**Отчет по тестовому заданию для стажера на позицию “Инженер” в ЦНИПР**

**Выполнил: Жуков Матвей Витальевич**

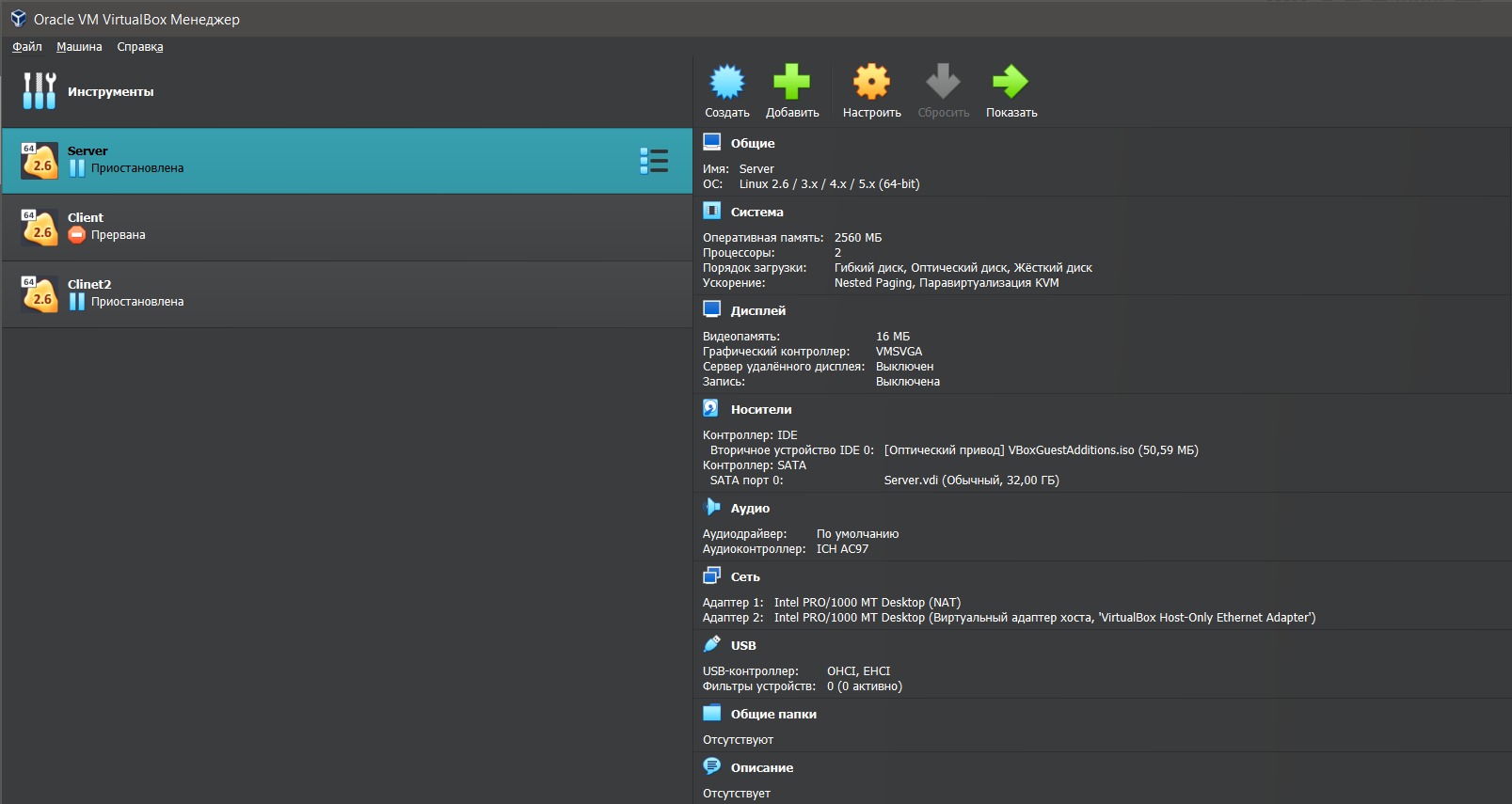
## Задание №1

1. **Создать две виртуальные машины (ВМ) под управлением любого Linux. Гипервизор – любой, работающий под Windows. Связать ВМ при помощи виртуальной сети. Проверить, что ping успешно проходит.**

Для того чтобы создать две виртуальные машины под управлением любого Linux, я взял в качестве гипервизора: VirtualBox. Он один из самых быстрых и удобных гипервизоров под Windows.

