УДК 004.056.53
DOI: 10.26102/2310-6018/2025.50.3.042

Защита от атак в режиме RTU протокола Modbus на основе криптографической верификации узлов ведущий-ведомый

А.А. Южаков, Е.Л. Кротова, Н.В. Ощепков✉

Пермский национальный исследовательский политехнический университет, Пермь, Российская Федерация

Резюме. В данной работе анализируются особенности протокола Modbus, с акцентом на его уязвимость в контексте безопасности и защиты передаваемой информации. Рассмотрены основные риски, связанные с использованием Modbus в системах автоматизации и управления технологическими процессами (АСУ ТП), включая отсутствие механизмов шифрования и аутентификации, что делает его уязвимым к различным видам атак, таким как перехват данных или несанкционированный доступ, а также варианты решения проблемы верификации узлов. Протокол Modbus является одним из наиболее распространённых и востребованных промышленных протоколов, активно используемых в системах автоматизации и управления различными технологическими процессами. Протокол отличается простотой реализации и широким распространением, что делает его привлекательным для внедрения в разнообразных отраслях промышленности. Тем не менее в режиме RTU протокола Modbus имеются недостатки, такие как уязвимость к атакам «человек посередине» и «подмена», что несёт в себе потенциальные риски для промышленных предприятий, использующих данный протокол на производстве. Наличие уязвимости обусловлено отсутствием встроенных механизмов аутентификации и верификации узлов, участвующих в передаче данных. Это создаёт риски, связанные с возможностью несанкционированного доступа и подмены информации в процессе обмена. В статье предложен метод повышения конфиденциальности при взаимодействии между узлами путём внедрения криптографических операций, позволяющих обеспечить проверку подлинности источника передаваемых данных посредством внедрения легковесного криптографического алгоритма, основанного на операции XOR с 16-битным секретом. Преимуществом предложенного метода является его совместимость с существующей реализацией протокола Modbus, минимальное влияние на производительность системы и отсутствие необходимости в глубокой модификации архитектуры. Также стоит отметить незначительное увеличение задержки при передаче данных (менее чем на 2 %) и потребления процессорного времени.

Ключевые слова: Modbus RTU, «человек посередине», фрейм, криптографическая защита, промышленный протокол.

Для цитирования: Южаков А.А., Кротова Е.Л., Ощепков Н.В. Защита от атак в режиме RTU протокола Modbus на основе криптографической верификации узлов ведущий-ведомый. *Моделирование, оптимизация и информационные технологии.* 2025;13(3). URL: https://moitvivt.ru/ru/journal/pdf?id=2021 DOI: 10.26102/2310-6018/2025.50.3.042

Protection against attacks in RTU mode of Modbus protocol based on cryptographic verification of master-slave nodes

A.A. Yuzhakov, E.L. Krotova, N.V. Oshchepkov✉

Perm National Research Polytechnic University, Perm, the Russian Federation

Abstract. This paper analyzes the features of the Modbus protocol, with an emphasis on its vulnerability in the context of security and protection of transmitted information. The main risks associated with the use of Modbus in automation and process control systems (APCS) are considered, including the lack of encryption and authentication mechanisms, which makes it vulnerable to various types of attacks, such

as data interception or unauthorized access, as well as options for solving the problem of node verification. The Modbus protocol is one of the most common and popular industrial protocols, actively used in automation systems and control of various technological processes. The protocol is easy to implement and widespread, which makes it attractive for implementation in various industries. However, the RTU mode of the Modbus protocol has disadvantages, such as vulnerability to man-in-the-middle and substitution attacks, which carries potential risks for industrial enterprises using this protocol in production. The vulnerability is due to the lack of built-in authentication and verification mechanisms for nodes involved in data transmission. This creates risks associated with the possibility of unauthorized access and substitution of information during the exchange process. The article proposes a method for increasing confidentiality during interaction between nodes by implementing cryptographic operations that allow for verification of the authenticity of the source of transmitted data by implementing a lightweight cryptographic algorithm based on the XOR operation with a 16-bit secret. The advantage of the proposed method is its compatibility with the existing implementation of the Modbus protocol, minimal impact on system performance, and no need for deep modification of the architecture. It is also worth noting a slight increase in data transmission latency (less than 2 %) and processor time consumption.

