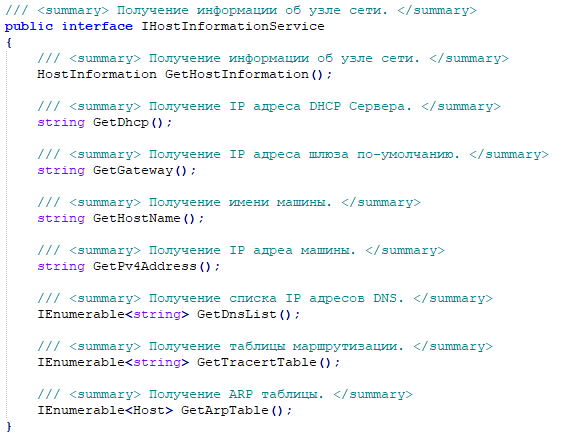


Рисунок 3 – Интерфейс IHostInformationService

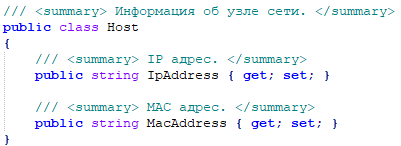


Рисунок 4 – Dto Host

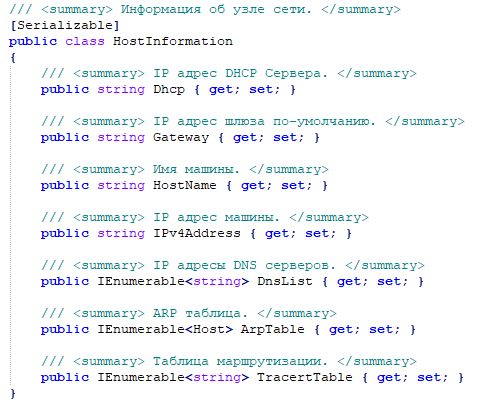


Рисунок 5 – Dto HostInformation

IWindowsManager – Интерес содержит в себе контракт для работы со специфическими вещями для ОС Windows который нет в стандартный библиотека .net такие как работа с протоколами ARP и ICMP (рисунок 6):

GetArpTable() - получение ARP таблицы для текущего узла сети, содержащий коллекцию Dto Host(рисунок 4)

GetTracertTable() - получение таблицы маршрутизации для текущего узла сети, до шлюза по умолчанию.

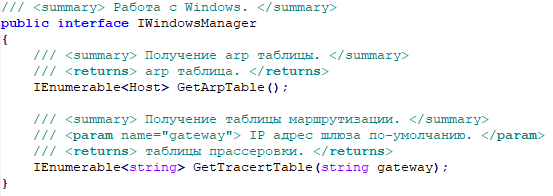


Рисунок 6 – Интерфейс IWindowsManager

IHttpClient – Интерес содержит в себе контракт для отправки, данный на сервер посредствам протокола HTTP (рисунок 7). Содержит в себе сигнатуру метода SendHostInformation для отправики Dto HostInformation с информацией о текущем узле сети на сервер для дальнейшей валидации.

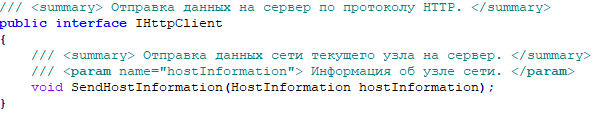


Рисунок 7 – Интерфейс IWindowsManager

## **Описание интерфейсов для Windows.**

WindowsHttpClientMock – класс реализующий интерфейс IHttpClient (рисунок 8), так как сервер еще не реализован, класс представляет собой заглушку для тестирования работоспособности сервиса. Класс WindowsHttpClientMock содержит реализацию метода SendHostInformation (рисунок 7), данный метод серилизует экземпляр класса HostInformation в формат json и сохраняет его в файл с имением текущей даты и времени.

