МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

«САМАРСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИМЕНИ АКАДЕМИКА С.П. КОРОЛЕВА»

Институт информатики, математики и электроники

Факультет информатики

Кафедра геоинформатики и информационной безопасности

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА

**(специалист)**

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

**РАЗРАБОТКА ЛАБОРАТОРНОГО СТЕНДА ДЛЯ МОДЕЛИРОВАНИЯ АТАК НА КАНАЛ ПЕРЕДАЧИ ДАННЫХ ПО ПРОТОКОЛУ MODBUS**

Выпускник Резванов Дмитрий Андреевич

(фамилия, имя, отчество)

Руководитель (Жмуров Д.Б.)

(фамилия,И.О.)

Рецензент ( ) (фамилия, И.О.)

Нормоконтролёр (Клевцова Е.В.)

(фамилия, И.О.)

****

**МИНОБРНАУКИ РОССИИ**

**федеральное государственное автономное**

**образовательное учреждение высшего образования**

**«самарский национальный исследовательский университет**

**имени академика с. п. королЕва»**

Кафедра геоинформатики и информационной безопасности

УТВЕРЖДАЮ

Заведующий кафедрой

\_\_\_\_\_\_\_\_\_\_\_\_\_\_ /В.В. Сергеев/

«\_\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_20\_\_\_ г.

**Задание на выпускную квалификационную работу (ВКР)**

Студенту Резванову Д.А. 6511 С 317 группы

1. Тема ВКР: **«Разработка лабораторного стенда для моделирования атак на канал передачи данных по протоколу Modbus»** утверждена приказом по университету от «?» марта 2017 г. № 291-ст.

2. Исходные данные к ВКР: ?

3. Перечень вопросов, подлежащих разработке в ВКР:

3.1. Изучение вопросов информационной безопасности в АСУ ТП.

3.2. Изучение протокола Modbus, поиск уязвимостей в протоколе, разработка методов атаки.

3.3. Разработка аппаратного и программного обеспечения лабораторного стенда.

3.4. Проверка реализуемости атак на лабораторном стенде.

3.5. Разработка рекомендаций по повышению устойчивости систем АСУ ТП к атакам на каналы передачи данных.

4. Консультанты по разделам ВКР:

раздел ВКР:\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

разрабатываемые вопросы: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ /\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_/

*должность, степень подпись И.О.Фамилия*

5. Дата выдачи задания: «?» марта 2017 г.

6. Срок представления на кафедру законченной ВКР: «?» июня 2017 г.

Руководитель ВКР

доцент кафедры ГИиИБ, к.т.н. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ /Д.Б. Жмуров/

*должность, степень подпись И.О.Фамилия*

Задание принял к исполнению \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ /Д.А. Резванов/

*подпись студента И.О.Фамилия студента*

**РЕФЕРАТ**

**Выпускная квалификационная работа специалиста:** ? c., ? рисунка, ? таблицы, ? источников.

Презентация: ? слайдов MicrosoftPowerPoint.

ИНФОРМАЦИЯ, АСУ ТП, MODBUS, STM32, МИКРОКОНТРОЛЛЕРЫ, RS-485, ПРОТОКОЛ ПЕРЕДАЧИ ДАННЫХ, UART, SPI.

Объектом разработки является лабораторный стенд для моделирования атак на канал передачи данных по протоколу Modbus на базе микроконтроллера семейства STM32.

Цель работы – анализ возможных направлений атак на протокол Modbus, разработка лабораторного стенда для моделирования атак, проверка реализуемости атак на лабораторном стенде, разработка рекомендаций по улучшению безопасности протокола Modbus.

Результатами данной работы являются лабораторный стенд, на котором можно моделировать атаки на канал передачи по протоколу Modbus, и новый протокол на основе Modbus, который поддерживает аутентификацию и устойчив к атакам подмены ведущего\ведомого устройств.

СОДЕРЖАНИЕ

[**ОПРЕДЕЛЕНИЯ, ОБОЗНАЧЕНИЯ И СОКРАЩЕНИЯ** 5](#_Toc513498445)

[**ВВЕДЕНИЕ** 6](#_Toc513498446)

[1 ПРОБЛЕМА БЕЗОПАСНОСТИ КАНАЛОВ ПЕРЕДАЧИ ДАННЫХ В АСУ ТП 7](#_Toc513498447)

[**1.1** **Понятие АСУ ТП** 7](#_Toc513498448)

[**1.2** **Актуальность проблемы безопасности АСУ ТП** 9](#_Toc513498449)

[**2** **ПРОТОКОЛ MODBUS** 13](#_Toc513498450)

[**2.1** **История и общие сведения о протоколе** 13](#_Toc513498451)

[**2.2** **Протокол Modbus RTU** 17](#_Toc513498452)

[**2.3** **Анализ возможных уязвимостей протокола Modbus** 24](#_Toc513498453)

[**3** **РАЗРАБОТКА ЛАБОРАТОРНОГО СТЕНДА** 28](#_Toc513498454)

[3.1 Общее описание лабораторного стенда 28](#_Toc513498455)

[**3.2** **Настройка стенда** 31](#_Toc513498456)

[**3.3** **Моделирование атак** 32](#_Toc513498457)

[**ЗАКЛЮЧЕНИЕ** 43](#_Toc513498458)

# **ОПРЕДЕЛЕНИЯ, ОБОЗНАЧЕНИЯ И СОКРАЩЕНИЯ**

**АСУ ТП** – автоматизированная система управления технологическим процессом.

**ИС** – интегральная схема.

**МК** – микроконтроллер.

**ПК** – персональный компьютер.

**ОС –** операционная система.

**UART** – универсальный асинхронный приёмопередатчик.

**SPI** – последовательный периферийный интерфейс.

**IIC** – последовательная ассиметричная шина для связи между ИС внутри электронных приборов.

# **ВВЕДЕНИЕ**

## ПРОБЛЕМА БЕЗОПАСНОСТИ КАНАЛОВ ПЕРЕДАЧИ ДАННЫХ В АСУ ТП

* 1. **Понятие АСУ ТП**

Автоматизированная система управления технологическим процессом (АСУ ТП) представляет собой комплекс технических и программных средств, которые предназначены для управления технологическим оборудованием на промышленных предприятиях. Структура системы приведена на рисунке 1.1. [1]

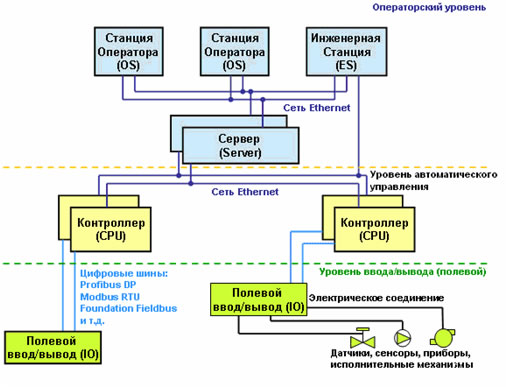


Рисунок 1.1 – структура АСУ ТП.

Как видно на рисунке, система делится на три уровня:

1. Нижний уровень (уровень ввода/вывода)
2. Средний уровень (уровень автоматического управления)
3. Верхний уровень (операторский уровень)

К нижнему уровню относится всё, чем управляют или с чего считывают показания. Например, электродвигатели, нагреватели, датчики температуры, давления и т.д. На рисунке 1.1 изображён пример исполнительного устройства.

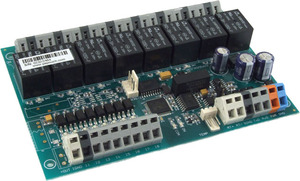


Рисунок 1.1 – релейный модуль с 8 релейными выходами и 8 логическими входами Quido RS 8/8.

Средний уровень – это промышленные логические контроллеры (ПЛК). Это центр всей системы, именно он в зависимости от заложенной в него программы опрашивает датчики, управляет механизмами, а также собирает в процессе работы данные и отправляет их на верхний уровень. На рисунке 1.2 изображён пример ПЛК.



Рисунок 1.2 – программируемый логический контроллер Siemens Simatic S7-300 с тремя модулями ввода.

Верхний уровень – это то, что позволяет общаться контроллеру и человеку. Обычно это ПК с установленным пакетом программ для сбора, обработки и отображения информации. Такой программный пакет сокращённо называется SCADA. На рисунке 1.3 изображён интерфейс SCADA-системы КРУГ-2000.

