|  |  |
| --- | --- |
|  | **ИНСТИТУТ ИНТЕЛЛЕКТУАЛЬНЫХ КИБЕРНЕТИЧЕСКИХ СИСТЕМ**  **Кафедра**  **«Криптология и кибербезопасность»** |

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**ОТЧЕТ**

**о БДЗ по дисциплине**

**«Информационная безопасность автоматизированных систем управления технологическим процессом»**

« Добавление модуля анализа промышленного протокола IEC104

в ПО Suricata»

|  |  |  |
| --- | --- | --- |
| Исполнители:  студенты гр. Б17-505: |  | Зоричев В.В. |
|  |  | Рудик М.В. |
|  |  | Савченко А.М. |
|  |  | Худоярова А.М. |
|  |  |  |
| Преподаватель: |  | Финошин М.А. |

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Москва – 2020**

СОДЕРЖАНИЕ

[1 Описание промышленного протокола IEC 104 3](#_Toc59357221)

[1.1 APDU 3](#_Toc59357222)

[1.2 APCI 4](#_Toc59357223)

[1.3 ASDU 4](#_Toc59357224)

[1.4 Таймеры 5](#_Toc59357225)

[2 Клиент-сервер для промышленного протокола IEC – 104 7](#_Toc59357226)

[2.1 Запуск 7](#_Toc59357227)

[2.2 Формирование трафика 7](#_Toc59357228)

[2.3 Содержание дампа 7](#_Toc59357229)

[3 Настройка Suricata 10](#_Toc59357230)

[4 Правила Suricata 11](#_Toc59357231)

[Заключение 16](#_Toc59357232)

# 1 Описание промышленного протокола IEC 104

Стандарт IEC 60870–5-104 (IEC 104): «Доступ к сети для IEC 60870-5-101 с использованием стандартных транспортных профилей» опубликован Международной электротехнической комиссией (IEC) в наборе стандартов IEC 60870-5.

IEC 60870–5 обеспечивает профиль связи для отправки базовых сообщений телеуправления (диспетчерское управление и сбор данных) между двумя системами через постоянно подключенные каналы передачи данных между этими системами. Стандарт IEC 104 определен для передачи сообщений телеуправления IEC 60870–5-101 (IEC 101) через TCP с использованием порта 2404. То есть IEC 104 инкапсулирует сообщения телеуправления IEC 101 в блок данных APDU (Application Protocol Data Unit), который передается как часть полезной нагрузки TCP (рис.1).

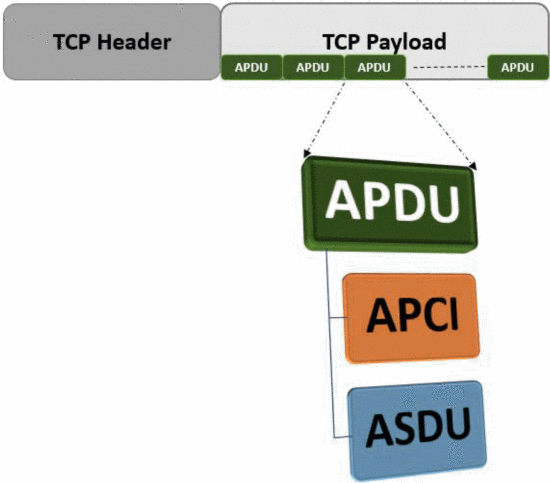


Рисунок 1 – APDU как полезная нагрузка TCP.

## 1.1 APDU

Поля APDU состоят из двух частей (рис. 2):

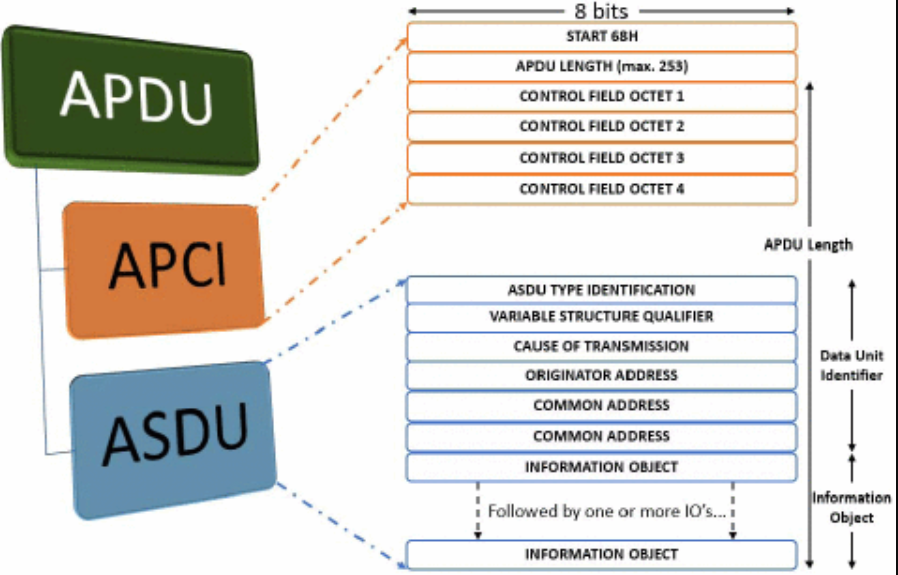


Рисунок 2 – Структура ADPU

1. APCI (Application Protocol Control Information) ⸺ информация управления, которая функционирует как заголовок;
2. ASDU (Application Protocol Data Unit) - блок данных, который содержит данные прикладного уровня IEC 101 то есть сообщения телеуправления.