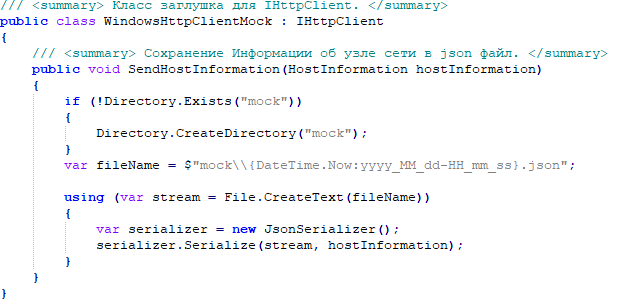


Рисунок 8 – Класс WindowsHttpClientMock

WindowsCmdManager – класс реализующий интерфейс IWindowsManager (рисунок 9). WindowsCmdManager – пример реализации интрефейса IWindowsManager с помощь стандартный утилит Windows, а именно arp.exe и tracert.exe. В методах GetTracertTable и GetArpTable происходит вызов этих утилит с перенаправлением потока вывода на методы ParseTracert и ParseArp и дальнейшего парсинга результата работы утилит arp.exe и tracert.exe (рисунок 10).



Рисунок 9 – Класс WindowsCmdManager

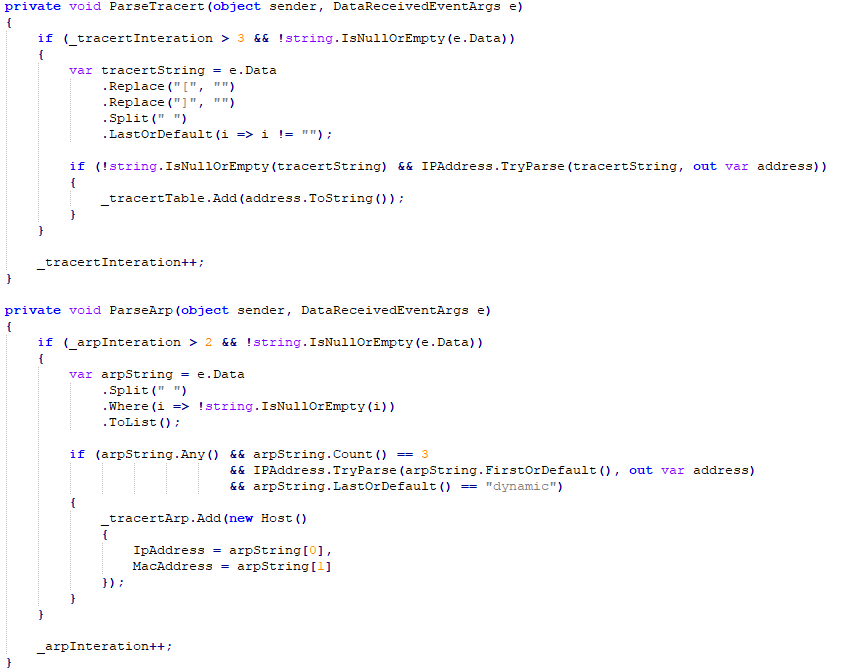


Рисунок 10 – Класс WindowsCmdManager

WindowsHostInformationService – класс реализующий интерфейс IHostInformationService (рисунок 11). Класс WindowsHostInformationService содержит в себе методы получающие информацию об узле сети. В классе WindowsHostInformationService есть две приватный переменный доступных только для чтения \_ipInterfaceProperties типа IPInterfaceProperties и \_manager типа IWindowsManager. В конструкторе с помощью такой техники как DI (Dependency injection) происходит внедрение зависимостей, делается это для большей гибкости и для удобства написания Unit тестов, так как можно указать любую реализацию реализующий интерфейс. Так же класс WindowsHostInformationService сдержит реализацию всех методов интерфейса IHostInformationService. Методы GetTracertTable и GetArpTable внутри себя вызывают методы из перемены \_manager, где может быть любой экземпляр класса реализующий интерфейс IWindowsManager.