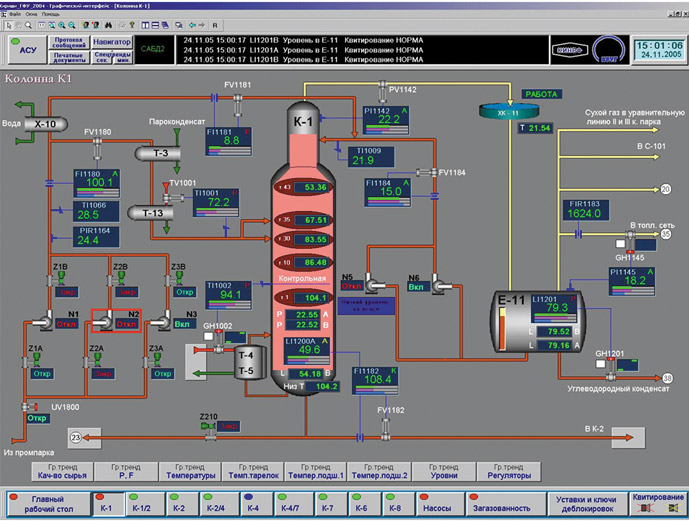


Рисунок 1.3 – интерфейс SCADA-системы КРУГ-2000. Установка газофракционирования

Для обмена между различными уровнями АСУ ТП используются каналы передачи данных. Между средним и верхним уровнем используется преимущественно сеть Ethernet, между средним и верхним – множество различных шин, таких как Modbus, Profibus, OPC и т.д.

* 1. **Актуальность проблемы безопасности АСУ ТП**

Так как в АСУ ТП широко используется сеть Ethernet, многие компоненты системы присутствуют в глобальной сети. [2] Чаще всего это SCADA-системы, на них приходится 70% компонентов АСУ ТП, которые доступны в глобальной сети. Ещё 27% - это программируемые логические устройства. Оставшиеся 3% - это остальные различные сетевые устройства (например, устройства нижнего уровня, которые используют сеть Ethernet). Соотношение компонентов АСУ ТП, которые присутствуют в глобальной сети, представлено на рисунке 1.4.

Рисунок 1.4 – соотношение компонентов АСУ ТП в глобальной сети

Как минимум 42% доступных через Интернет компонентов АСУ ТП содержат уязвимости. Ещё 41% находится в зоне риска. Таким образом, доля систем, безопасность которых достоверно подтверждена, составляет всего лишь 17%. Если смотреть статистику по странам, то наибольшая доля уязвимых систем, которые видны из сети Интернет, приходится на Швейцарию, там уязвимы все подобные системы. В России, например, уязвима ровно половина (50%) таких систем АСУ ТП.

Резкий рост интереса к проблемам безопасности АСУ ТП произошёл в 2010 году, когда был обнаружен компьютерный червь Stuxnet. Червь использовал четыре уязвимости системы Microsoft Windows, через которые распространялся первоначально через сменные носители, а затем по локальной сети и через Интернет. Незамеченным он оставался благодаря украденным настоящим цифровым подписям компаний Realtek и JMicron. Червь отличается высокой избирательностью, целенаправленно ищет и заражал компьютеры с установленной SCADA-системой WinCC и системой программирования ПЛК Step 7. Вирус заражет файлы проекта, перехватывая на себя связь между WinCC и ПЛК при помощи подмены динамической библиотеки связи на свою. Таким образом, червь может замаскировать своё присутствие и установить руткит непосредственно в ПЛК. Руткит атаковет только ПЛК Siemens Simatic S7-300, к которым подключены частотные преобразователи только от двух поставщиков: Vacon и Fararo Paya. Кроме того, он атакует только те приводы, которые вращаются с частотой от 807 Гц до 1210 Гц, в этом диапазоне как раз вращаются газовые центрифуги для разделения изотопов. Когда все критерии выполнены, руткит изменяет скорость вращения до 1410 Гц, затем до 2 Гц, а затем до 1064 Гц, при этом системе сбора данных сообщается нормальное значение. Такие резкие колебания скорости вывели порядка 1000 центрифуг из строя. Из-за специфических требований к системе, существует предположение, что Stuxnet является специализированной разработкой спецслужб Израиля и США, направленной против ядерных проектов Ирана. [3]  
 Всё это привело к тому, что с 2010 по 2012 год было обнаружено в 20 раз больше уязвимостей в системах АСУ ТП, чем за период с 2005 по 2010 год.

Но при этом существует ещё один путь проникновения в систему, которому не уделяется достаточное внимание – через каналы связи нижнего уровня с верхним. Такой путь гораздо более трудоёмкий, так как в большинстве случаев требуется доступ непосредственно в охраняемый периметр предприятия, но от этого не менее опасный, ведь в случае с Stuxnet первичное заражение произошло через USB-носитель, что автоматически означает доступ к периметру на самой ранней стадии распространения. Атаки на данные каналы менее распространённые, но всё же встречаются. Например, в январе 2016 году на GitHub в открытый доступ был опубликован исходный код фреймворка для тестирования на проникновение под названием smod. Данный фреймворк был разработан для оценки безопасности протокола Modbus. В итоге это привело к тому, что за 2016 год количество атак на АСУ ТП, использующие протокол Modbus, выросло на 110% по сравнению с 2015 годом. [4]

Из протоколов передачи данных между нижним и верхним уровнями самыми распространёнными являются Modbus, Profibus и OPC. На рисунке 1.5 представлена распространённость данных протоколов. У Modbus (RTU+TCP) и Profibus примерно равные доли по 33%. Доля OPC составляет 25%, оставшиеся 9% приходятся на различные другие протоколы.

Рисунок 1.5 – распространённость промышленных протоколов передачи данных

Из двух лидеров рынка, Modbus и Profibus, первый более простой для понимания и реализации. В данной работе в качестве цели для анализа и моделирования атак будет рассматриваться последовательная (RTU) версия протокола Modbus, так как TCP-версия протокола использует в качестве среды передачи Ethernet и атаки на неё осуществляются теми же путями, что и на каналы связи между верхним и средним уровнями.

1. **ПРОТОКОЛ MODBUS**
   1. **История и общие сведения о протоколе**

Modbus – открытый коммуникационный протокол, базирующийся на архитектуре master-slave (ведущий-ведомый). Применяется в промышленности для связи между электронными устройствами через последовательные линии связи RS-232, RS-485, RS-422 и сети TCP/IP.

Протокол был разработан компанией Modicon (в настоящее время – Schneider Electric) для использования в своих программируемых логических контроллерах. Первая версия спецификации была опубликована в 1979 году. Стандарт протокола описывает лишь структуру обмена сообщениями, без привязки к конкретному физическому уровню. Но в большинстве случаев используются последовательные линии и сеть Ethernet.

Благодаря своей гибкости, но в то же время и простоте реализации на слабых вычислительных платформах (таких как микроконтроллеры), протокол Modbus получил крайне широкое распространение в сфере промышленной автоматики.

Спецификация протокола устанавливает три основных режима передачи данных: Modbus ASCII (American Standard Code for Information), Modbus RTU (Remote Terminal Unit) и Modbus TCP (Transmission Control Protocol). Первые два используются для линий последовательной связи, а последний – для сети TCP/IP. Основное отличие – в режиме Modbus TCP в сообщении отсутствует контрольная сумма, так как контроль целостности сообщения обеспечивается контрольной суммой в заголовке сегмента TCP.

В структуре сети допускается наличие только одного ведущего устройства и до 247 ведомых устройств. Каждое ведомое устройство имеет свой уникальный (в пределах конкретной сети) адрес от 1 до 247. Ведущее устройство не имеет адреса вообще. Адреса 248 – 255 зарезервированы, адрес 0 зарезервирован для широковещательной передачи сообщений. Сообщение с этим адресом должны обрабатывать все устройства, но отвечать на него не нужно. По спецификации протокола Modbus, инициировать передачу сообщения может только ведущее устройство, ведомое устройство делать этого не имеет права. Поэтому если ведомое устройство зафиксировало на своей стороне какое-либо событие (например, сработал концевой выключатель сигнализации), то ведущее устройство об этом не узнает до тех пор, пока само не отправит сообщение с запросом статуса этого события.

Сообщения, исходящие от ведущего устройства, называются запросами, а ответные сообщения от ведомого – ответами. Обмен ведётся в полудуплексном режиме, это значит, что в любой момент времени передачу ведёт только одно устройство.

* 1. **Физический уровень RS-485**

В качестве физического уровня для передачи сообщений в лабораторном стенде используется RS-485, как наиболее распространённый в промышленности.

RS-485 – стандарт физического уровня для асинхронного интерфейса (полное название ANSI TIA/EIA-485-A:1998). В качестве физической среды для передачи и приёма данных используется одна витая пара, иногда с экранирующей оплёткой или общим проводом. Соответственно, используемая сетевая топология – «общая шина», структура которой показана на рисунке 2.1.