## 1.2 APCI

APCI - это 6-октетный заголовок, который включает в себя начальный октет (68H), за которым следуют 1-октетное поле, определяющее длину APDU (4 октета полей управления и длина ASDU) и 4-октетные поля управления. Два младших бита первого октета поля управления определяют тип сообщения телеуправления, которое может иметь формат:

1. I (00b) – передает информацию датчиков и управления, и за APCI всегда следует ASDU;
2. S-формат (01b) – используется для отправки подтверждений порядкового номера;
3. U-формат (11b) – используется для отправки сообщений поддержания активности соединения.

## 1.3 ASDU

ASDU – это структура данных переменной длины, которая состоит из идентификатора блока данных фиксированной длины DUI (Data Unit Identifier) и информационного объекта переменной длины IO (Information Object). ASDU состоит из следующих полей (рис. 3):

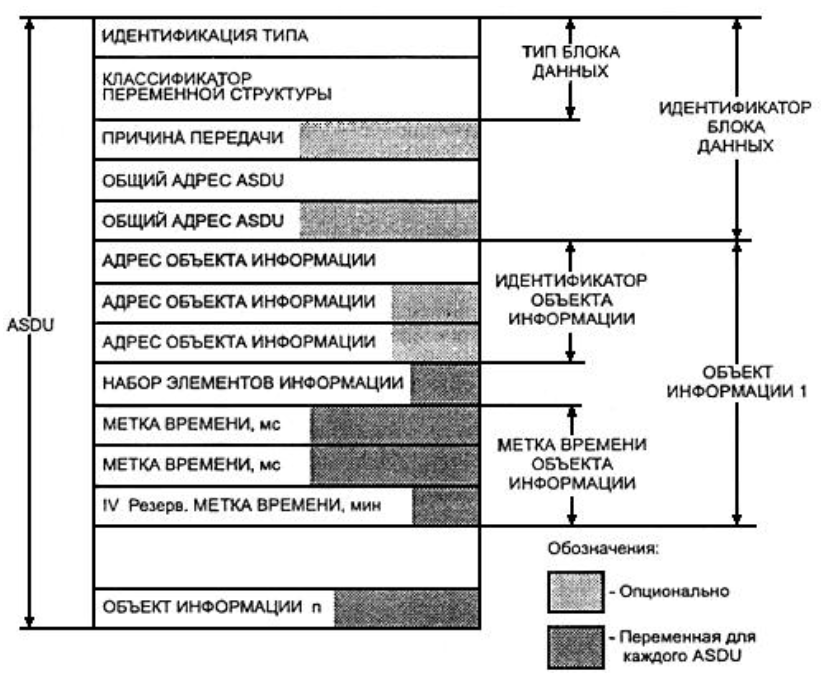


Рисунок 3 – Структура ASDU

1. Идентификатор типа ASDU – определяет структуру, тип и формат следующего(их) за ним объекта(ов) информации (IO). IO с временной меткой или без нее отличаются различными значениями идентификатора типа. При приеме ASDU со значениями идентификатора типа, не входящими в заранее определенный перечень, посылается отрицательная квитанция;
2. Классификатор переменной структуры;
3. Причина передачи (рис. 4) – направляет ASDU определенной прикладной задаче для обработки. Бит P/N показывает, какое (положительное или отрицательное) подтверждение активации требуется для первичной прикладной функции. В случае, когда бит P/N не используется, он равен нулю;

Рисунок 4 – Поле причины передачи ASDU

1. Общий адрес ASDU – связан со всеми объектами в данном ASDU. Глобальный адрес (65535) – это широковещательный адрес, обращенный ко всем станциям данной системы. В ответ на переданные в направлении управления ASDU с широковещательным адресом, в направлении контроля должны передаваться ASDU, содержащие специально определенный общий адрес;
2. Адрес объектов информации – используется как адрес получателя в направлении управления и как адрес источника в направлении контроля;
3. Элементы информации;
4. Метка времени.

## 1.4 Таймеры

IEC 104 также определяет следующие таймеры:

1. T0 – тайм-аут этого времени вызовет запрос SYN;
2. T1 – тайм-аут этого таймера вызовет активный запрос закрытия для установленного соединения. Кроме того, событие тайм-аута T1 на управляющей станции может автоматически инициировать новое резервное соединение и автоматическое переключение соединения;
3. T2 – истечение этого таймера заставит приемник отправить S-формат с принятым порядковым номером (сообщение ACK), который является подтверждением на уровне приложения для получения определенного количества APDU;
4. T3 – истечение этого таймера указывает, что не было получено сообщений IEC 104 с другой стороны для данного открытого TCP-соединения. Прием любого типа сообщения, I-формата, S-формата или U-формата сбрасывает этот таймер.

# 2 Клиент-сервер для промышленного протокола IEC – 104

За основу был взят следующий симулятор IEC-104 трафика: <https://github.com/RocyLuo/IEC104TCP>. Однако он не является полностью рабочим, поскольку написан для Python2, а также содержит ошибку в коде.

Поэтому данная реализация была модифицирована:

1. Интерпретирована под Python3;
2. Устранена ошибка в библиотеке scapy ⸺ файл fields.py на локальном компьютере необходимо заменить 167-ю строку на строку:

return copy.copy(x), ошибка именно в стандартной библиотеке;

1. Выполнено корректное завершение соединения между клиентом и сервером;
2. Исправлены названия файлов, чтобы они соответствовали своей реальной функциональности, то есть сервер в изначальном симуляторе на самом деле должен быть клиентом, а клиент сервером.

## 2.1 Запуск

Необходимо установить библиотеку scapy следующей командой:

python3 -m pip install scapy

Для запуска примера выполнить следующие команды:

python3 EchoIEC104Client.py

python3 example.py

Данные для теста описаны в файле iec104\_tcp\_packets.py и представляют собой APDU (APCI и ASDU).

## 2.2 Формирование трафика

После запуска example.py формируется передача данных по IEC-104.