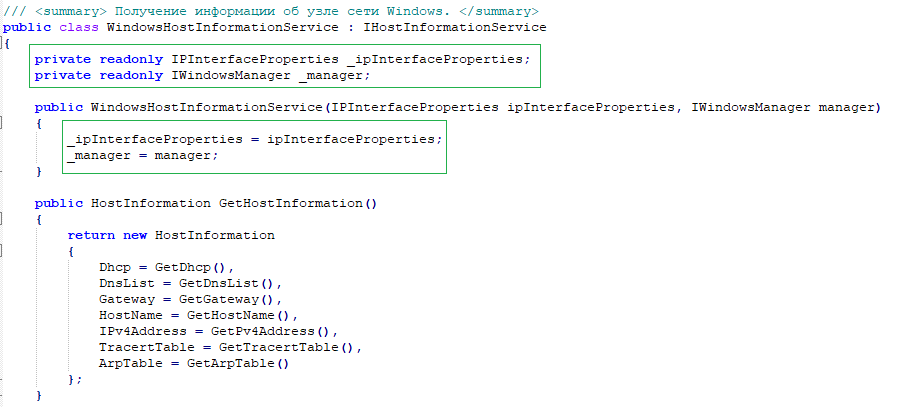


Рисунок 11 – Класс WindowsHostInformationService

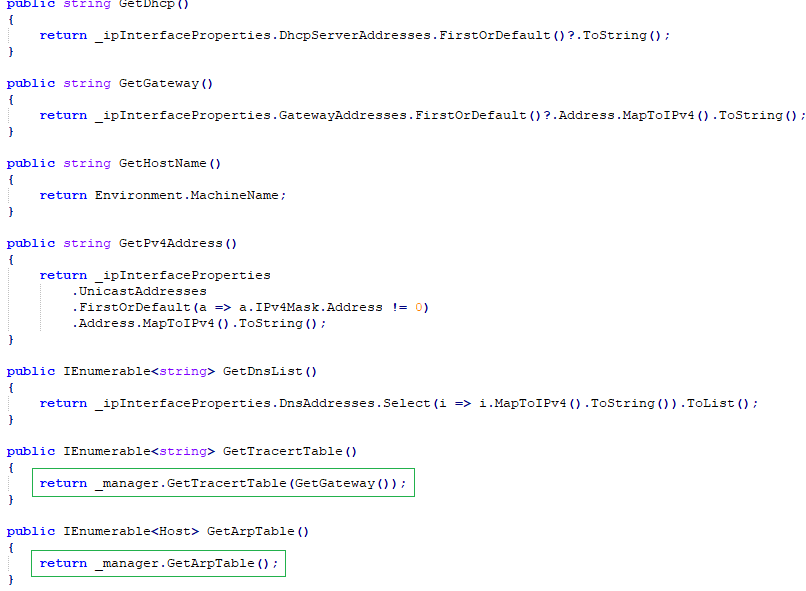


Рисунок 12 – Класс WindowsHostInformationService

## **Описание NetworkMonitor.WindowsService**

NetworkMonitor.WindowsService – сборка для создания клиента в виде службы window. Для этого необходимо добавить к проекту следующие Nuget пакеты Microsoft.Extensions.Hosting, Microsoft.Extensions.Hosting.WindowsServices и ссылка на проекты NetworkMonitor.Common и NetworkMonitor.Implementation. На рисунке 13 изображен кода из файла Program.cs в котором происходит конфигурация сервиса. Получение параметров из файла appsettings.json. И внедрение зависимостей в IoC контейнер. И также добавление WindowsWorker в котором будет выполнятся все логика. Рисунок 14. На рисунке 14 показан код работы метода ExecuteAsync в котором происходит получение настроек узла и отправку их на сервер (пока с помощью заглушки пишется в json файл). Затем идет задержка, значение которой берется из файла конфигурации и итерация повторяется. Для создания сервиса Windows необходимо скомпилировать сборку NetworkMonitor.WindowsService и пополнить bat скрипт правами администратора указав в нем путь до NetworkMonitor.WindowsService.exe “create NetworkMonitor.WindowsService.bat” код которого представлен на рисунке 15. После чего можно запустить утилиту services.msc и убедиться, что создана новая служба, которую можно настроить и запускать или выключать. По умолчанию для служб Windows используется папка C:\Windows\System32. Можно увидеть, что NetworkMonitor.WindowsService создал папку mock и сохраняет туда значение Dto HostInformation серилизуя их в json файл рисунок 17, рисунок 18. Запустив утилиту Event Viewer, можно увидеть, что системная служба пишет логи о своей работе рисунок 18.