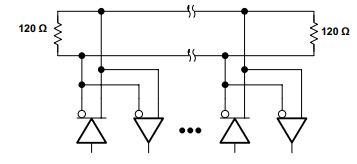


Рисунок 2.1 – структура топологии «общая шина».

По стандарту, максимальная длина линии может составлять 1200 метров. При такой большой длине возникают эффекты длинных линий, так как длинную витую пару можно представить как распределённую ёмкость и индуктивность. Так как витая пара на всей своей протяжённости имеет одинаковую конструкцию, следовательно, и одинаковые емкостные и индуктивные свойства на единицу длины, то данное свойство характеризуется таким параметром как волновое сопротивление. Поэтому, чтобы ослабить отражение сигнала от концов витой пары, в начале и в конце линии между проводниками пары устанавливается так называемый терминатор – согласующий резистор, сопротивление которого равно волновому сопротивлению витой пары. В большинстве случаев используются резисторы номиналом 120 Ом. На рисунке 2.2 представлены осциллограммы сигнала на стороне приёмника, в обоих случаях использовался приёмопередатчик ADM3485, витая пара с волновым сопротивлением 120 Ом и резистор-терминатор номиналом 120 Ом со стороны передатчика. Сигнал слева снят с резистором-терминатором номиналом 54 Ом со стороны приёмника, сигнал справа – с резистором номиналом 120 Ом.

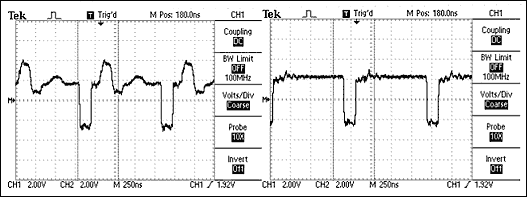


Рисунок 2.2 – осциллограммы сигнала на стороне приёмника.

Как видно на осциллограммах, в случае установки резистора-терминатора неверного номинала возникает рассогласование, которое становится причиной отражений сигнала. В результате корректный приём становится невозможным.

Режим передачи – дифференциальный. Это означает, что за логический 0 или логическую 1 принимается не напряжение относительно общего провода, а напряжение между проводниками пары. Сделано это в первую очередь для защиты от внешних помех. Так как проводники пары скручены между собой, то внешняя помеха будет наведена на оба проводника с одинаковым знаком, амплитудой и фазой. Такая помеха называется синфазной. В итоге, разница напряжений между линиями пары останется неизменной и помеха не окажет никакого влияния на передаваемые данные. Линии пары обозначаются как A - неинвертирующая линия, B инвертирующая линия и С – необязательный общий провод. За логическую 1 принимается положительное напряжение (A-B) > +200мВ, за логический 0 – отрицательное напряжение (A-B) < -200мВ. Состояние, когда напряжение меньше 200мВ, стандартом не определено. При этом стандарт ограничивает допустимый диапазон напряжений линий A и B относительно общего провода диапазоном от -7В до +12В.

Из-за того, что любой реальный, а не теоретический, приёмник и передатчик имеют конечный импеданс, максимальное количество устройств в одной линии ограничено. Для облегчения проектирования сети RS-485 была введена гипотетическая единица нагрузки, так называемая «единица нагрузки». По стандарту, максимальная нагрузочная способность линии – 32 единицы нагрузки. Количество единиц нагрузки указывается в технической документации приёмопередатчика. Например, приёмопередатчик ADM3485 нагружает линию как 1 единичная нагрузка, а MAX487 – как ¼ единичной нагрузки. Для данного конкретного примера, если вся линия будет состоять из однотипных приёмопередатчиков, то всего в линию можно будет включить 32 устройства с ADM3485 и 128 устройств с MAX487. Для обхода ограничения на максимальное количество устройств в линии может применяться специальное устройство – повторитель интерфейса RS-485. Повторитель считывает сигнал с одной линии и передаёт его в другую и наоборот. Помимо этого, некоторые повторители могут обладать такой полезной функцией как гальваническая развязка – сигнал передаётся из одной линии в другую без непосредственного электрического контакта. В итоге, цепи оказываются электрически разделёнными, что ограничивает передачу энергии из одной цепи в другую, но возможна передача сигнала.

* 1. **Протокол Modbus RTU**

Режим RTU (Remote Terminal Unit) является самым наиболее часто применяемым режимом Modbus для последовательных сетей. Главное преимущество RTU перед ASCII – компактность. В режиме ASCII каждый байт данных кодируется двумя байтами. Например, для передачи байта 0x45 в режиме ASCII необходимо передать два байта, ASCII-код символов ‘4’ и ‘5’. Однако, есть и преимущество – для устройств, которые не умеют отсчитывать малые временные интервалы, ASCII-режим становится единственно возможным, так как конец пакета определяется не по интервалу тишины на линии, а по последовательности CR+LF. В промышленности датчики и исполнительные устройства реализуются, в основном, на микроконтроллерах. Так как большинство микроконтроллеров имеют в своём составе минимум один аппаратный таймер, данное преимущество ASCII-режима не имеет особого значения, на первый план выходит потребление оперативной памяти. В этом плане Modbus RTU выигрывает у Modbus ASCII примерно в два раза.

Общая структура сообщения Modbus RTU приведена на рисунке 2.3.

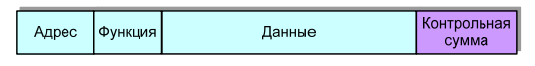


Рисунок 2.3 – структура сообщения Modbus RTU.

Адрес – адрес ведомого устройства, которому предназначен запрос.

Функция – поле длиной 1 байт, которое говорит ведомому, какое именно действие от него требуется.

Данные – информация, необходимая ведомому устройству для выполнения заданной ведущим устройством функции или данные, передаваемые ведомым в ответ. Формат данного поля зависит от конкретной функции.

Контрольная сумма – поле длиной 2 байта, контрольная сумма формата CRC16-Modbus.

Максимальная длина сообщения ограничена 256 байтами. [5]

Спецификация протокола определяет четыре типа данных:

1. Флаги – 1 бит, чтение и запись
2. Дискретные входы – 1 бит, только чтение
3. Регистры хранения – 16 бит, чтение и запись
4. Регистры ввода – 16 бит, только чтение

Флаги – 1-битная структура данных, доступная для чтения и записи. Основное применение - управление устройствами, которые могут находится только в двух состояниях, такие как электромагнитное реле или светодиод.

Дискретные входы – 1-битная структура данных, доступная только для чтения. Основное применение – внешние датчики, которые могут находится в двух состояниях, такие как концевые выключатели.

Регистры хранения – 16-битная структура данных, доступная для чтения и записи. Основное применение – различные настройки ведомого устройства, такие как адрес, скорость обмена и т.д.

Регистры ввода – 16-битная структура данных, доступная только для чтения. Основное применение – хранение показаний различных датчиков, например, датчик температуры, влажности, давления и т.д.

У каждого элемента есть адрес, адресация начинается с 0. Адресация внутри своей группы уникальна. Например, регистр хранения с адресом 1 и регистр ввода с тем же адресом – это разные регистры. Соответственно, и доступ к ним осуществляется с помощью разных кодов функций.

Рассмотрим коды функций. Они делятся на 5 групп, первые три группы называются стандартными:

1. Чтение данных
2. Запись одного значения
3. Запись нескольких значений
4. Пользовательские функции
5. Зарезервированные функции

Для пользовательских кодов функций выделены два диапазона: 0x41 – 0x48, 0x64 – 0x6E. При этом не гарантируется, что какое-то другое устройство не будет использовать тот же код для выполнения другой функции.

В категорию зарезервированных попадают коды, которые не являются стандартными, но уже используются устройствами, которые производят другие различные компании. Это коды 0x09, 0x0A, 0x0D, 0x0E, 0x29, 0x2A, 0x5A, 0x5B, 0x7D, 0x7E, 0x7F.

Чтение данных осуществляется с помощью следующих кодов функций:

0x01 – чтение значений из нескольких регистров флагов

0x02 – чтение значений из нескольких дискретных входов

0x03 – чтение значений из нескольких регистров хранения

0x04 – чтение значений из нескольких регистров ввода

Запрос состоит из адреса первого элемента и количества, которое необходимо прочитать. Так как длина пакета данных ограничена, максимальное количество входов, которое можно прочитать за один раз, ограничено 2000. Максимальное количество регистров, доступное для чтения за один раз – 125. Структура запроса приведена в таблице 2.1.