## 2.3 Содержание дампа

При запуске формируются блоки данных на прикладном уровне (ASDU) с идентификаторами типа 45, 46, 47, 48, 58, 59, 60, 61, 62, 50, 101, 103 и управляющая информация прикладного уровня (APCI с форматами поля управления "s", "i", "u"):

1. 45(C\_SC\_NA\_1) - однопозиционная команда;
2. 46(C\_DC\_NA\_1) - двухпозиционная команда;
3. 47(C\_RC\_NA\_1) - команда пошагового регулирования;
4. 48(C\_SE\_NA\_1) - команда уставки, нормализованное значение;
5. 50(C\_SE\_NC\_1) - команда уставки, короткий формат плавающей запятой;
6. 58(C\_SC\_TA\_1) - однопозиционная команда с меткой времени СР56Время2а;
7. 59(C\_DC\_TA\_1) - двухпозиционная команда с меткой времени СР56Время2а;
8. 60(C\_RC\_TA\_1) - команда пошагового регулирования с меткой времени СР56Время2а;
9. 61(C\_SE\_TA\_1) - команда уставки, нормализованное значение с меткой времени СР56Время2а;
10. 62(C\_SE\_TB\_1) - команда уставки, масштабированное значение с меткой времени СР56Время2а;
11. 101(C\_CI\_NA\_1) - команда опроса счетчиков;
12. 103(C\_CS\_NA\_1) - команда синхронизации часов.

Результирующий дамп находится в файле result.pcanpg. На рисунке 1 представлен общий сформированный дамп трафика при передаче данных по IEC-104.

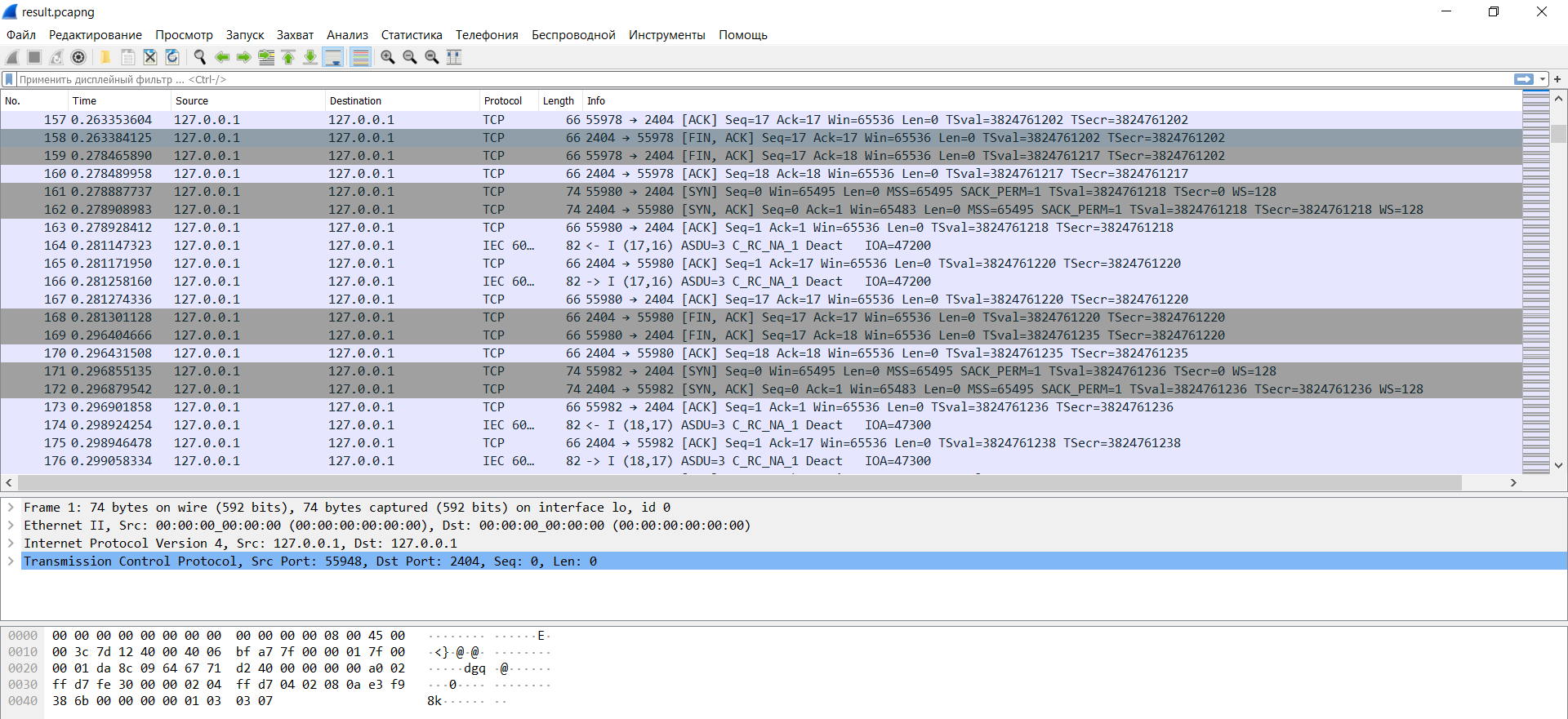


Рисунок 1 — общий сформированный дамп трафика при передаче данных по IEC-104.

На рисунке 2 представлен пакет, с типом ASDU №45, то есть это однопозиционная обычная команда, что также видно на скриншоте в расшифровке команды.