В качестве версии Linux мною был выбран дистрибьютор **Ubuntu** версии **22.04,** из всех имеющихся дистрибьюторов, он самый удобный в использовании.

Создал две Виртуальные Машины (ВМ) и назвал **Server** и **Client2** соответственно:

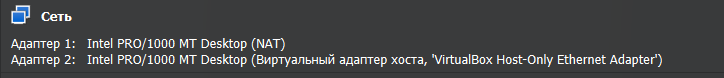
Скриншот (1)

Для ВМ Server было выделено 32 Гб место на жестком диске, а для Client2 было выделено 42 Гб.

В настройках для каждой ВМ в разделе **Сеть** подключил 2 адаптера:

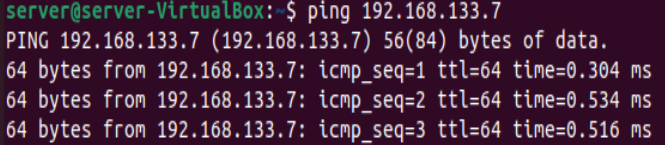
**NAT** – Для того чтобы ВМ имели доступ в интернет.

**Виртуальный адаптер хоста** – VirtualBox Host-Only Ethernet Adapter, для соединения Клиента и Сервера между собой.



Скриншот (2)

С помощью команды **ping 192.168.133.7** проверил, что ping проходит и ВМ связаны.



Скриншот (3)

1. **На одной ВМ (Клиент) разработать с использованием C/C++ ПО Агент, который будет запускаться автоматически вместе с Клиентом, регулярно собирать какие-либо данные о функционировании Клиента и отправлять их на другую ВМ (Сервер\_ML). Какие собирать данные остается на усмотрение исполнителя, но желательно, чтобы они были важны с точки зрения информационной безопасности. Идеально, если по этим данным можно обнаружить атаку на/компрометацию Клиента.**

Код Клиента на С++ будет выложен на Git, для того чтобы ПО Агент запускался автоматически вместе с Клиентом, нужно написать небольшой скрипт на Python и прописать его запуск в автозагрузку системы с помощью **stystemd.**

**Для этого потребуется:**

1)Написать скрипт на Python (Код скрипта будет на Git(script.py))

2)Сделать скрипт исполняемым с помощью команды в Терминале: sudo chmod +x script.py

3)Дать права пользователя на исполнение этого скрипта, для этого прописываем в терминале: sudo vim /etc/systemd/system/script.service

И прописываем в нашем файле script.service следующее:

**[Unit]**

**Description = Скрипт для запуска ПО Агент (название, которое будет отображаться)**

**[Service]**

**Type=oneshot**

**ExecStart=/home/client/script.py – расположение нашего файла(скрипта)**

**[Install]**

**WantedBy=multi-user.target**

Перейдем к сбору данных с Клиента, в моем случаем, я решил собирать информацию о запущенных приложениях на ВМ, мною были выбраны такие приложения, как:

1)*LibreOffice –* Аналог Microsoft Office на Windows

2)*Gnumeric* - Аналог таблицы Microsoft Excel на Windows

3)*Firefox* – Браузер

4)Терминал

5)Диспетчер задач

6)Настройки Linux

7)PlayOnLinux – Мною установленное приложение, через которое можно установить разного рода программы, которые нельзя напрямую поставить в Linux, специально установил, чтобы через него можно было отслеживать компрометацию Клиента.

8)Утилита "Расширенную конфигурацию сети"- через нее можно редактировать подключение к сети

9) Утилита "Пароли и Ключи"- через нее можно смотреть хранилище ключей

Из вышеперечисленных приложений, только первые 3 будут имитировать легитимную работу сотрудника, остальные же программы, которые позволяют редактировать настройки операционной системы, не должны запускаться сотрудником вовсе, либо же могут быть запущены администратором в некоторой своей комбинации в рабочее время. В любом другом случае можно предположить, что Клиент был атакован.