Keywords: Modbus RTU, man-in-the-middle, frame, cryptographic protection, industrial protocol.

For citation: Yuzhakov A.A., Krotova E.L., Oshchepkov N.V. Protection against attacks in RTU mode of Modbus protocol based on cryptographic verification of master-slave nodes. *Modeling, Optimization and Information Technology*. 2025;13(3). (In Russ.). URL: https://moitvivt.ru/ru/journal/pdf?id=2021 DOI: 10.26102/2310-6018/2025.50.3.042

Введение

Протокол Modbus [1] является одним из наиболее распространённых решений на рынке промышленных протоколов, используемых в автоматизированных системах управления технологическим процессом. К его достоинствам можно отнести простоту реализации, лёгкость внедрения и открытый исходный код. Тем не менее режим RTU [2] протокола имеет недостаток в виде отсутствия механизма верификации отправителя в процессе передачи. Наличие открытости исходного кода протокола требует повышения уровня конфиденциальности данных в процессе передачи.

Материалы и методы

Рассмотрим возможность возникновения двух сценариев атаки.

1. Подмена узла (Spoofing attack) [3]. Данный тип атаки предполагает имитацию легитимного ведущего (master) или ведомого (slave) устройства в сети Modbus RTU. Злоумышленник, проникнув в защищенный периметр, осуществляет подмену одного или нескольких сетевых узлов. При успешной реализации атаки злоумышленник получает возможность:

– Формировать и передавать управляющие команды, маскируясь под ведущее устройство, что позволяет произвольно изменять параметры работы любого узла в сети.

– Считывать конфиденциальные данные с ведомых устройств.

– Имитировать ответы ведомых устройств, вводя в заблуждение ведущий узел и нарушая достоверность передаваемой информации.

2. Атака «человек посередине» (man-in-the-middle) [4]. В рамках данной атаки злоумышленник осуществляет перехват сетевого трафика между взаимодействующими узлами с возможностью его модификации или анализа. После проникновения в защищенную сеть атакующий внедряется в канал связи, становясь промежуточным звеном. Это позволяет:

– Осуществлять пассивный перехват данных с целью их последующего анализа.

– Активно модифицировать передаваемые сообщения, подменяя команды управления или показания датчиков.

Одним из критических последствий данной атаки является введение оператора в заблуждение. Например, если оператор отправляет команду на изменение параметра ведомого устройства, а злоумышленник блокирует или модифицирует эту команду, оператор получит ложное подтверждение выполнения операции. Это может привести к некорректному функционированию системы и возникновению аварийных ситуаций.

При обоих сценариях, злоумышленник получает несанкционированный доступ к информации, что может привести к нарушению технологического процесса, маскировке реальных параметров, фальсификации показаний датчиков, подмене команд и т. д.

Эксперимент проводился на базе персонального компьютера с ОС Linux [5]. При программной реализации рассматривались несколько возможных вариантов протокола на языке Python3: pymodbus[1], minimalmodbus[2], umodbus[3]. Minimalmodbus не удовлетворил требованиям в модификации исходного кода, а umodbus является реализацией для микроконтроллеров на базе ОС Micropython. Поэтому в результате был выбран PyModbus.

Для создания шины передачи данных была использована утилита socat[4], позволяющая создать двунаправленный канал передачи данных на одном персональном компьютере путём резервирования портов. В реальных условиях атака подмены может быть реализована путем физического отключения устройства и подмены на устройство злоумышленника. В программной же реализации данное событие смоделировано путем изменения поведения устройства после отправки определенного количества кадров канального уровня модели OSI [6] (далее фреймов), либо же по истечении промежутка времени от начала отправки. Атака «человек посередине» также реализуется путем встраивания вредоносного устройства в канал передачи данных. В эксперименте это реализуется созданием виртуального шлюза, пропускающего фреймы через себя.