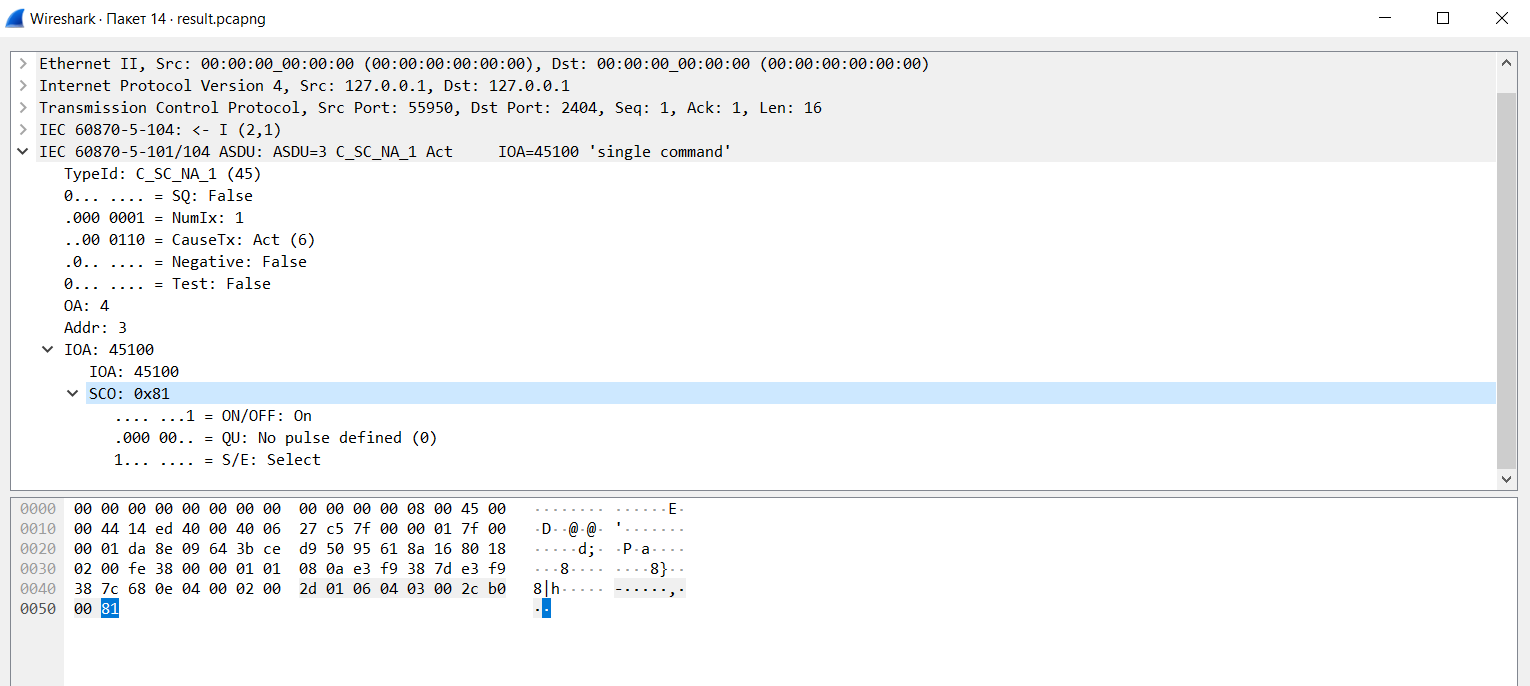
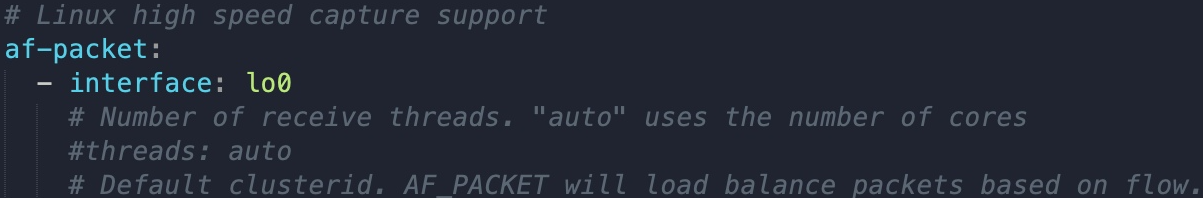
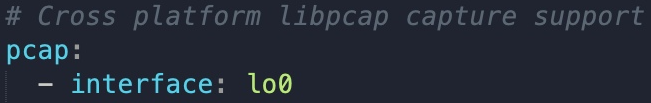


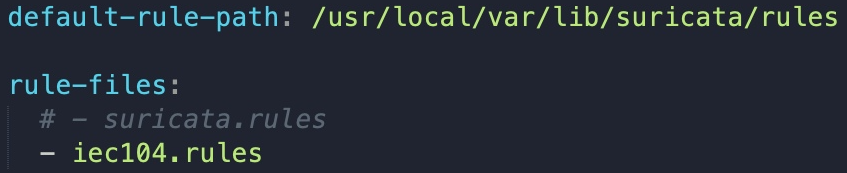
Рисунок 2 — блок данных ASDU с типом №45.

# 3 Настройка Suricata

Исправления в конфигурационном файле suricata.yaml:







Запуск Suricata:

suricata -c /usr/local/etc/suricata/suricata.yaml -i lo0 —set capture.disable-offloading=false

Просмотр логов для проверки работы правил:

tail -f /usr/local/var/log/suricata/fast.log

Пакеты для тестирования:

* ('START','auto','if',[45,1,Cause\_Act,4,3,(45000,SCO\_Off\_Np\_Se)]),
* ('START','auto','if',[58,1,Cause\_Act,4,3,(45200,SCO\_Off\_Np\_Se)]),
* ('START','auto','if',[61,1,Cause\_Deact,4,3,(48250,10,0x80)]),
* ('START','auto','if',[61,1,Cause\_Deact,4,3,(48300,101,0x80)]),
* ('START','auto','if',[101,1,Cause\_Deact,4,4,(0,0x05)]),
* ('START','auto','if',[103,1,Cause\_Deact,4,4,(0,0x05)]),
* ('START','auto','sf'),
* ('START','auto','uf',STARTDT\_ACT),
* ('START','auto','uf',TESTFR\_ACT),
* ('START','auto','uf',TESTFR\_CON).