Таблица 2.1 – структура запроса на чтение данных

|  |  |  |
| --- | --- | --- |
| Код функции | 1 байт | 0x01, 0x02, 0x03, 0x04 |
| Начальный адрес | 2 байта | 0x0000 – 0xFFFF |
| Количество входов для чтения | 2 байта | Для 0x01, 0x02 – от 1 до 2000 (0x07D0). Для 0x03,0x04 – от 1 до 125 (0x007D) |

В ответе передаются запрошенные данные. Значения регистров хранения и дискретных входов передаются по два байта на один регистр, порядок байтов MSB (Most Signification Byte) – то есть, старший байт идёт первым. Флаги и дискретные входы передаются в упакованном виде, один флаг или вход занимает 1 бит. Единица – включённое состояние, ноль – выключенное. Младший бит первого байта содержит значение первого запрошенного флага или входа, второй бит – второго и т.д. Первый бит второго байта содержит значение девятого флага или входа. Если запрошенное количество не кратно 8 – лишние биты заполняются нулями.

Структура ответа показана в таблице 2.2

Таблица 2.2 – структура ответа на запрос чтения данных.

|  |  |  |
| --- | --- | --- |
| Код функции | 1 байт | 0x01, 0x02, 0x03, 0x04 |
| Количество байт | 1 байт | Для 0x01, 0x02:  N = количество выходов/8. Если остаток > 0, то N = N+1  Для 0x03, 0x04:  N = 2\*n, где n – количество регистров |
| Данные | N байт |  |

Запись одного значения осуществляется с помощью следующих кодов функций:

0x05 – запись значения одного флага

0x06 – запись значения в один регистр хранения

Запрос состоит из адреса элемента и устанавливаемого значения. Для флага значение 0xFF00 означает включенное состояние, 0x0000 – выключенное. В случае записи других значений – ведомое устройство сообщит об ошибке. В случае успешного выполнения команды – ведомое устройство вернёт копию запроса. Структура запроса представлена в таблице 2.3.

Таблица 2.3 – структура запроса записи одного значения

|  |  |  |
| --- | --- | --- |
| Код функции | 1 байт | 0x05, 0x06 |
| Адрес флага или регистра | 2 байта | 0x0000 – 0xFFFF |
| Записываемые данные | 2 байта |  |

Запись нескольких значений осуществляется с помощью следующих кодов функций:

0x0F – запись значений нескольких флагов

0x10 – запись значений в несколько регистров хранения

Запрос состоит из адреса первого элемента, количества записываемых элементов, количества передаваемых байт и самих данных. Упаковка данных полностью аналогична упаковке данных в командах чтения. Ответ состоит из начального адреса и количества записанных элементов. Структура запроса представлена в таблице 2.4, структура ответа в таблице 21.5.

Таблица 2.4 – структура запроса записи нескольких значений

|  |  |  |
| --- | --- | --- |
| Код функции | 1 байт | 0x0F, 0x10 |
| Начальный адрес | 2 байта | 0x0000 – 0xFFFF |
| Количество записываемых элементов | 2 байта | Для 0x0F – от 1 до 2000 (0x07D0). Для 0x10 – от 1 до 123 (0x007B) |
| Количество байт | 1 байт | Для 0x0F:  N = количество выходов/8. Если остаток > 0, то N = N+1  Для 0x10:  N = 2\*n, где n – количество регистров |
| Данные | N байт |  |

Таблица 2.5 – структура ответа на запрос записи нескольких значений

|  |  |  |
| --- | --- | --- |
| Код функции | 1 байт | 0x0F, 0x10 |
| Начальный адрес | 2 байта | 0x0000 – 0xFFFF |
| Количество записываемых элементов | 2 байта | Для 0x0F – от 1 до 2000 (0x07D0). Для 0x10 – от 1 до 123 (0x007B) |

В результате обмена могут возникать некоторые ошибки. Ошибки могут быть связаны с искажениями при передаче данных или в логике сообщений (когда запрос принят верно, но не может быть выполнен).

Ошибки первого типа обнаруживаются при вычислении контрольной суммы. Если вычисленная сумма не совпадает с указанной в сообщении – сообщение игнорируется. Для предотвращения зависания системы в целом, ведущее устройство должно отсчитывать определённый таймаут. По истечении таймаута, устройство может сделать вывод, что ведомое устройство не присутствует на линии, либо произошли искажения при передаче.

При ошибках второго типа ведомое устройство посылает ведущему специальное сообщение об ошибке. Признаком того, что текущее сообщение сообщает об ошибке, является установленный в единицу старший бит кода функции. Следующий за кодом функции байт сообщает о типе ошибки. Стандартные коды ошибок перечислены в таблице 2.6.

Таблица 2.6 – список стандартных кодов ошибок

|  |  |  |
| --- | --- | --- |
| Код ошибки | Расшифровка | Примечание |
| 0x01 | Принятый код функции не поддерживается устройством |  |
| 0x02 | Адрес данных, указанный в запросе, недоступен |  |
| 0x03 | Недопустимое значение в поле данных | Данная ошибка будет вызываться, если, например, для функции записи значения одного флага (0x05) указать значение, отличное от 0xFF00 или 0x0000 |
| 0x04 | Невосстанавливаемая ошибка |  |
| 0x05 | Ведомое устройство приняло запрос, но на выполнение требуется время | Данная ошибка генерируется для того, чтобы ведомое устройство не генерировало ошибку таймаута ответа |
| 0x06 | Ведомое устройство занято | При получении данной ошибки ведущее устройство должно повторить свой запрос позднее |

* 1. **Анализ возможных уязвимостей протокола Modbus**

На рисунке 2.4 показана временная диаграмма одного сеанса обмена (запрос ведущего + ответ ведомого) в идеальном случае, когда ведомому не требуется время на обработку). t3.5 – интервал тишины, после которого пакет считается принятым.



Рисунок 2.4 – временная диаграмма обмена в идеальном случае

Но указанная выше ситуация в реальной системе не будет наблюдаться никогда по причине того, что ведомому необходимо время на проверку контрольной суммы, разбор принятого пакета, опрос датчиков\входов, формирование ответа и т.д. Поэтому реальная картина будет соответствовать той, что показана на рисунке 2.5. tобработки – время, необходимое ведомому на обработку запроса. Пример: ведомое устройство имеет в своей основе 32-разрядный микроконтроллер STM32 с ядром Cortex-M3, работающим на частоте 72МГц, обмен ведётся на скорости 9600. Обработка запроса на запись 17 регистров флагов занимает ? мкс, что соответствует ? символов на данной скорости обмена. В случае использования 8 и 16-разрядных архитектур это время может быть больше в несколько раз.

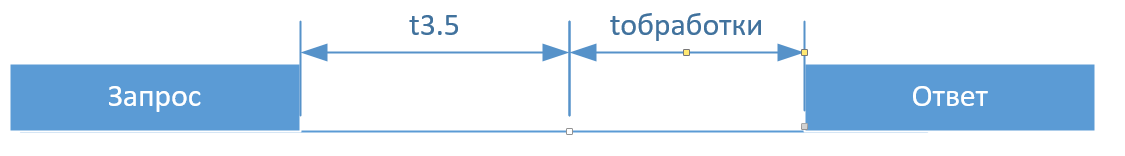


Рисунок 2.5 – временная диаграмма обмена в реальной системе

Далее необходимо проанализировать, как можно вмешаться в работу системы. Так как физическая среда передачи – витая пара, способ только один – физически подсоединиться к линии. Если линия имеет большую протяжённость (по стандарту до 1200 метров, но с повторителями может быть и гораздо больше), то поиск вторгшегося устройства становится очень непростой задачей. Далее необходимо понять, как именно вмешиваться в обмен.

Первый, самый очевидный способ, который не требует сложной разработки – непрерывный спам случайными данными. В данном случае, периодически будет возникать ситуация, когда ведущее или ведомое устройство выставляет на линии один логический уровень, а злоумышленник – другой. В результате во время передачи этого бита разница напряжений между линиями A и B будет примерно в районе 0 В, что соответствует неопределённому состоянию. В итоге, обмен непременно будет нарушен. Также, этот способ атаки можно рассматривать как атаку на переполнение буфера. Поступающие данные хранятся в буфере, который располагается в оперативной памяти ведущего или ведомого. Если в ПО устройства отсутствует проверка на предмет того, не превысило ли принятое количество байт максимальную длину пакета, то после 256 байт произойдёт переполнение буфера, так как устройство злоумышленника отправляет данные непрерывно, без интервалов тишины, предусмотренных по стандарту. Защитой от этого может служить прекращение приёма после 256 принятых байт, либо использование 8-битной переменной для хранения количества принятых байт. Во втором случае произойдёт переполнение переменной и отсчёт количества принятых байт начнётся с нуля. Временная диаграмма обмена для этого случая показана на рисунке 2.6.