Предложенная идея заключается в использовании 16-битной вставки (далее «секрет») путём операции XOR [7] с двумя байтами передаваемого фрейма протокола Modbus RTU. Сегмент фрейма, к которому применяется операция XOR, показан на Рисунке 1.

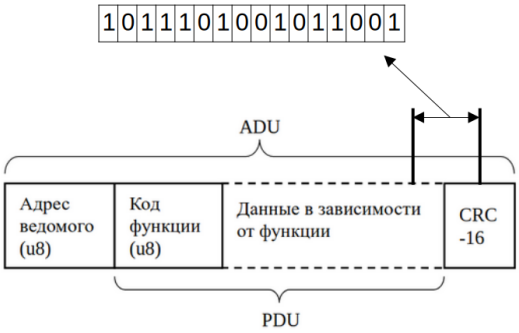

Рисунок 1 – Сегмент фрейма Modbus RTU, подлежащий модификации
Figure 1 – Modbus RTU frame segment to be modified

[1] Welcome to PyModbus's documentation! – PyModbus 4.0.9dev4 documentation. URL: https://pymodbus.readthedocs.io/en/latest/ (дата обращения: 15.06.2025).
[2] Welcome to MinimalModbus' documentation! – MinimalModbus 2.1.1 documentation. URL: https://minimalmodbus.readthedocs.io/en/stable/ (дата обращения: 15.06.2025).
[3] uModbus – uModbus 1.0.0 documentation. URL: https://umodbus.readthedocs.io/en/latest/ (дата обращения: 15.06.2025).
[4] Rieger G. Ubuntu Manpage: socat – Multipurpose relay (SOcket CAT). Ubuntu Manpage. URL: https://manpages.ubuntu.com/manpages/bionic/man1/socat.1.html (дата обращения: 17.06.2025).

Следует подчеркнуть, что значение «секрета» должно находиться в интервале от 2^{15} до 2^{16}, что необходимо для обеспечения модификации обоих байтов фрейма. Кроме того, каждое устройство обладает уникальным персонализированным секретом, который интегрируется непосредственно в программный код устройства.

На Рисунке 2 представлен модифицированный алгоритм взаимодействия между ведущим и ведомым устройством Modbus RTU в контексте предложенного метода.