Сбор информации ведется через терминал, название каждого процесса из списка для **Ubuntu** версии **22.04,** привожу ниже в виде процесс-название процесса в диспетчере задач*:*

1)LibreOffice – libreoffice

2)Gnumeric - gnumeric

3)Firefox – firefox

4)Терминал – gnome-terminal-server

5)Диспетчер задач – gnome-system-monitor

6)Настройки Linux – gnome-control-center

7)PlayOnLinux – mainwindow.py

8)Утилита "Расширенную конфигурацию сети"- nm-connection-editor

9) Утилита "Пароли и Ключи"- seahorse

1. **На Сервере\_ML разработать с использованием Python/C/C++ ПО Детектор, который будет регулярно принимать данные от Агента, оцифровывать их и выполнять классификацию или обнаружение аномалий с помощью ML. Механизм Детектора остается на усмотрение исполнителя.**

Реализация подключения Сервера к Клиенту и ПО Детектор так же написаны на С++. Для того чтобы их подключить между собой мы прописываем в коде семейство адресов **IPv4** и указываем адрес Сервера, к которому будет подключаться Клиент, в моем случае, это 192.168.133.5, данный адрес мы получили вследствие того, что подключили 2 адаптер к ВМ в виде **Виртуальный адаптер хоста** – VirtualBox Host-Only Ethernet Adapter*.*

Реализация ML была вынесена в скрипт на Python, данный скрипт будет вызываться в основном приложении на C++, после получения информации от Клиента. Логическое дерево выглядит так: Клиент передает информацию Серверу 🡺 Сервер оцифровывает их, после вызывает скрипт на Python и передает в него информацию, скрипт в свою очередь исследует на аномальность полученные данные 🡺 Если аномалии выявлены, предупреждает об этом, выводя сообщение об этом в тестовый документ.

Так же отдельно от всего собирается информация о входе клиента в систему, данная информация считывается с файла **wtmp**, который находится по пути **/var/log,** собранная информация выглядит так:

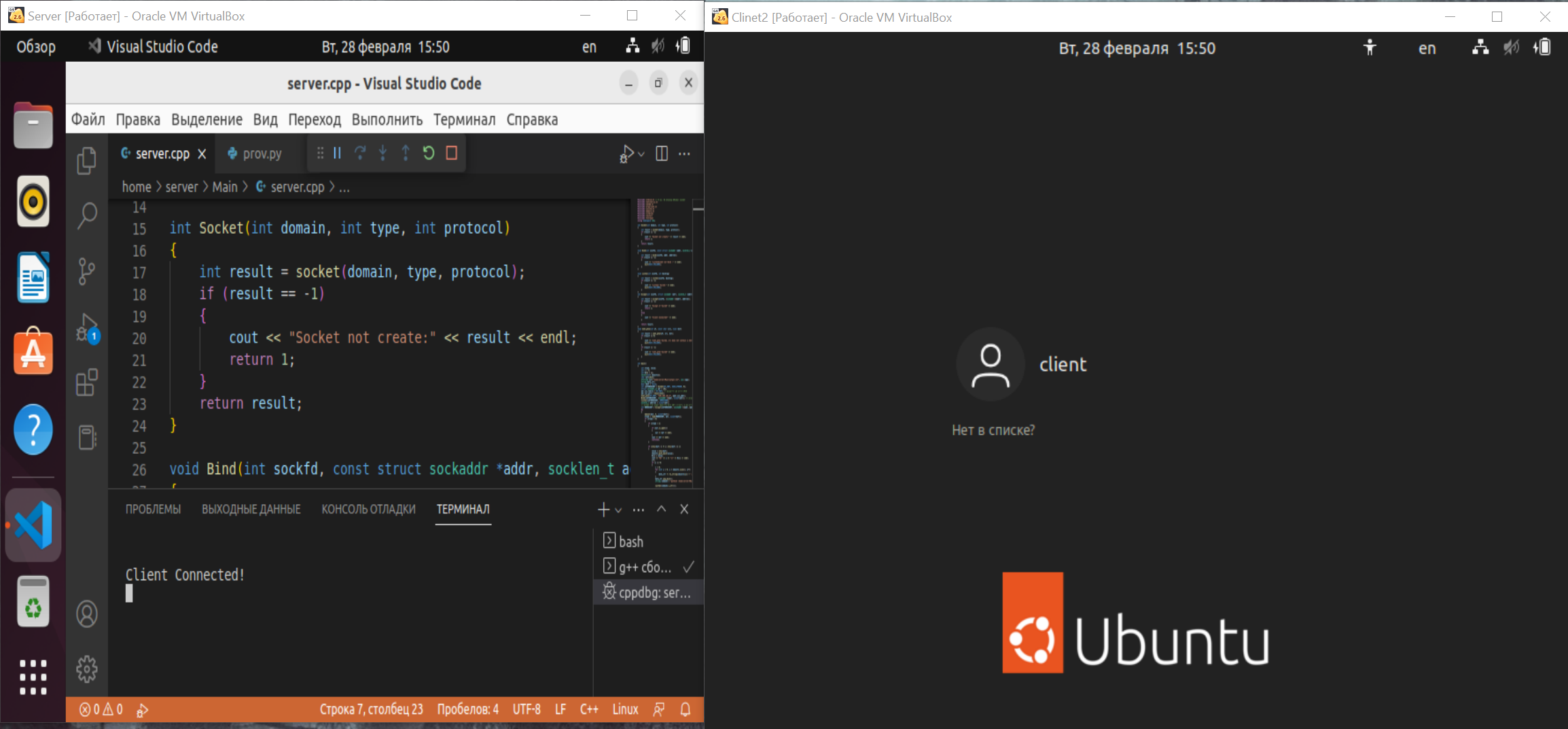


Выводится она поверх результирующей информации о состоянии Клиента.

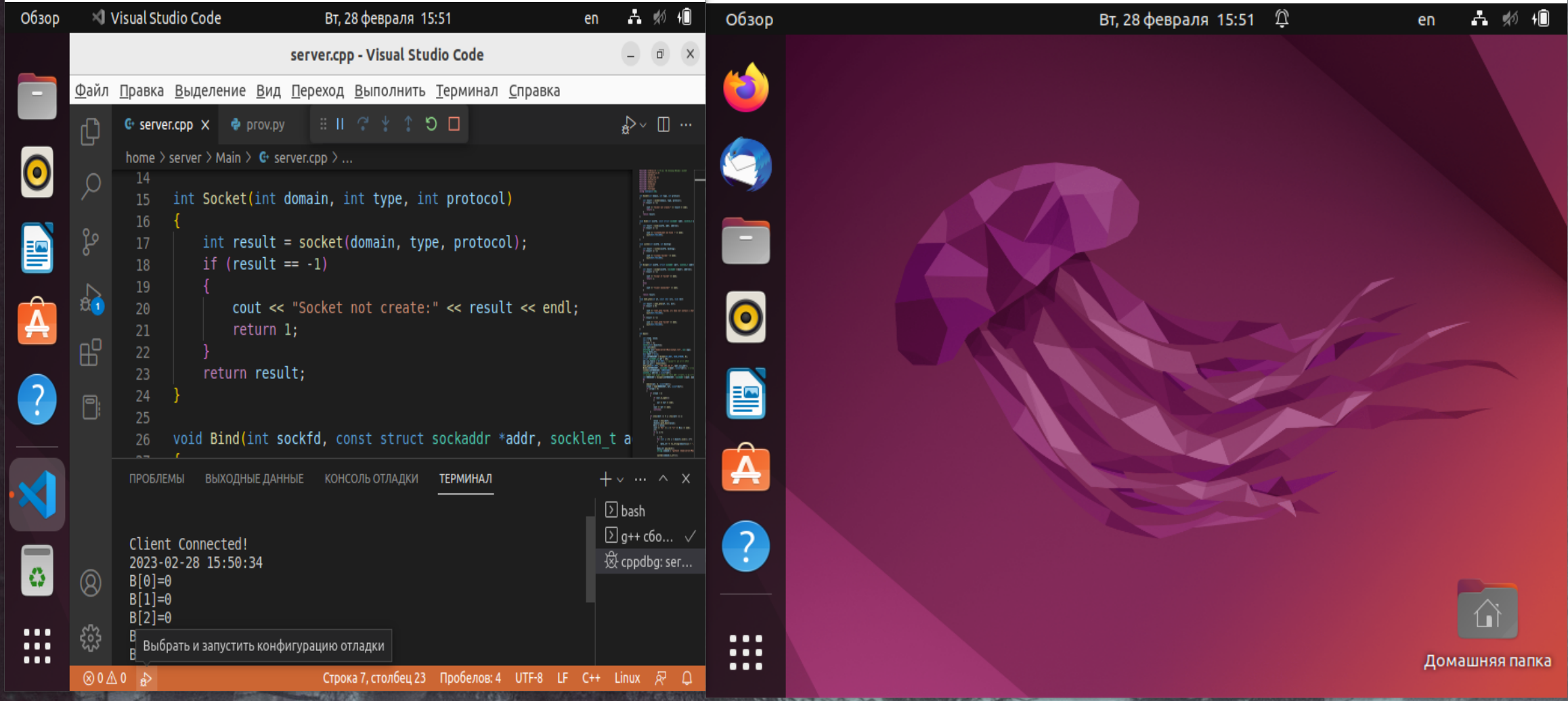
1. **Провести тестирование всей системы, демонстрирующее, что при выполнении аномальных операций на Клиенте Сервер\_ML их своевременно (в пределах периода получения данных от Клиента) обнаруживает и выдает сообщение об этом на консоль или в файл.**