Рисунок 2.6 – временная диаграмма атаки «спам случайными данными»

Второй способ основан на использовании интервала t3.5+tобработки, называется «ложный ведомый». Действительно, если нам примерно известно среднее время обработки запроса настоящим ведомым, то можно взять микроконтроллер с производительностью сильно превосходящей производительность ведомого и отправить ложный ответ в то время, пока настоящий ведомый будет обрабатывать запрос. Ответ же настоящего ведомого будет отброшен ведущим, так как он уже принял ответ от ложного ведомого. Временная диаграмма обмена для этого случая показана на рисунке 2.7. t1 – время обработки запроса ложного ведомого.



Рисунок 2.7 – временная диаграмма атаки «ложный ведомый»

Данный тип атаки является самым опасным с точки зрения технологического процесса. Допустим, ведомое устройство – поплавок с концевым выключателем, который контролирует наполнение какой-либо ёмкости. Ведущее устройство постоянно опрашивает состояние выключателя и в зависимости от него выдаёт команды на насос. Если в данном случае осуществить атаку «ложный ведомый», то можно заставить ведущего думать, что концевой выключатель не сработал. Тогда ведущее устройство будет продолжать выдавать команды на насос, что при отсутствии защиты от перелива будет означать разлив жидкости.

Третий способ основан на том, что запросы в реальной линии идут не непрерывно, а периодически. Между сеансами обмена между ведущим и ведомым случаются длинные временные интервалы, в течении которых можно успеть отправить запрос ведомому и получить от него ответ. Соответственно, способ атаки называется «ложный ведущий». Временная диаграмма данной атаки приведена на рисунке 2.8.

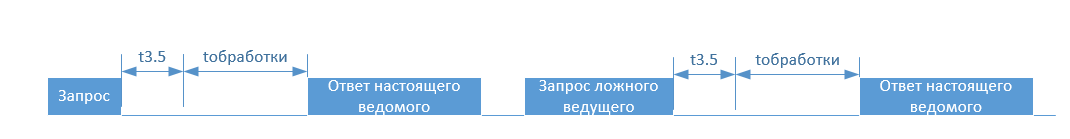


Рисунок 2.8 – временная диаграмма для атаки «ложный ведущий»

Опасность данной атаки состоит в том, что если точно известен тип ведомого устройства и оно позволяет изменять настройки обмена (например, скорость) через регистры хранения, то ложный ведущий может перезаписать эти самые настройки. В результате, настоящий ведущий уже не сможет установить связь с ведомым при следующем сеансе обмена.

Для моделирования данных атак был разработан лабораторный стенд.

# **РАЗРАБОТКА ЛАБОРАТОРНОГО СТЕНДА**

## Общее описание лабораторного стенда

Общая структура лабораторного стенда показана на рисунке 3.1.

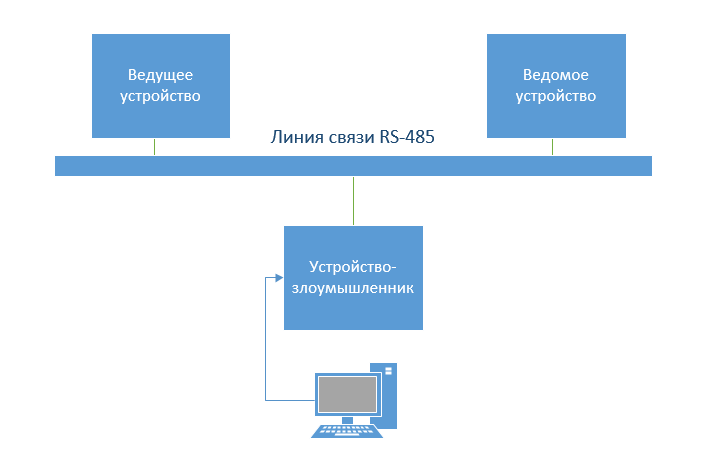


Рисунок 3.1 – структура лабораторного стенда

Как видно на рисунке 3.1, стенд состоит из трёх устройств, связанных одной общей информационной шиной RS-485. В целях снижения временных и материальных затрат, ведущее и ведомое устройство – это два экземпляра одного и того же устройства, необходимый режим выбирается при начальной настройке.

Общая схема ведущего\ведомого устройства приведена на рисунке 3.2.



Рисунок 3.2 – общая схема ведущего\ведомого устройства

Центром устройства является микроконтроллер фирмы STMicroelectronics, семейства STM32, STM32F107VCT6. Микроконтроллер имеет на борту одно 32-битное ядро Cortex-M3, 256кБ Flash-памяти под программу, 64кБ оперативной памяти, а также целый спектр различной периферии, от IIC, SPI и UART до Ethernet 10/100.

Для отображения информации на борту устройства имеется цветной жидкокристаллический дисплей с разрешением 240x320 пикселей, диагональю 2.2 дюйма и контроллером ILI9341. Обмен между дисплеем и микроконтроллером осуществляется по последовательной шине SPI. В качестве библиотеки графического интерфейса на стороне микроконтроллера используется библиотека STemWin. Именно она диктует основные требования к количеству оперативной и Flash-памяти. Разработчик библиотеки вместе с библиотекой предоставляет проект для симуляции интерфейса на ПК с использованием среды Visual Studio. Поэтому, для определения количества требуемой памяти микроконтроллера, графический интерфейс был разработан и запущен на ПК. Таким образом, для конкретного графического интерфейса потребовалось 20кБ оперативной и 100кБ Flash-памяти, что составляет примерно 31% от имеющейся оперативной и 39% от имеющейся Flash-памяти.

Для отладки при разработке прошивки в составе устройства имеется внутрисхемный программатор-отладчик ST-Link v2. Интерфейс связи с компьютером – USB, для защиты компьютера и устройства от разницы потенциалов общего провода и, соответственно, от протекания выравнивающих токов, интерфейс гальванически изолирован с помощью микросхемы фирмы Analog Devices ADuM4160.

Для работы с шиной RS-485 используется приёмопередатчик фирмы Analog Devices ADM3485. Данный приёмопередатчик может работать на скоростях до 10Мбит\с, входное сопротивление равно 12 кОм, таким образом, на одной шине может располагаться до 32 таких приёмопередатчиков. В целях защиты устройства от неблагоприятных внешних воздействий со стороны шины RS-485, линия связи между микроконтроллером и приёмопередатчиком гальванически развязана с помощью оптронов. Таким образом, в случае мощных внешних наводок, приёмопередатчик может выйти из строя, но микроконтроллер и остальные компоненты устройства останутся невредимыми.

Также на устройстве имеются датчики, а именно – датчик атмосферного давления BMP180 фирмы Bosch, датчик температуры и влажности SHT2 фирмы Sensirion. BMP180 способен измерять давление в диапазоне от 30000 до 110000 Па, обладая при этом крайне низким шумом в 2 Па и малыми размерами корпуса, всего 3.6x3.8 мм. SHT21 способен измерять температуру в диапазоне от -40 до 125 °C с точностью 0.3 °C и относительную влажность воздуха в диапазоне от 0 до 100% с точностью 2%.

Для моделирования различных исполнительных устройств на плате устройства размещены 15 светодиодов, объединённых в 3 группы по 5 светодиодов, и 7 выходов типа «открытый коллектор», управляемые с помощью драйвера ULN2003. Максимальный выходной ток одного выхода равен 500 мА.

Для применения в будущем в составе устройства имеется Bluetooth BLE на основе чипа NRF8001 фирмы Nordic Semiconductor, а также преобразователь USB-UART на основе чипа CP2102 фирмы Silicon Labs.

Как видно, ведущее\ведомое устройство располагает крайне широким набором периферийных устройств, что вполне позволяет использовать устройство не только в качестве компонента лабораторного стенда, но и как, например, устройство для обучения разработке программ под микроконтроллеры семейства STM32.

Теперь рассмотрим устройство-злоумышленника. Его общая схема приведена на рисунке 3.3.

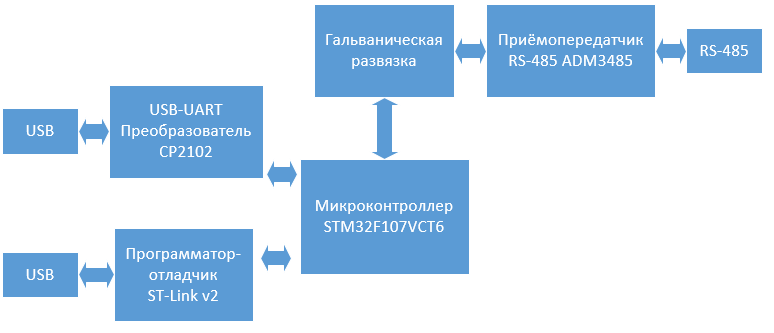


Рисунок 3.3 – общая схема устройства-злоумышленника.