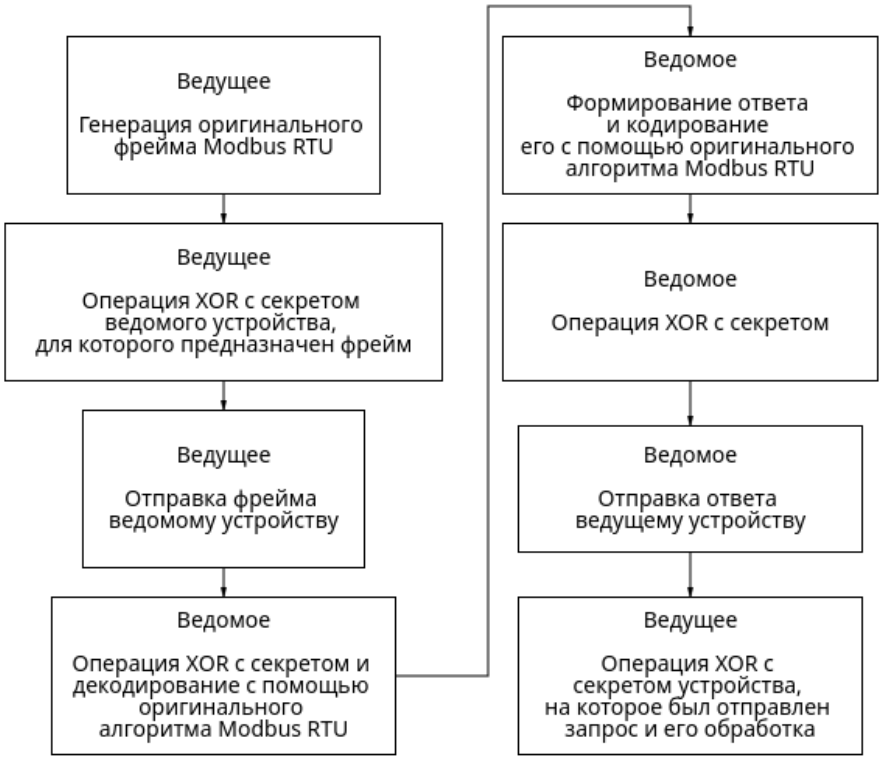

Рисунок 2 – Алгоритм модифицированного взаимодействия узлов
Figure 2 – Modified node interaction algorithm

Из Рисунка 2 видно, что неизменность сеанса передачи данных между устройствами достигается благодаря свойству операции сложения по модулю 2: A XOR A = 0.

Программно модификация алгоритма на языке Python3 выглядит следующим образом (Рисунок 3).

```python
import struct


class CipherMixin:
    def cipher_decipher(self, packet: bytes, secret: int) -> bytes:
        target_bytes = packet[-3:-1]
        original_value = struct.unpack('>H', target_bytes)[0]
        new_bytes = struct.pack('>H', original_value ^ secret)
        return packet[:-3] + new_bytes + packet[-1:]   # Остаток пакета (CRC)
```

Рисунок 3 – Функция модификации фрейма с секретом ведомого устройства
Figure 3 – Frame modification function with slave secret

Рассмотрим четыре сценария атаки подмены:

1. Злоумышленник отправляет фрейм, сгенерированный базовым алгоритмом Modbus, в качестве ведущего устройства на модифицированное ведомое устройство.

2. Модифицированное ведущее устройство отправляет на ведомое устройство злоумышленника фрейм.

3. Злоумышленник отправляет фрейм, сгенерированный модифицированным алгоритмом Modbus, в качестве ведущего устройства на модифицированное ведомое устройство с попыткой подобрать секрет.

4. Модифицированное ведущее устройство отправляет на модифицированное ведомое устройство злоумышленника фрейм с попыткой подобрать секрет.

На Рисунке 4 представлена более детальная схема взаимодействия узлов при указанных сценариях:

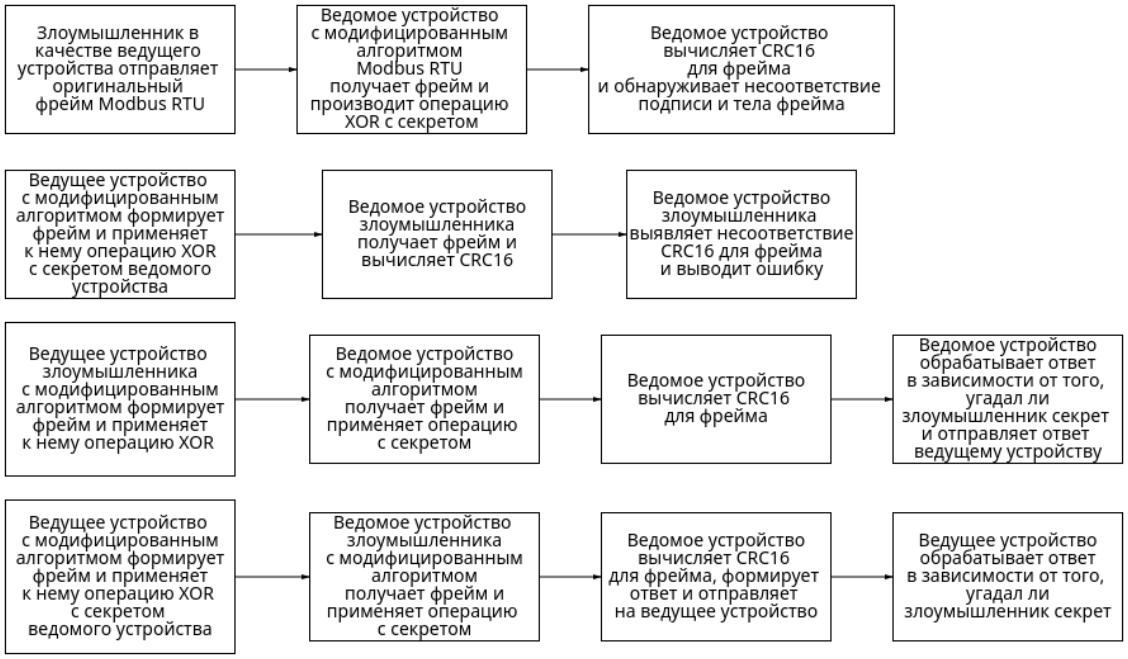

Рисунок 4 – Схема взаимодействия узлов при различных сценариях
Figure 4 – Scheme of interaction of nodes in different scenarios

Для воспроизведения первых двух видов атак, необходимо выполнить следующие шаги:

1. Создать двунаправленный канал связи с помощью следующей команды: socat -d -d pty, raw, echo = 0 pty, raw, echo = 0.

2. Совершить подмену ведущего или ведомого устройства в схеме взаимодействия на программные реализации, использующие оригинальный алгоритм формирования фрейма.

3. Отправить фрейм или же принять его.

В первых двух вариантах как модифицированное ведущее устройство, так и модифицированное ведомое устройство отклоняют запрос/ответ от устройств злоумышленника, поскольку происходит модификация последнего байта PDU и первого байта CRC16 [8]. Устройство злоумышленника отвергает запрос, так как считает, что фрейм сформирован некорректно на основании вычисления контрольной суммы. Это свидетельствует о том, что оригинальный алгоритм формирования фрейма для RTU режима протокола Modbus не позволяет злоумышленнику выдать себя за легитимное

устройство. Библиотека pyModbus также сообщает о проблеме при проверке подписи: DEBUG:pymodbus.logging:Frame check failed, possible garbage after frame, testing.

Для воспроизведения 3 и 4 видов атак злоумышленнику необходимо заранее быть осведомленным об использовании модифицированного алгоритма, использующего индивидуальный секрет устройства. Шаги будут следующие:

1. Создать двунаправленный канал связи с помощью следующей команды: socat -d -d pty, raw, echo = 0 pty, raw, echo = 0.
2. Совершить подмену ведущего или ведомого устройства в схеме взаимодействия на программные реализации, использующие модифицированный алгоритм формирования фрейма.
3. Совершить попытку подбора индивидуального секрета ведомого устройства.
4. Отправить фрейм.

В 3 варианте атаки легитимное ведомое устройство будет отвергать ответ злоумышленника до тех пор, пока злоумышленник не подберет индивидуальный секрет для ведомого устройства путем отправки запросов. В тестовой среде с помощью метода brute-force [9] необходимо будет перебрать 2^{15} вариантов, что займет порядка 9 часов в виду программной реализации протокола. В библиотеке pyModbus происходит задержка в 2 секунды, в течение которой ведущее устройство ожидает ответ от ведомого. В программной реализации данного типа атаки ведущее устройство злоумышленника получает сгенерированный фрейм, но отвергает запрос ввиду ошибки при проверке контрольной суммы.

В 4 варианте злоумышленник, получив фрейм от легитимного ведомого устройства, может подобрать секрет на самом устройстве методом перебора, не будет задействовать канал передачи данных как в случае 3. Перебор может быть осуществлен методом грубой силы, а количество возможных вариантов составит 2^{15}, что на современных компьютерах займет не менее 0,1 секунды. Несмотря на это, злоумышленник должен быть осведомлен о том, какие сегменты фрейма были отредактированы.