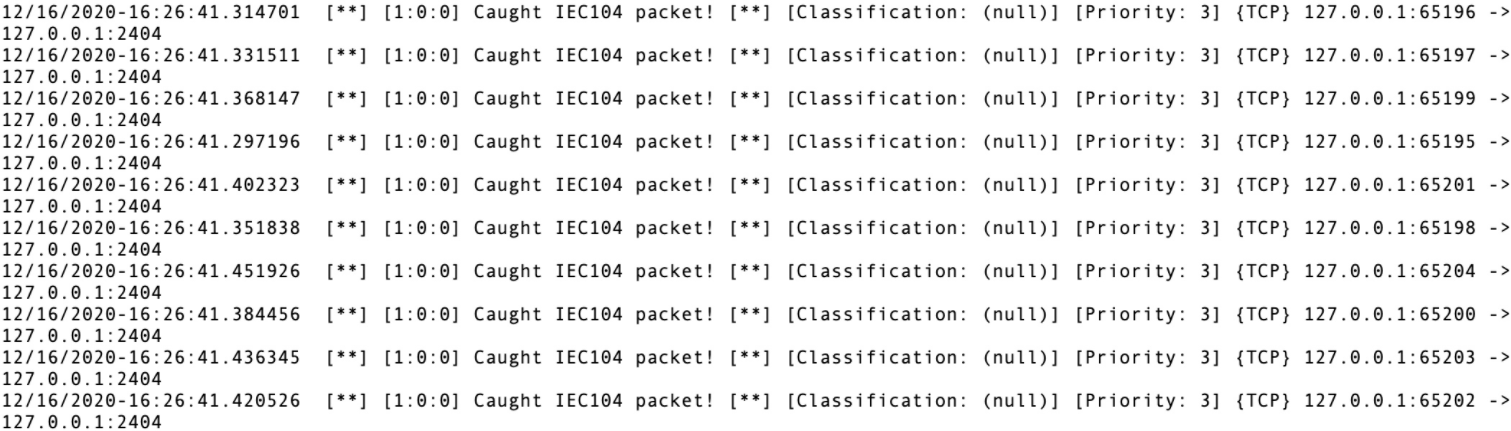
# 4 Правила Suricata

Для детектирования различных пакетов протокола IEC104 с помощью ПО Suricata написаны следующие правила:

1. Детектирует пакеты IEC104:

alert tcp 127.0.0.1 any -> 127.0.0.1 2404 (msg: "Caught IEC104 packet!"; content: "|68|"; startswith;)

Результат работы:

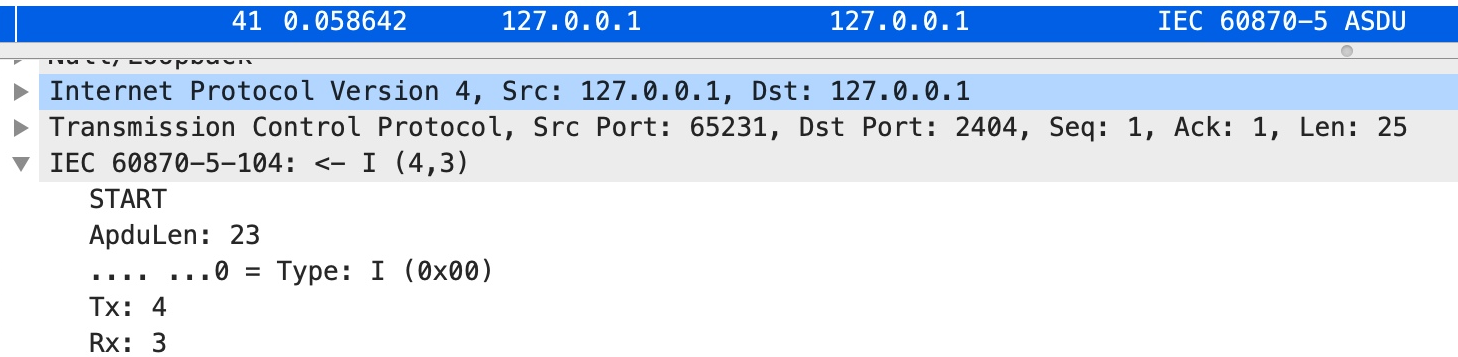


1. Детектирует пакеты с счетчиком переданных блоков Tx, равным 4:

alert tcp 127.0.0.1 any -> 127.0.0.1 2404 (msg: "Tx=4 packet"; content: "|08 00|"; offset: 2; depth: 2; )

Результат работы:





1. Детектирует пакеты с счетчиком принятых блоков Rx, равным 3:

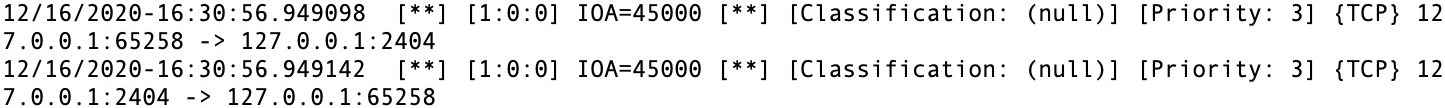
alert tcp 127.0.0.1 2404 -> 127.0.0.1 any (msg: "Rx=3 packet"; content: "|68|"; content: "|06 00|"; distance: 3; within: 5; rawbytes; )

Результат работы:



1. Детектирует пакеты с адресом объекта информации, равным 45000:

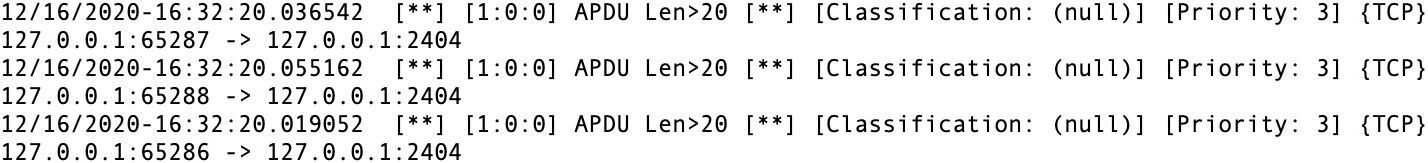
alert tcp 127.0.0.1 any <> 127.0.0.1 2404 (msg: "IOA=45000"; content: "|c8 af 00|"; offset: 12; depth: 15; rawbytes; ) Результат работы:



1. Детектирует пакеты с длиной APDU больше 20 байт:

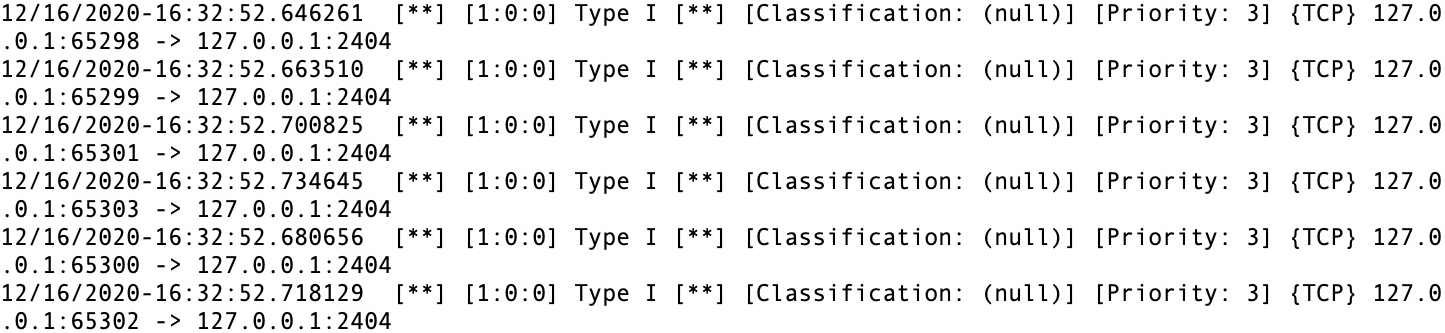
alert tcp 127.0.0.1 any -> 127.0.0.1 2404 (msg: "APDU Len=20"; content: "|68|"; byte\_test:1,>,20,0,relative; )

Результат работы:



1. Детектирует пакеты с типом I сообщения телеуправления:

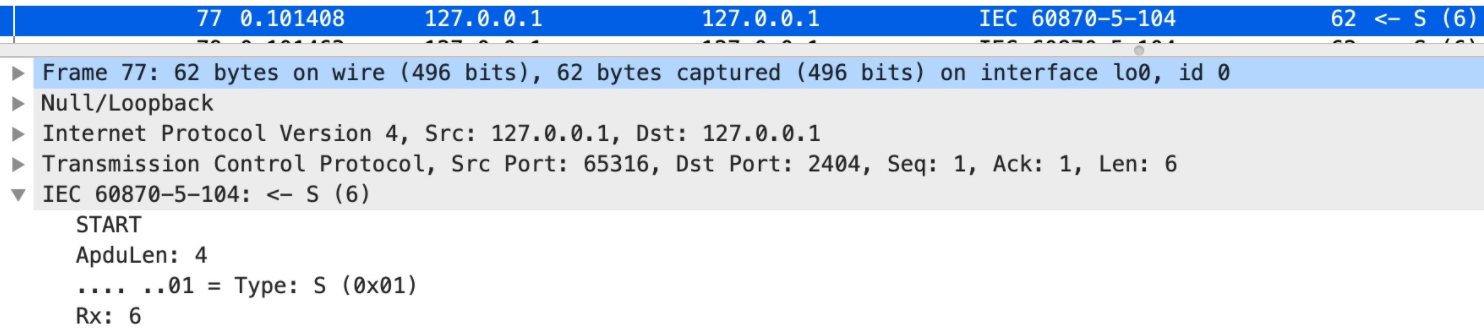
alert tcp 127.0.0.1 any -> 127.0.0.1 2404 (msg: "Type I"; content: "|68|"; byte\_test:1,=,0,1,relative, bitmask 0x01; ) Результат работы:



1. Детектирует пакеты с типом S сообщения телеуправления:

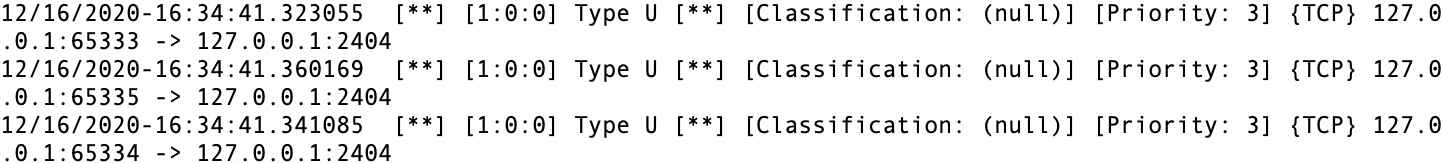
alert tcp 127.0.0.1 any -> 127.0.0.1 2404 (msg: "Type S"; content: "|68|"; byte\_test:1,=,1,1,relative, bitmask 0x03; ) Результат работы:





1. Детектирует пакеты с типом U сообщения телеуправления:

alert tcp 127.0.0.1 any -> 127.0.0.1 2404 (msg: "Type U"; content: "|68|"; byte\_test:1,=,3,1,relative, bitmask 3; ) Результат работы:

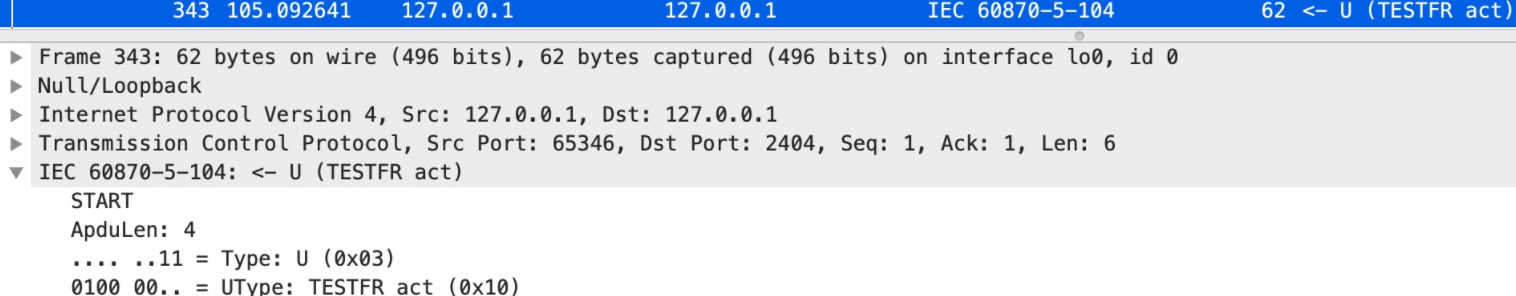


1. Детектирует пакеты с тестовым APDU (TESTFR = act):

alert tcp 127.0.0.1 any -> 127.0.0.1 2404 (msg: "Type U TESTFR ACT"; content: "|68|"; byte\_test:1,=,67,1,relative, bitmask 255; )

Результат работы:

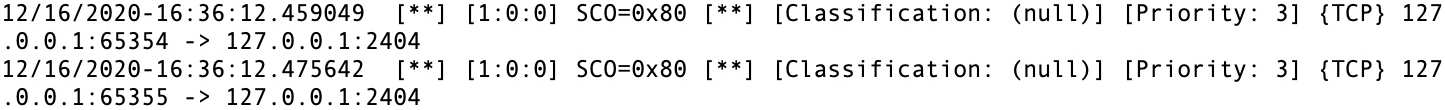




10) Детектирует однопозиционные команды включения:

alert tcp 127.0.0.1 any -> 127.0.0.1 2404 (msg: "SCO=0x80"; content: "|68|"; byte\_test:1,=,0x80,15; )

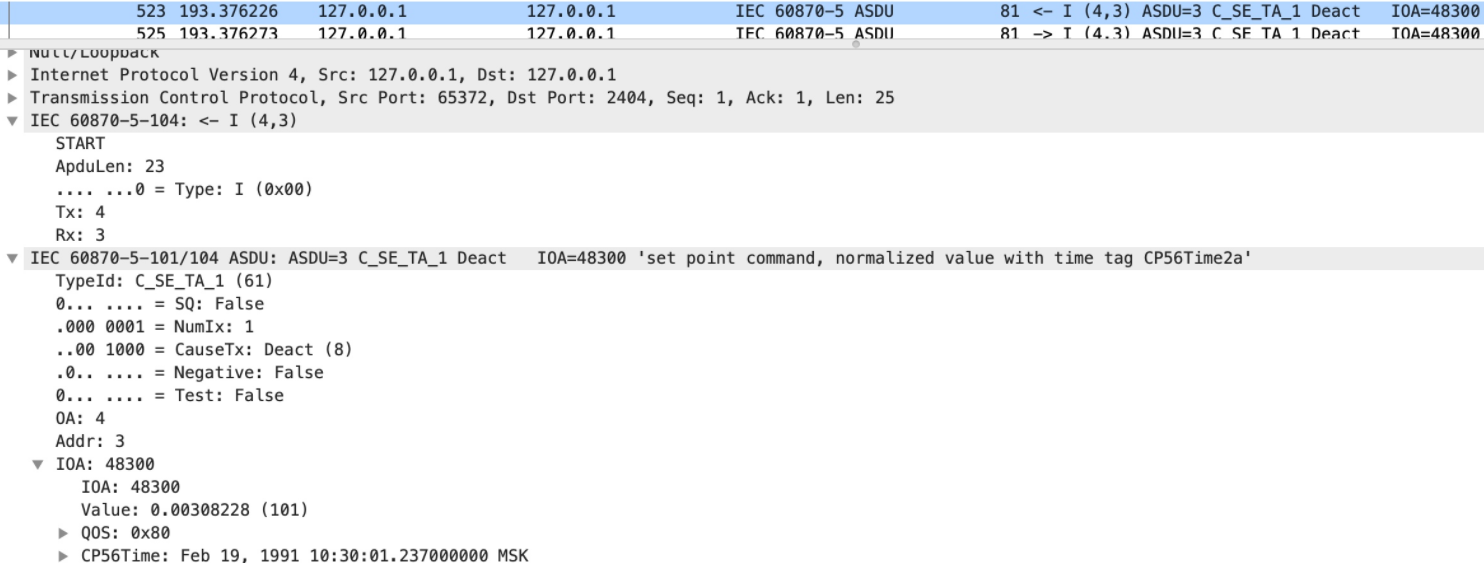
Результат работы:



11) Детектирует пакеты со значением информационного объекта (IO), равным 101:

alert tcp 127.0.0.1 any -> 127.0.0.1 2404 (msg: "Value=101"; content: "|68|"; content: "|65|"; distance: 14; within: 15; ) Результат работы:



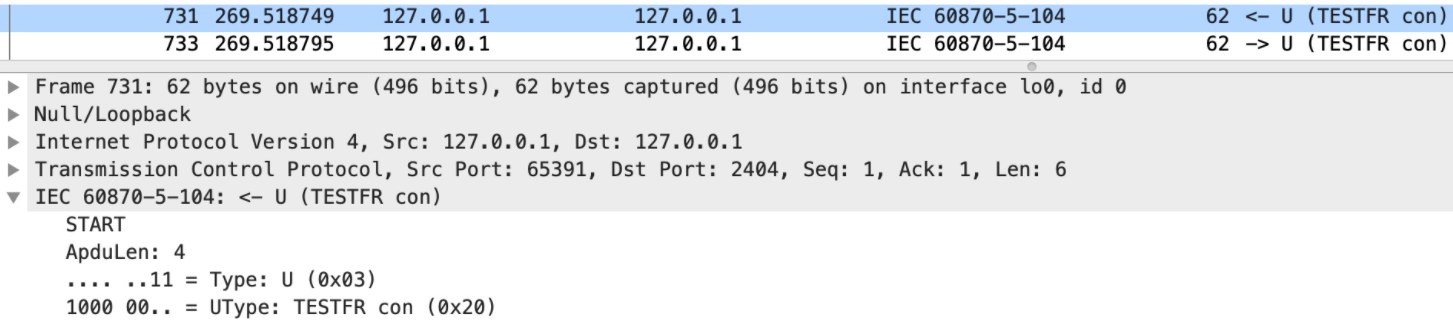


12) Детектирует пакеты с подтверждающим соединение APDU (TESTFR = con:

alert tcp 127.0.0.1 any -> 127.0.0.1 2404 (msg: "Type U TESTFR CON"; content: "|68|"; byte\_test:1,=,131,1,relative, bitmask 255; )

Результат работы:

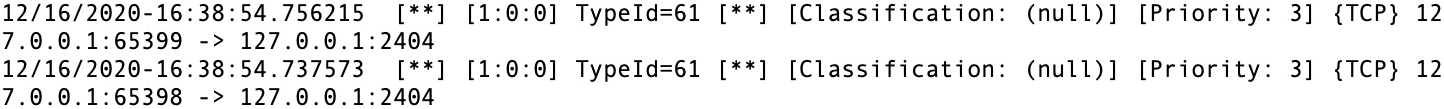




13) Детектирует пакеты c ID типа передаваемой информации, равным 61:

alert tcp 127.0.0.1 any -> 127.0.0.1 2404 (msg: "TypeId=61"; content: "|68|"; content: "|3d|"; distance: 5; within: 6;)

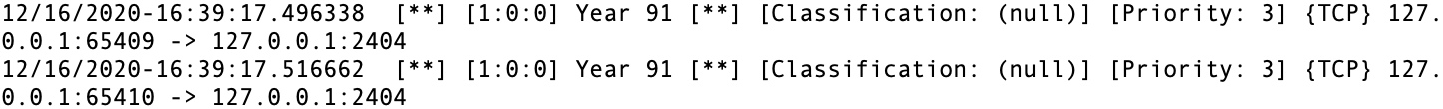
Результат работы:



14) Детектирует передача пакетов, отправленных со значением поля года равным 91:

alert tcp 127.0.0.1 any -> 127.0.0.1 2404 (msg: "Year 91"; content: "|5b|"; offset: 0x17; depth: 0x18; )

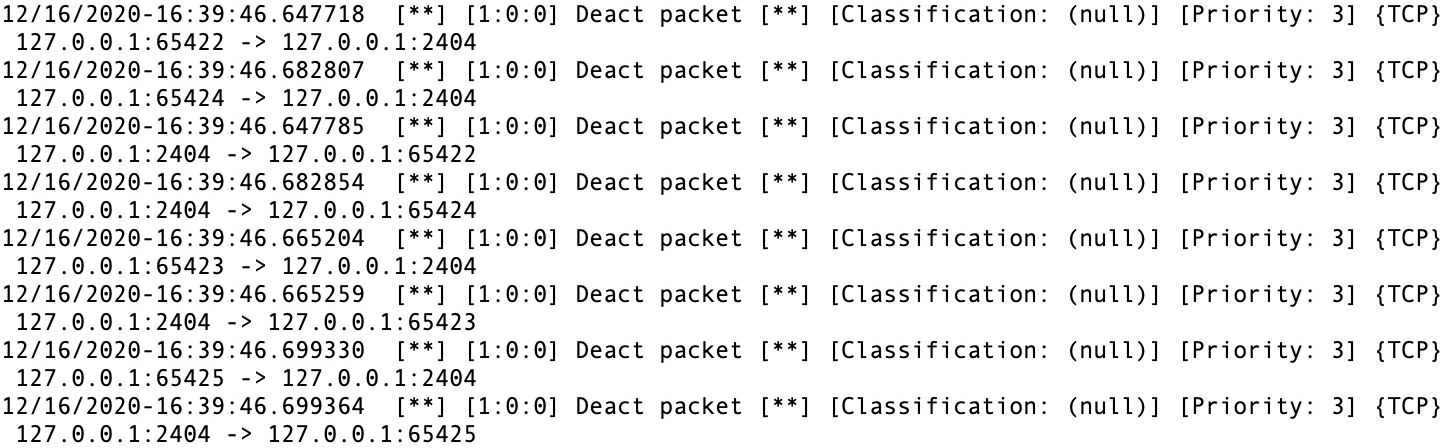
Результат работы:



15) Детектирует деактивацию постоянной циклической или периодической передачи адресуемого объекта:

alert tcp 127.0.0.1 any <> 127.0.0.1 2404 (msg: "Deact packet"; content: "|68|"; byte\_test:1,=,0x08,7,relative; )

Результат работы:



Заключение

По итогам выполнения большого домашнего задания были реализованы клиент серверное приложение для генерирования трафика протокола IEC-104 и было описано 15 правил для разбора и детектирования определенных полей протокола IEC-104.