Как видно, устройство-злоумышленник по своей структуре сильно проще. Хотя, теперь нет особых требований к количеству памяти, для унификации был применён тот же микроконтроллер, что и в ведущем\ведомом устройстве. Приёмопередатчик также остался тем же. Так как теперь устройство питается непосредственно от ПК, отпала необходимость в гальванической развязке интерфейса USB. Устройство-злоумышленник запрограммировано так, что любой байт на линии отправляется в USB-UART преобразователь. Таким образом, в специальной программе-терминале на ПК в любой момент времени можно наблюдать ситуацию на линии.

* 1. **Настройка стенда**

Прежде, чем начать работу по моделированию атак, необходимо произвести первоначальную настройку ведущего и ведомого устройств.

Адрес задаётся в диапазоне от 1 до 247, скорость обмена – одно из фиксированных значений. Возможные значения скорости (в бод\с): 1200, 2400, 4800, 9600, 19200, 38400, 57600, 115200. Наиболее устоявшаяся в промышленности скорость – 9600 бод\с. Помимо адреса и скорости обмена, у каждого из устройств есть индивидуальные настройки. Для ведущего устройства задаётся период обмена – временной промежуток между двумя запросами. Для ведомого устройства задаётся время обработки – время, которое устройство будет ждать, прежде чем приступить к обработке запроса. По сути, это время tобработки, показанное на рисунке 2.5. Оба параметра задаются в пределах от 1 до 1000 мс с шагом в 1 мс. При моделировании атак использовались следующие настройки:

Адрес – 0x09

Скорость обмена – 9600 бод\с

Период обмена – 400 мс

Время обработки – 160 мс

* 1. **Моделирование атак**

Теперь непосредственно попробуем смоделировать атаки. В стенде ведущее устройство постоянно опрашивает ведомое и в зависимости от показаний датчика температуры управляет «нагревателем» - светодиодом. Если температура выше 32 °C, «нагреватель» выключается. Когда температура падает ниже 31 °C - «нагреватель» включается. Таким образом, между 31 и 32 °C образуется гистерезис – зона, в которой «нагреватель» не меняет своё состояние.

Первой атакой идёт спам случайными данными, так как это наиболее лёгкая в реализации атака. Случайные данные генерируются с помощью программного генератора псевдослучайных чисел на основе регистра сдвига с линейной обратной связью (LFSR). Блок-схема алгоритма атаки показана на рисунке 3.4.

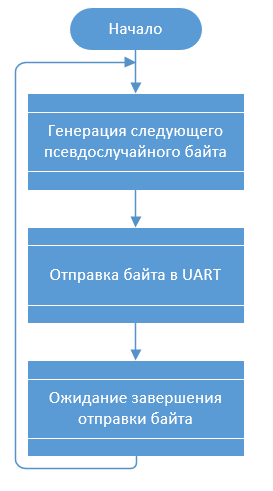


Рисунок 3.4 – блок-схема алгоритма атаки «спам случайными данными»

Состояние ведущего и ведомого устройств до атаки показано на рисунке 3.5. Слева – ведомое устройство, справа – ведущее.

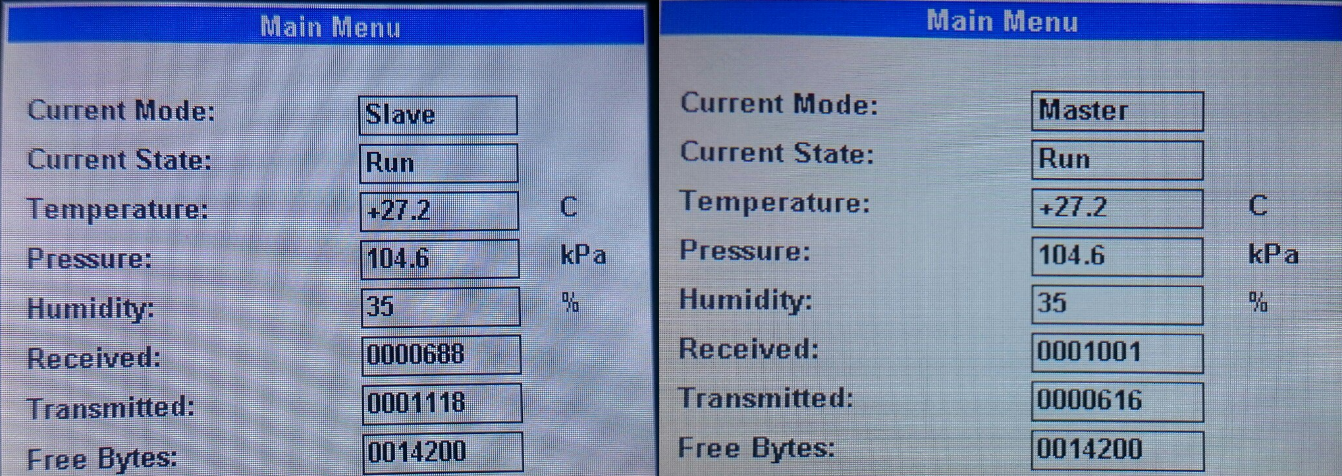


Рисунок 3.5 – состояние устройств до атаки «спам случайными данными»

Как видно на рисунке, данные датчиков у ведущего и ведомого устройств совпадают. Теперь включим устройство-злоумышленник и зажмём пальцем датчик температуры на ведомом устройстве. Состояние устройств во время атаки показано на рисунке 3.6. Слева – ведомое устройство, справа – ведущее.

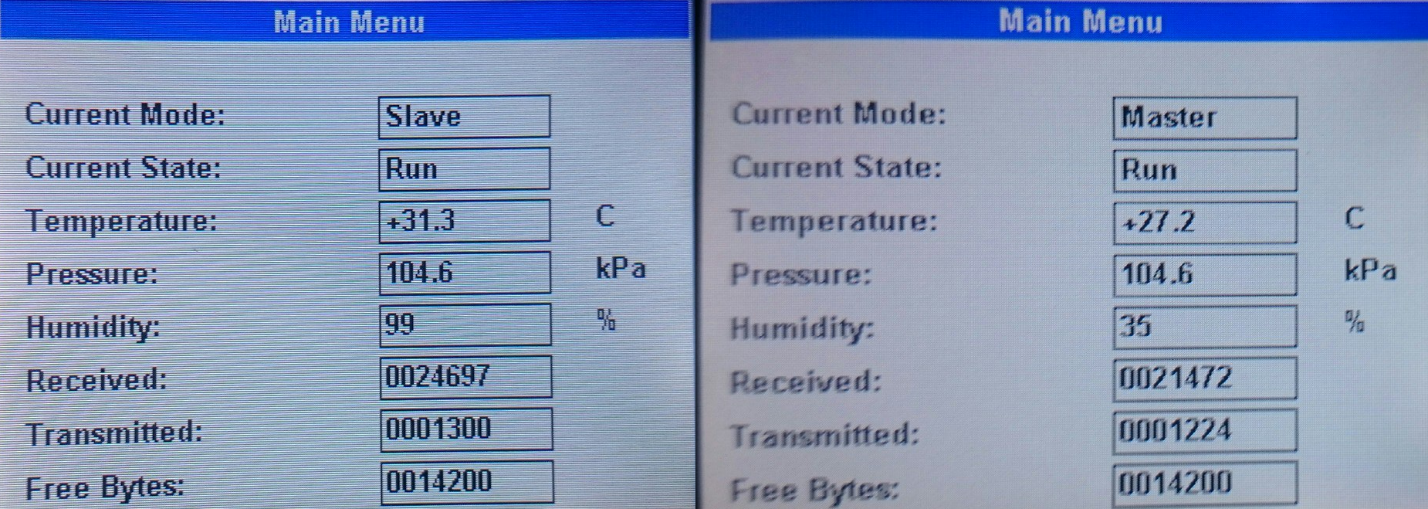


Рисунок 3.6 - состояние устройств во время атаки «спам случайными данными»

На рисунке видно, что обмен между устройствами прекратился, ведущий не получает показаний датчиков, а счётчик полученных байтов очень быстро растёт. Влажность стала 99% из-за того, что окно датчика температуры было закрыто пальцем. Переполнения буфера в данном случае не произойдёт, так как в обоих устройствах присутствует проверка, не превысило ли число принятых байт максимальную длину пакета.

Следующей идёт атака «ложный ведомый». Блок-схема атаки представлена на рисунке 3.7.