В случае осуществления злоумышленником атаки «человек посередине», пакет может быть перехвачен, но, так как были изменены полезная нагрузка и первый байт CRC16, злоумышленник не сможет корректно обработать данные с целью прослушки. Также злоумышленник не сможет произвести модификацию фрейма налету, так как это будет обнаружено как ведомым, так и ведущим устройствами ввиду осуществления операции XOR с полученным фреймом. На Рисунке 5 представлена схема атаки.

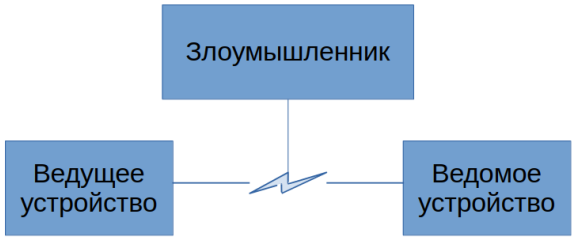

Рисунок 5 – Схема атаки на режим RTU протокола Modbus
Figure 5 – Scheme of attack on RTU mode of Modbus protocol

Для реализации атаки, необходимо зарезервировать уже 4 порта. Это может быть достигнуто путем запуска утилиты socat в двух различных терминалах.

Создадутся 2 пары портов, например 10–11, 12–13. Чтобы осуществить перехват, злоумышленнику необходимо стать посредником в данной схеме, то есть слушать на

портах 11 и 12. Таким образом, ведущее устройство будет посылать запрос на 11 порт, но перед этим фрейм будет перехвачен, затем отправлен с 12 порта на 13. В данной схеме злоумышленник становится посредником в канале передачи, что даёт ему возможность прослушивать весь трафик. Несмотря на это, если злоумышленник не знает об используемом модифицированном алгоритме, он не сможет ни изменить фрейм, ни корректно его обработать. Программная реализация перехвата на Serial портах представлена на Рисунке 6.

```python
import serial
import threading

MASTER_PORT = "/dev/pts/11"  # Ведущее
SLAVE_PORT = "/dev/pts/12"   # Ведомое

def forward(src, dst, label):
    while True:
        data = src.read(1)
        if data:
            remaining = src.in_waiting
            if remaining:
                data += src.read(remaining)
            print(f"[{label}] {data.hex()}")
            dst.write(data)

def main():
    master = serial.Serial(MASTER_PORT, baudrate=9600, timeout=0.1)
    slave = serial.Serial(SLAVE_PORT, baudrate=9600, timeout=0.1)

    t1 = threading.Thread(target=forward, args=(master, slave, "Master → Slave"))
    t2 = threading.Thread(target=forward, args=(slave, master, "Slave → Master"))
    t1.start()
    t2.start()
    t1.join()
    t2.join()

if __name__ == "__main__":
    main()
```

Рисунок 6 – Программная реализация перехвата пакетов между ведущим и ведомым устройством

Figure 6 – Code implementation of packet interception between master and slave device

В случае, если злоумышленник не знает об используемом алгоритме, он не сможет корректно обработать фрейм с целью получения информации. Если же злоумышленник предпримет попытку отредактировать фрейм, то как ведущее устройство, так и ведомое обнаружат модификацию в момент вычисления контрольной суммы после выполнения операции XOR. В случае же, если злоумышленник знает об используемом алгоритме, то ему также придется перебрать 2^{15} вариантов секрета, что для устройства-перехватчика занимает также не менее 0,1 секунды.

Результаты

Благодаря предложенному методу происходит модификация PDU и CRC16, вычисленной на основе PDU. Таким образом, скрываются значения полезной нагрузки и злоумышленник, выступающий в качестве посредника в канале передачи данных, не сможет определить изначальный вид PDU. Более того, благодаря отсутствию у злоумышленника информации об используемом «секрете» он не сможет выдать себя за master или slave устройство, пока им не будет предпринята попытка подбора секрета, но только в некоторых схемах взаимодействия.