Рисунок 13 – Конфигурация сервиса NetworkMonitor.WindowsService



Рисунок 14 – Конфигурация сервиса NetworkMonitor.WindowsService



Рисунок 15 – Скрипт create NetworkMonitor.WindowsService.bat

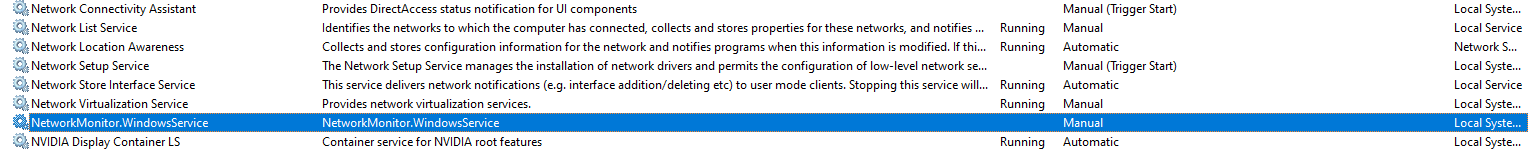


Рисунок 16 – Скрипт services.msc

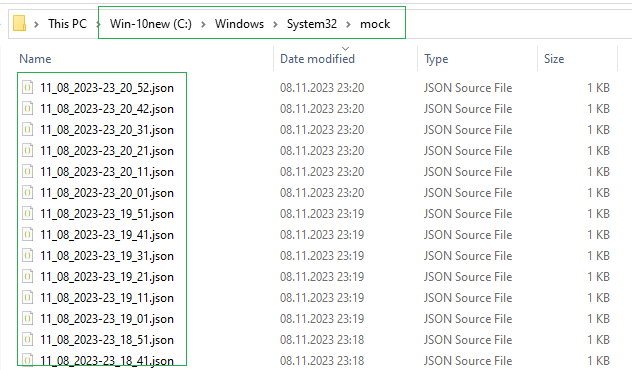


Рисунок 17 – Папка mock.



Рисунок 18 – Пример json файла.