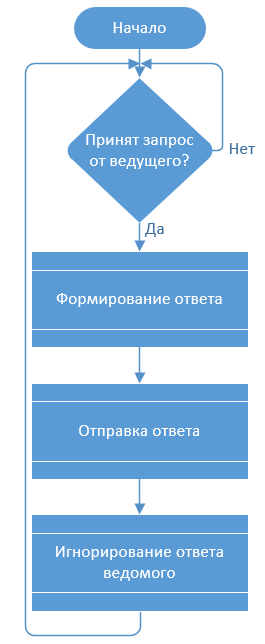


Рисунок 3.7 – блок-схема атаки «ложный ведомый»

Состояние ведущего и ведомого устройств до атаки показано на рисунке 3.8. Слева – ведомое устройство, посередине – ведущее, справа – состояние светодиода, который имитирует нагреватель.

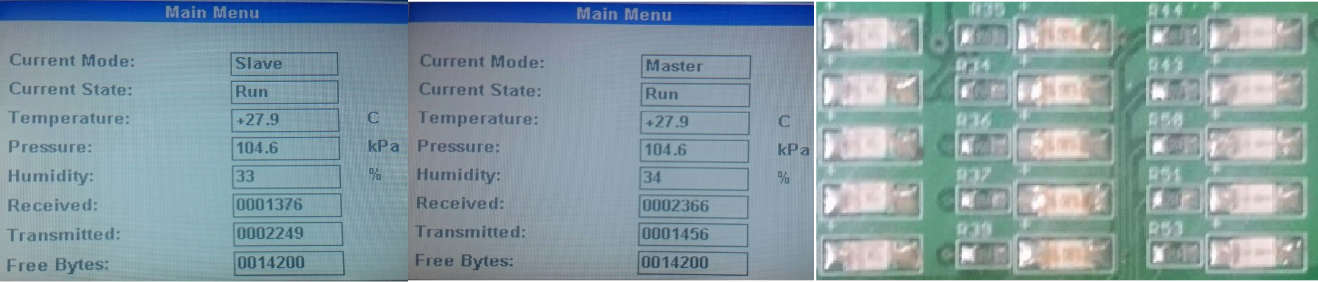


Рисунок 3.8 – состояние устройств до атаки «ложный ведомый»

Как видно на рисунке, температура равна 27.9 °C, поэтому светодиод выключен. Теперь включим устройство-злоумышленник. Состояние устройств во время атаки показано на рисунке 3.9. Слева – ведомое устройство, посередине – ведущее, справа – состояние светодиода, который имитирует нагреватель.

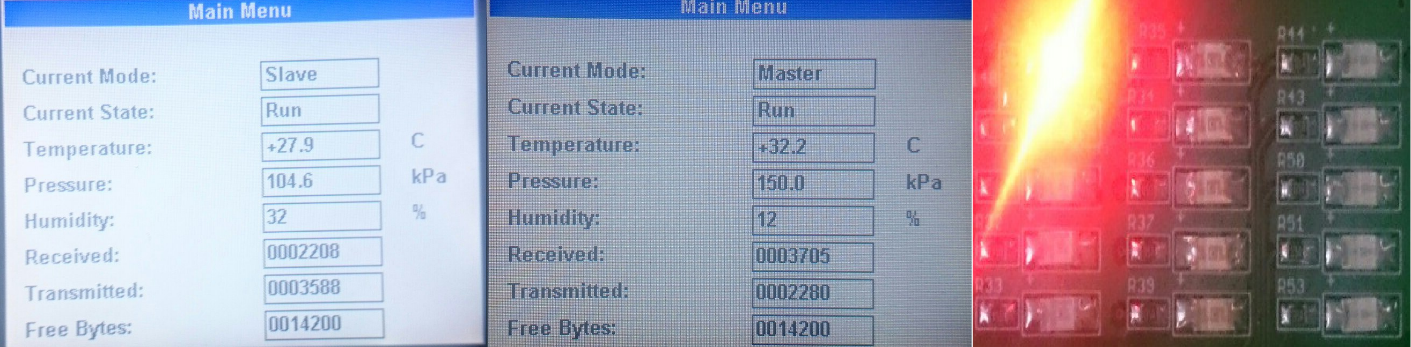


Рисунок 3.9 - состояние устройств во время атаки «ложный ведомый»

На рисунке 3.9 видно, что данные датчиков, которые получило ведущее устройство, не совпадают с данными на ведомом устройстве. Так как ведущее устройство получило ложные данные о температуре (истинная температура 27.9 °C, полученная от устройства-злоумышленника – 32.2 °C), светодиод, имитирующий нагреватель, включился. На реальном производстве такая ситуация вполне могла бы привести к выходу механизмов из строя.,

Последней идёт атака «ложный ведущий». Для этой атаки необходимо уменьшить время обработки ведомого до 70 мс, чтобы устройство-злоумышленник успело сделать запрос до запроса настоящего ведущего. Блок-схема атаки представлена на рисунке 3.10.

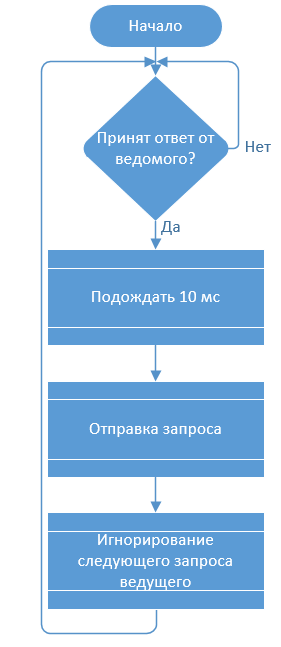


Рисунок 3.10 - блок-схема атаки «ложный ведущий»

Так как данная атака не наглядна с точки зрения данных на дисплее, необходимо подключить устройство-злоумышленник к ПК. В специальной программе-терминале можно просмотреть ситуацию на линии и убедиться, что атака сработала. На рисунке 3.11 показана ситуация, происходящая в линии при атаке.

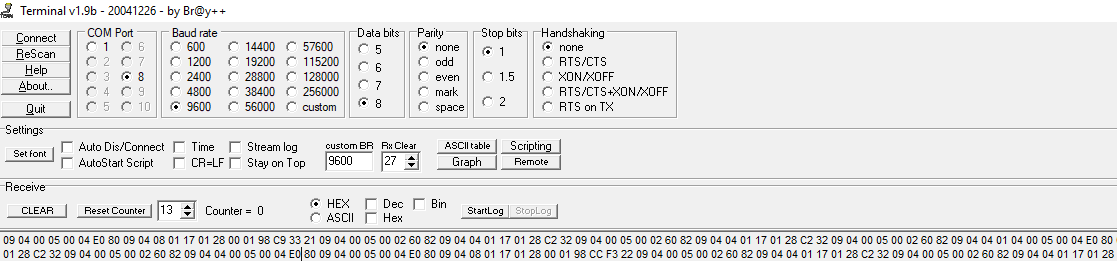


Рисунок 3.11 – ситуация на линии при атаке «ложный ведущий»

Для удобства разобьём данные на пакеты. Получается следующее:

1. 09 04 00 05 00 04 E0 80
2. 09 04 08 01 17 01 28 00 01 98 C9 33 21
3. 09 04 00 05 00 02 60 82
4. 09 04 04 01 17 01 28 C2 32

1-ый пакет – это запрос ведущего на чтение 4-х регистров ввода, начиная с адреса 0x05. В регистре 0x05 хранится температура, в 0x06 – влажность, в 0x07-0x08 – атмосферное давление.

2-ой пакет – это ответ ведомого на 1-ый пакет.

3-ий пакет – это запрос устройства-злоумышленника на чтение 2-х регистров ввода, начиная с адреса 0x05.

4-ый пакет – это ответ ведомого на 3-ий пакет, то есть на запрос устройства-злоумышленника.

Как можно видеть, данная атака действительно сработала. Главная опасность этой атаки состоит в том, что если ведомое устройство позволяет настраивать параметры обмена через запись в регистры хранения, то устройство-злоумышленник, зная точно тип устройства, а, соответственно, в какие регистры нужно писать, может перезаписать параметры обмена на свои и тогда легитимный ведущий при следующем сеансе обмена не сможет осуществить опрос ведомого.