На Рисунке 7 представлены графики, отражающие задержку получения ответа при различных значениях бод [10]. При использовании скорости передачи 9600 бод задержка между отправкой и получением ответа при использовании модифицированного алгоритма увеличилась в среднем на 0,13 мс, что составляет 1,47 % от изначального среднего значения в 8,7 мс. При использовании скорости передачи 19200 бод задержка увеличилась на 0,1 мс, что составляет 1 % от изначального среднего значения в 17,22 мс.

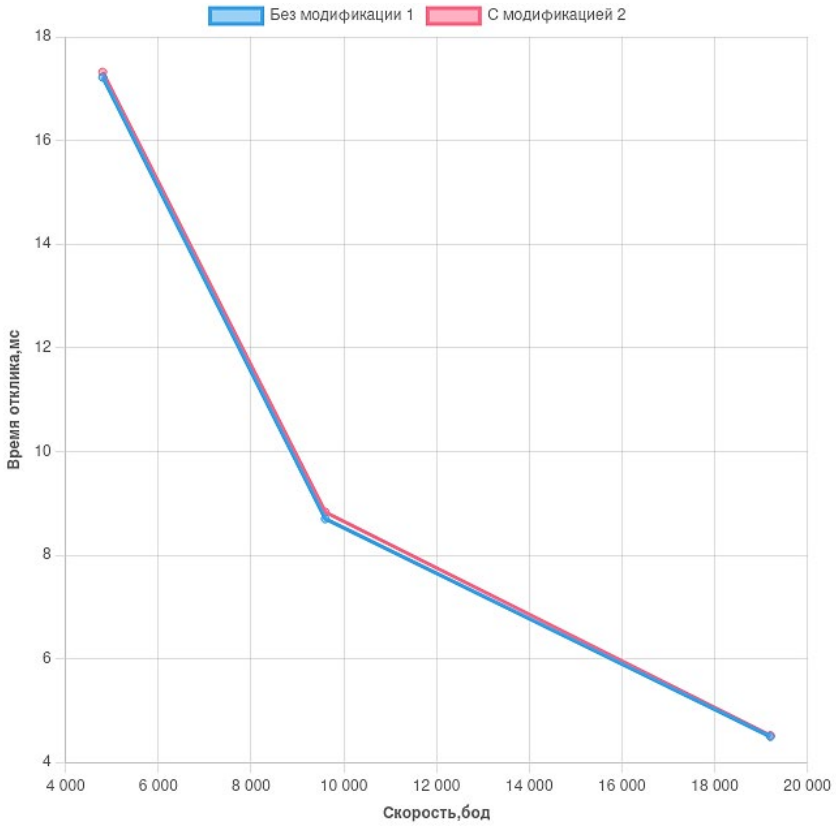

Рисунок 7 – Графики задержки ответа
Figure 7 – Response latency graphs

Заключение

Исходя из вышеизложенного, можно сделать вывод, что предложенная модификация с большой вероятностью исключает атаки подмены и «человек посередине» для устройств в случае, если злоумышленник не осведомлен о модифицированном протоколе. В случае если злоумышленник внедрил модифицированный протокол и подменил ведущее устройство, тогда возможно подобрать секрет только лишь отправкой пакетов. В случае осведомленности о модифицированном алгоритме и подмены ведомого устройства или перехвата фреймов в атаке «человек посередине» возможен подбор секрета методом полного перебора, что занимает много времени.

СПИСОК ИСТОЧНИКОВ / REFERENCES

1. Томас Дж. Введение в протокол Modbus. Часть 1. *СТА: Современные технологии автоматизации.* 2009;(2):52–57.