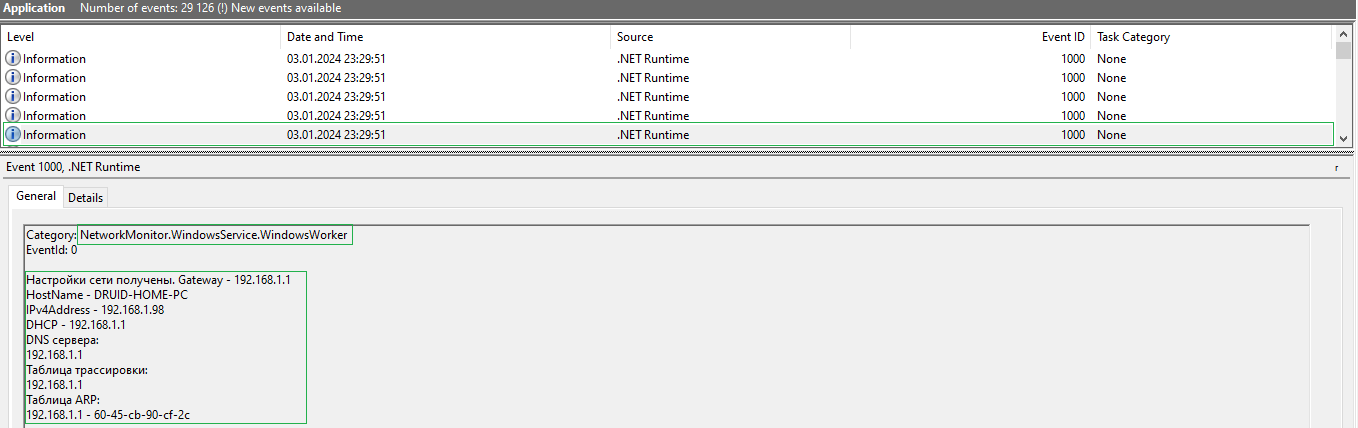


Рисунок 19 – Логи Windows отображаемые в программе Event Viewer.

**ЗАКЛЮЧЕНИЕ**

Основная идея разработки написать сетевое клиент-серверное приложение, которое позволяло бы централизованно выявлять атаки MITM (“человек посередине») и оперативно реагировать на инциденты. В статье описывается написание первой части этого приложения, а именно клиента, который планируется устанавливать на каждый узел сети, для отправки информации о сетевых настройках, а также ARP таблицы и таблицы трассировки. Описана архитектура клиент-серверного приложения, а также показан процесс компиляции и запуска приложения клиента в качестве системной службы Windows.