В итоге, можно сделать вывод, что все три атаки легко осуществимы. Из полученных результатов следуют основные слабые места протокола Modbus:

1. Общая шина для всех устройств. В случае спам-атаки, связь обрывается сразу со всеми подключенными в линию устройствами. К сожалению, данное слабое место практически неустранимо, так как проводить отдельную линию связи для каждого устройства крайне затратно. К тому же, в данном случае будет необходимо несколько ПЛК, так как у одного просто не хватит интерфейсных выходов.
2. Отсутствие шифрования. Все данные передаются в открытом виде, любое устройство, подключенное к линии, может видеть весь обмен.
3. Отсутствие аутентификации. Любое устройство, подключенное в линию, может инициировать обмен или отвечать на запросы.
   1. **Методы защиты от уязвимостей**

Причины, по которым протоколы передачи данных между нижним и верхним уровнем передают данные без аутентификации и в открытом виде довольно понятны. Промышленная автоматика – довольно консервативная область, в то время, когда разрабатывались текущие лидеры рынка – протоколы Modbus и Profibus (принят в Германии в 1987 году) реализация аутентификации и криптоалгоритмов на микроконтроллерах была крайне затруднительной. Например, чрезвычайно популярное в 1980-х годах семейство микроконтроллеров MCS-51 (Intel 8051) стандартно располагало 4 кБ памяти программ и 128 байтами памяти данных. В такой конфигурации контроллер даже не мог полностью реализовывать Modbus, так как памяти данных не хватало на хранение пакета максимальной длины. Поэтому во многих микроконтроллерах этого семейства на кристалле располагали дополнительный блок памяти данных, объёмом обычно 2-4 кБ. Либо подключалась внешняя оперативная память. Но даже если памяти было достаточно, вычислительной мощности для криптографии и\или аутентификации всё равно не хватало. Если взглянуть на синтетические тесты, ядро 8051 имеет производительность всего лишь порядка 0.0187 Dhrystone MIPS на 1 МГц тактовой частоты. [6] Современный микроконтроллер семейства STM32 STM32L021D4, стоимостью всего 0.692$ при партии от 1000 штук [7] имеет производительность 0.95 Dhrystone MIPS на 1 МГц тактовой частоты, что в почти 51 раз выше. [8]. К тому же, стандартное ядро 8051 имеет максимальную тактовую частоту 12МГц, в то время как ядро современных микроконтроллеров работает на гораздо более высокой частоте (ядро вышеупомянутого STM32L021D4 может работать на частотах до 32 МГц).

По совокупности вышеуказанных причин, защита каналов связи в основном ограничивается периметральной системой охраны, защитой автоматизированного рабочего места оператора и межсетевыми экранами, в случае если каналом связи выступает сеть Ethernet. Но, как показала ситуация с вирусом Stuxnet, атака может начаться и изнутри. Только вместо заражённого USB-носителя в промышленную сеть может включаться устройство-злоумышленник. Поэтому для новых разрабатываемых устройств имеет смысл разрабатывать и применять протоколы с аутентификацией и шифрованием, так как многие микроконтроллеры уже оснащаются аппаратной поддержкой криптографии (вышеупомянутый STM32L021D4 имеет аппаратную поддержку 128-битного AES).

Например, можно незначительно переработать протокол Modbus, добавив в него аутентификацию, например, на основе дайджеста. В этом случае к пакету Modbus добавляется случайное значение nonce, призванное затруднить криптоанализ, и собственно дайджест, который получается следующим образом:

digest = MD5(data:nonce:password), где data – собственно пакет Modbus без контрольной суммы

Для того, чтобы общий пакет не стал слишком большим, размер nonce ограничивается 16 байтами. Таким образом, максимальная длина пакета возрастает до 288 байт: 256 байт пакет Modbus + 16 байт nonce + 16 байт digest.

Итоговый вид пакета данных представлен на рисунке 3.12.



Рисунок 3.12 – структура пакета протокола с аутентификацией

Контрольная сумма считается точно так же, как и в Modbus, nonce и digest при подсчёте контрольной суммы тоже учитываются.

Соответственно, обмен будет происходить следующим образом:

1. Ведущее устройство генерирует nonce, на основе него, данных и пароля считает digest. На основе пакета данных (адрес, код функции и данные) вычисляется контрольная сумма. Значение nonce на данном этапе необходимо запомнить до получения ответа или истечения таймаута ответа.
2. Ведомое устройство при приёме сначала проверяет контрольную сумму. Затем, зная nonce, пароль и данные вычисляет digest и проверяет, совпадает ли digest в пакете с вычисленным. В случае несовпадения запрос не обрабатывается и ответ не генерируется.
3. Если digest совпал, ведомое устройство генерирует ответ точно таким же образом, как ведущее. Значение nonce ведомое устройство должно сгенерировать заново.
4. Ведущее устройство после проверки контрольной суммы должно проверить, было ли значение nonce сгенерировано заново. Если оно осталось тем же – ответ не обрабатывается.

При этом пароль необходимо записать в оба устройства на этапе настройки, когда сеть считается защищённой. Для этого можно воспользоваться тем, что Modbus позволяет расширять протокол за счёт пользовательских функций.

Для установки пароля добавим функцию с кодом 0x41. Длина пароля из соображений разумности ограничена 16 байтами. Структура запроса приведена в таблице 3.13, ответ ведомого является повтором запроса.

Таблица 3.13 – структура запроса для установки пароля

|  |  |  |
| --- | --- | --- |
| Код функции | 1 байт | 0x41 |
| Количество байт далее | 1 байт | N, от 1 до 16 |
| Пароль | N байт |  |

Так как в начальном состоянии (из коробки) ведомое устройство не имеет пароля, то для его установки необходимо однократно сформировать запрос с кодом 0x41 как в стандартном Modbus, без nonce и digest, ведь при первоначальной настройке сеть считается безопасной. Если же пароль уже установлен ранее, то запрос на установку пароля обязан содержать nonce и digest.

Для установки пароля на стороне ведущего устройства необходимо предусмотреть отдельный сервисный путь, который, для безопасности, не должен задаваться средствами данного протокола. В идеальном случае, для сервисных нужд должен быть выделен отдельный интерфейс.

# **ЗАКЛЮЧЕНИЕ**

В ходе настоящей работы были проанализированы уязвимости промышленного канала связи на основе протокола Modbus, разработаны способы атаки. Для моделирования данных атак и наглядной демонстрации последствий атак был разработан лабораторный стенд.

В ходе моделирования были получены следующие результаты:

1. При проведении атаки «спам случайными данными» весь обмен в линии полностью прекращается.
2. Атака «ложный ведомый» с точки зрения работоспособности системы является самой опасной, так как происходит подмена показаний датчиков и ведущее устройство перестаёт получать корректную обратную связь от ведомого.
3. Атака «ложный ведущий» является опасной с точки зрения того, что позволяет в некоторых случаях (когда ведомое устройство позволяет настраивать параметры обмена через регистры хранения Modbus) оборвать обмен ведущего с конкретным ведомым.

На основе данных результатов был предложен новый протокол на основе Modbus, который позволяет проводить аутентификацию устройств. Данный протокол защищён от атак «ложный ведомый» и «ложный ведущий» до тех пор, пока пароль содержится в секрете и неизвестен атакующей стороне. К сожалению, способ защиты от атаки «спам случайными данными» существует только единственный – подключение каждого ведомого устройства к ведущему по своей индивидуальной линии. Данное решение крайне затратно по денежным средствам и обслуживанию, так как требуется установка большого количества ПЛК и протягивание огромного количества витых пар. Соответственно, данное решение может применимо только тогда, когда ведомых устройств не больше, чем интерфейсов RS-485 у ПЛК.

**СПИСОК ИСПОЛЬЗОВАННОЙ ЛИТЕРАТУРЫ**

1. Структура распределённой АСУ ТП: [Электронный ресурс]. URL: (дата обращения: 05.05.2018)
2. Безопасность промышленных систем: [Электронный ресурс]. URL: https://www.ptsecurity.com/upload/corporate/ru-ru/analytics/SCADA-analytics-rus.pdf (дата обращения: 05.05.2018)
3. Stuxnet и промышленная безопасность: [Электронный ресурс]. URL: http://www.phocus-scada.com/rus/pub/Stuxnet&IndustrialSecurity.html (дата обращения: 05.05.2018)
4. Число атак на АСУ ТП в 2016 году выросло на 110%: [Электронный ресурс]. URL: https://www.securitylab.ru/news/484930.php (дата обращения: 05.05.2018)
5. Modbus Application Protocol Specification v1.1b3: [Электронный ресурс]. URL: http://www.modbus.org/docs/Modbus\_Application\_Protocol\_V1\_1b3.pdf (дата обращения: 06.05.2018)
6. Lightweight 8051 compatible CPU :: Overview: [Электронный ресурс]. URL: <https://opencores.org/project/light52> (дата обращения: 07.05.2018)
7. STM32L021D4P7 - ARM Microcontroller, ARM Cortex-M0+ Microcontrollers, ARM Cortex-M0+, 32bit, 32 MHz, 16 KB, 2 KB: [Электронный ресурс]. URL: http://uk.farnell.com/stmicroelectronics/stm32l021d4p7/mcu-arm-cortex-m0-32mhz-tssop/dp/2851007 (дата обращения: 07.05.2018)
8. STM32L021 Datasheet: [Электронный ресурс]. URL: <http://www.st.com/resource/en/datasheet/stm32l021g4.pdf> (дата обращения: 07.05.2018)