2. Томас Дж. Введение в протокол Modbus. Часть 2. Modbus Serial и Modbus TCP. *СТА: Современные технологии автоматизации.* 2009;(3):22–26.
3. Alakbarov R.G., Hashimov M.A. Application and Security Issues of Internet of Things in Oil-Gas Industry. *International Journal of Education and Management Engineering.* 2018;8(6):24–36. https://doi.org/10.5815/ijeme.2018.06.03
4. Арзуманян Э.А., Чумаков А.А. МИТМ-атака. Угроза информационной безопасности в РФ. *Znanstvena Misel*. 2019;(8–1):37–40.
Arzumanyan E., Chumakov A. MITM Attack. Threat to Information Security in the Russian Federation. *Znanstvena Misel*. 2019;(8–1):37–40. (In Russ.).
5. Таненбаум Э., Бос Х. *Современные операционные системы.* Санкт-Петербург: Питер; 2015. 1120 с.
Tanenbaum A.S., Bos H. *Modern Operating Systems.* Saint Petersburg: Piter; 2015. 1120 p. (In Russ.).
6. Прокопенко Л.Л., Ионан Ю.Э. Модель OSI для организации компьютерных сетей. *Вестник образовательного консорциума среднерусский университет. Информационные технологии.* 2021;(1):40–42. https://doi.org/10.52374/52100412_2021_17_1_40
Prokopenko L.L., Ionan Yu.E. OSI Model for Organizing Computer Networks. *Vestnik obrazovatel'nogo konsortsiuma srednerusskii universitet. Informatsionnye tekhnologii.* 2021;(1):40–42. (In Russ.). https://doi.org/10.52374/52100412_2021_17_1_40
7. Гашков С.Б. *Сложение однобитных чисел. Треугольник Паскаля, салфетка Серпинского и теорема Куммера.* Москва: МЦНМО; 2014. 40 с.
8. Бородулин В. Сравнительные характеристики алгоритмов расчёта CRC16 последовательным и табличным способом на примере микроконтроллера AVR. *Современная электроника.* 2008;(2):74–77.
9. Мансур А.М. Алгоритм предобработки данных для нейросетевой системы определения автоматического подбора пароля. *Молодой исследователь Дона.* 2018;(6):34–38.
Mansour A.M. Data Pre-Processing Algorithm for the Neural Network System for Determining Automatic Password Selection. *Young Don Researcher.* 2018;(6):34–38. (In Russ.).
10. Коршунов В.Н. Увеличение скорости передачи информации по оптическим кабелям. *Кабели и провода*. 2017;(1):16–19.

ИНФОРМАЦИЯ ОБ АВТОРАХ / INFORMATION ABOUT THE AUTHORS

Южаков Александр Анатольевич, доктор технических наук, профессор, заведующий кафедрой «Автоматика и телемеханика», Пермский национальный исследовательский политехнический университет, Пермь, Российская Федерация.
e-mail: mailto:uz@at.pstu.ru

Alexander A. Yuzhakov, Doctor of Engineering Sciences, Professor, Head of the Department of Automation and Telemechanics, Perm National Research Polytechnic University, Perm, the Russian Federation.

Кротова Елена Львовна, кандидат физико-математических наук, доцент кафедры «Высшая математика», Пермский национальный исследовательский политехнический университет, Пермь, Российская Федерация.
e-mail: mailto:lenkakrotova@yandex.ru

Elena L. Krotova, Candidate of Physical and Mathematical Sciences, Associate Professor at the Department of Higher Mathematics, Perm National Research Polytechnic University, Perm, the Russian Federation.

Ощепков Никита Владимирович, аспирант кафедры «Автоматика и телемеханика», Пермь, Российская Федерация.
e-mail: maserati_2000@mail.ru

Nikita V. Oshchepkov, Postgraduate at the Department of Automation and Telemechanics, Perm, the Russian Federation.

Статья поступила в редакцию 18.07.2025; одобрена после рецензирования 18.08.2025; принята к публикации 26.08.2025.

The article was submitted 18.07.2025; approved after reviewing 18.08.2025; accepted for publication 26.08.2025.