**Литература**

1. Tanenbaum, Andrew S. and Austin, Todd. Structured computer organization. 6th ed. Pearson Education, Inc., publishing as Prentice Hall. 2015. – 801 p.
2. Шахнович, И.А. Современные технологии беспроводной связи. М: Техносфера, 2006. – 288 с.
3. Олифер В. Г., Олифер Н. А. Компьютерные сети. Принципы, технологии, протоколы. СПб.: Питер. – 2010. – 916 с.
4. Снейдер, И. Эффективное программирование TCP/IP. Библиотека программиста : пер. с анг. – М.: ДМК Пресс. – 2019. – 322 с.
5. Биджиева С.Х., Шебзухова К.В. Сетевые протоколы передачи данных: преимущества и недостатки // Тенденции развития науки и образования. 2022. Т. 86. № 1. С. 43–45. doi: 10.18411/trnio-06-2022-14.
6. X. Hou, Z. Jiang, and X. Tian. The Detection and Prevention for ARP Spoofing based on SNORT // IEEE International Conference on Computer Application and System Modeling (ICCASM'10), 2010. vol. 5, pp. 125–137.
7. Дементьев В.Е., Чулков А.А. Кибервоздействия на протоколы сетей передачи данных // Изв. ТулГУ. Технические науки. 2020. № 10. С. 245–254.
8. Aldaoud M. et al. DHCP attacking tools: an analysis //Journal of Computer Virology and Hacking Techniques. – 2021. – Т. 17. – С.119-129.
9. Muthalagu R., Sanjay S. Evil Twin Attack Mitigation Techniques in 802.11 Networks //International Journal of Advanced Computer Science and Applications. – 2021. – Т. 12. – №. 6.
10. Mehrotra T. A review on attack in wireless and computer networking //Asian Journal of Multidimensional Research. – 2021. – Т. 10. – №. 10. – С. 1457-1463.
11. Степанов, П. П. Безопасная передача сообщений с разделением данных через почтовые серверы / П. П. Степанов, Г. В. Никонова // Вопросы кибербезопасности. – 2024. Том 60, № 2. – С. 120-129. DOI: 10.21681/2311-3456-2024-2-120-129.
12. Man K., Zhou X., Qian Z. DNS cache poisoning attack: Resurrections with side channels //Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security. – 2021. – С. 3400-3414.
13. Mvah F. et al. GaTeBaSep: game theory-based security protocol against ARP spoofing attacks in software-defined networks //International Journal of Information Security. – 2023. – С. 1-15.
14. Stepanov P.P. Attack on the Address Resolution Protocol / Stepanov P.P., Nikonova G.V., Pavlychenko T.S., Gil A.S. // 2020 International Conference Engineering and Telecommunication (En&T), 2020, pp. 1-3.
15. Galal A. A., Ghalwash A. Z., Nasr M. A New Approach for Detecting and Mitigating Address Resolution Protocol (ARP) Poisoning //International Journal of Advanced Computer Science and Applications. – 2022. – Т. 13. – №. 6.
16. Stepanov P.P. The problem of security address resolution protocol / P.P. Stepanov, G.V. Nikonova, T.S. Pavlychenko, A.S. Gil // Journal of Physics: Conference Series. - 2021, Vol. 1791, p.p. 1-8.
17. Степанов, П. П. Особенности работы протокола разрешения адресов в компьютерных сетях / П. П. Степанов, Г. В. Никонова, Т. С. Павлюченко, В. В. Соловьев // Программная инженерия. – 2022. Том 13, № 5. – С. 211-218.
18. Shah Z, Cosgrove S. Mitigating ARP Cache Poisoning Attack in Software-Defined Networking (SDN): A Survey. *Electronics*. 2019; 8(10):1095. https://doi.org/10.3390/electronics8101095.
19. Hijazi, S.; Obaidat, M. Address resolution protocol spoofing attacks and security approaches: A survey. Secur. Priv. 2019, 2, e49
20. [Shashi Shaw](https://ieeexplore.ieee.org/author/37085480387), Prasenjit Choudhury. A new local area network attack through IP and MAC address spoofing // IEEE International Conference on Advances in Computer Engineering and Applications. Ghaziabad, India. DOI: 10.1109/ICACEA.2015.7164728.
21. Jen-Hao Kuo; Siong-Ui Te; Pang-Ting Liao [et all.]. An evaluation of the virtual router redundancy protocol extension with load balancing // 11th Pacific Rim International Symposium on Dependable Computing (PRDC'05) Hunan, China, Added to IEEE Xplore: 20 March 2006. DOI: [10.1109/PRDC.2005.16](https://doi.org/10.1109/PRDC.2005.16).
22. Maximov R.V., Sokolovsky S.P., Telenga A.P. Model of client-server information system functioning in the conditions of network reconnaissance. CEUR Workshop Proceeding. 2019. pp. 44-51.
23. [Mujahid Shah](https://ieeexplore.ieee.org/author/37086803375), [Sheeraz Ahmed](https://ieeexplore.ieee.org/author/37086133637), [Khalid Saeed](https://ieeexplore.ieee.org/author/37086692637), [Muhammad Junaid](https://ieeexplore.ieee.org/author/37086803611), [Hamayun Khan](https://ieeexplore.ieee.org/author/37086803793), [Ata-ur Rehman](https://ieeexplore.ieee.org/author/37086804526). Penetration Testing Active Reconnaissance Phase – Optimized Port Scanning With Nmap Tool.  [// 2nd International Conference on Computing, Mathematics and Engineering Technologies (iCoMET)](https://ieeexplore.ieee.org/xpl/conhome/8671662/proceeding), 2019. Sukkur, Pakistan. DOI: [10.1109/ICOMET.2019.8673520](https://doi.org/10.1109/ICOMET.2019.8673520).
24. Tanenbaum, Andrew S. and Austin, Todd. Structured computer organization. 6th ed. Pearson Education, Inc., publishing as Prentice Hall. 2015. – 801 p.
25. Hamilton Turner, Jules White, Jaime Camelio, Christopher Williams, Brandon Amos and Robert Parker. Bad Parts: Are Our Manufacturing Systems at Risk of Silent Cyberattacks? //IEEE Security & Privacy IEEE Computer Society. 2015.

1. Степанов Петр Петрович, старший преподаватель, федерального государственного автономного учреждения высшего образования «Омский государственный технический университет» (ОмГТУ), г. Омск, Россия, E-mail: omsk.petr@gmail.com [↑](#footnote-ref-1)
2. Никонова Галина Владимировна, кандидат технических наук, доцент, федерального государственного автономного учреждения высшего образования «Омский государственный технический университет» (ОмГТУ), г. Омск, Россия. E-mail: nikonova@omgtu.ru [↑](#footnote-ref-2)