Тестируем выявление атаки на клиент:

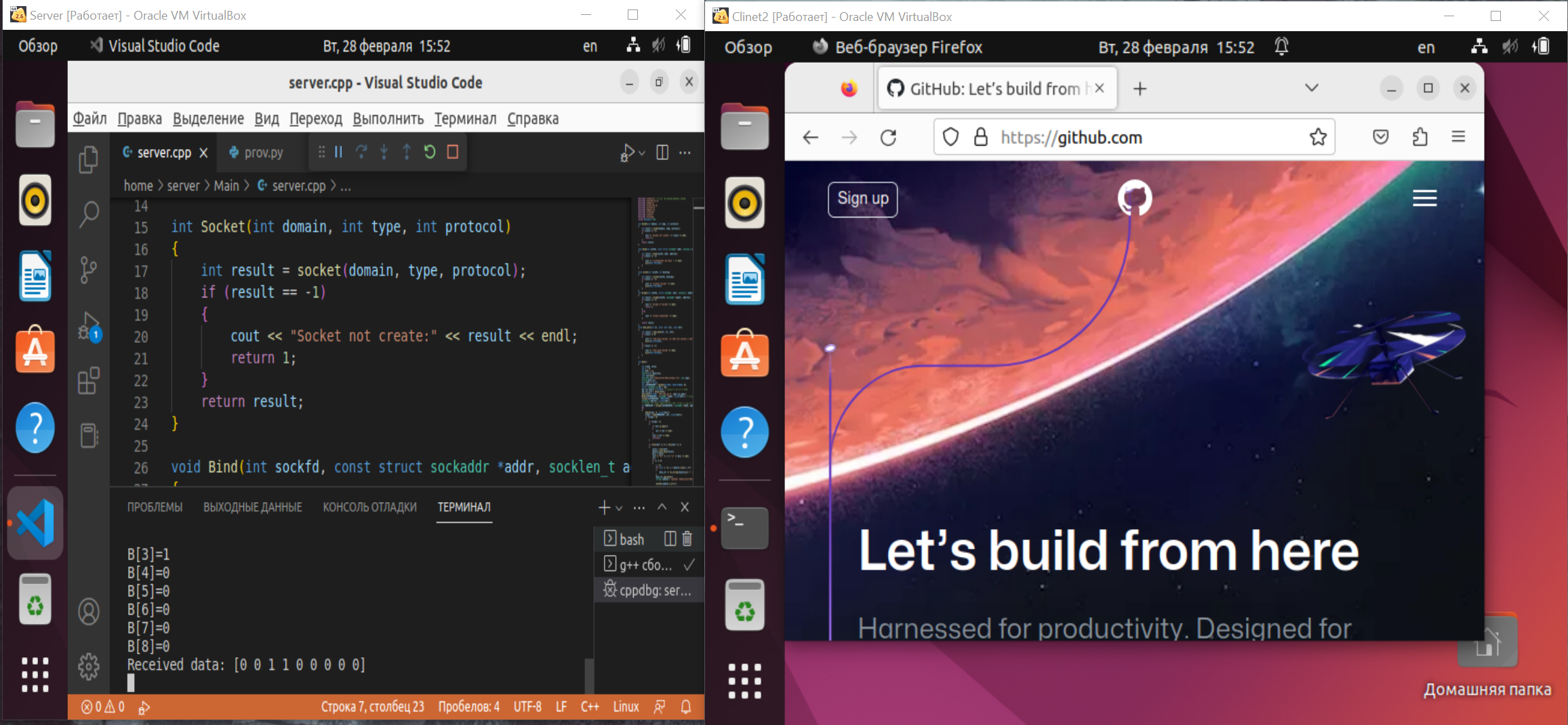
**1**)Подключение Клиента к Серверу, сразу после запуска:

Скриншот (4)

Как только Клиент авторизовался, информация о входе была записана в файл “output.txt” и пошел сбор информации, как на Скриншоте (5)

Скриншот (5)

2) Передача информации от Клиента к Серверу:

Скриншот (6)

Информация, которая выводится в Терминале Сервера:

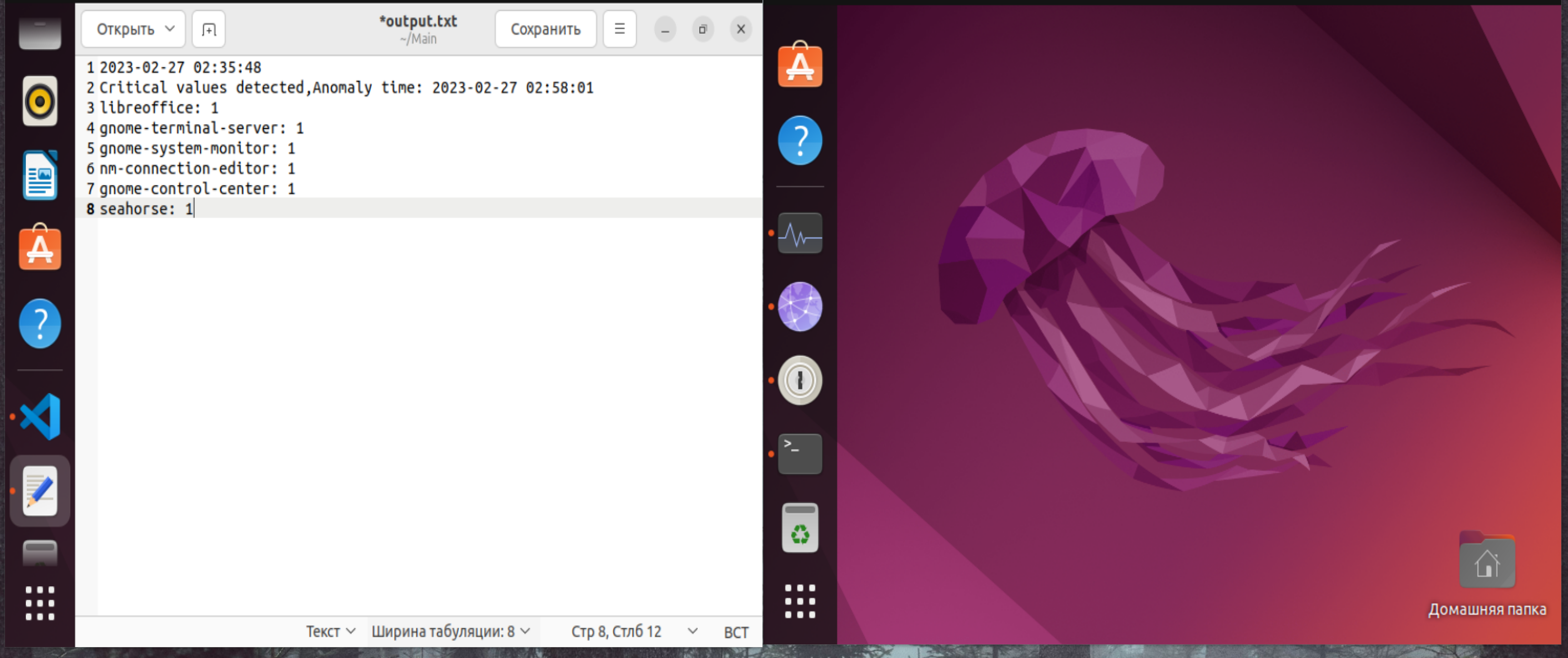
1. Заполняющийся массив, это логические данные, говорящие о запуске разных процессов, на скриншоте запущен Firefox и Терминал.

2. Следующую строчку выводит скрипт на Python, она сигнализирует об успешном принятии данных скриптом.

Проверка на обнаружение Аномалий с помощью ML:

Создадим такую ситуацию, при которой будет запущены такие процессы, как: Терминал, Системный монитор, Расширенная конфигурация сетей, Настройки, Пароли и Ключи и LibreOffice.

Из всех вышеперечисленных процессов только LibreOffice является легитимным процессом, наш скрипт на Python распознал атаку на Клиент и вывел информацию о запущенных процессах и времени, когда данное происшествие произошло в файл “output.txt”.

****

Скриншот (7)

Как видно из Скриншота (7), ПО Детектор отрабатывает корректно, он выписывает нелегитимные сессии в указанный нами файл.

Порядок вывода в файл:

1. Время, когда пользователь зашел в систему.
2. Сообщение о выявлении атаки и времени обнаружении.
3. Список запущенных процессов.