Проектная работа

Информатика

«Создание дополнительной системы безопасности для дистрибутивов Linux»

Автор: Миронов Артём Игоревич

11 класс МБОУ “Средняя общеобразовательная школа №46” г. Калуги

Научный руководитель:

Иванова Татьяна Анатольевна,

учитель физики МБОУ “Средняя общеобразовательная школа №46” г. Калуги

2023 год

ПАСПОРТ ПРОЕКТА	
Тема проекта	Создание дополнительной системы безопасности для дистрибутивов Linux
Исполнитель проекта	Миронов Артём Игоревич
Куратор проекта	Иванова Татьяна Анатольевна
Название общеобразовательного учреждения	МБОУ СОШ №46
Год разработки учебного проекта	2023
Актуальность	Любой хорошо разбирающийся в Linux человек способен осуществить несанкционированное подключение к дистрибутиву Linux клиента, что чаще всего приводит к краже персональных данных и возможному заражению целевой системы.
Проблема	Необходимость создания дополнительной системы безопасности помимо стандартного межсетевого экрана при его необходимом отключении или настройки таким образом, что появляется возможность подключения третьим лицом.
Продукт	Дополнительная система безопасности для операционной системы Linux.

Гипотеза	При необходимости отключения межсетевого экрана, отвечающего за общую безопасность клиента, или его некорректную или неаккуратную настройку, появляется возможность подключения к целевой системе третьим лицом из любой точки мира. Созданная дополнительная система безопасности разрешает только те подключения, которые находятся в базе компьютера, к которой с некоторой периодичностью обращается система безопасности с целью определения, являются ли все подключения санкционированными или нет. В случае, если подключение несанкционированно, то система клиента предпринимает все попытки для отключения нарушителя.
Цель	Обеспечить операционную систему Linux дополнительной системой безопасности.
Индикаторы	Звёзды на странице продукта, выложенного в качестве репозитория на github

Задачи	Ознакомится с литературой о том, что такое Linux, какие бывает дистрибутивы и что это такое, о видах и протоколах сетевых соединений, языке программирования Python и о том, как работать на Linux Ознакомится с самыми популярными дистрибутивами Linux и выбрать один для работы Изучить интернет на предмет наличия подобных дополнительных систем безопасности Обучиться программировать на языке программирования Python на более продвинутом уровне Изучить работу всех функций, пакетов и методов, необходимых для написания рабочего кода будущей системы безопасности Изучить команды в Linux, отвечающие за сканирование активных соединений и их отключения Написать код для системы безопасности Выполнить проверку эффективности работы системы безопасности Представить готовый продукт
Тип проекта	Практико-ориентировочный Исследовательский

ВВЕДЕНИЕ

Рассматривая переход на новую операционную систему, которой по итогу стала операционная система Linux, а конкретнее, один из его дистрибутивов, я занимался изучением вопроса информационной безопасности, так как некоторые дистрибутивы Linux является Open-Source проектами (проектами с открытым исходным кодом), что обозначает, что доступ к исходному коду может получить любой желающий человек, а это может являться причиной для беспокойства любого пользователя, поскольку можно найти “дыру” в исходном коде дистрибутива, которую любой человек может использовать не в благих намерениях. С другой стороны, доступ к исходному коду конкретного дистрибутива или самому ядру Linux открывает целый ряд возможностей как для создания собственной системы, используемой для конкретных устройств, так и утилит/программ/скриптов, выполняющих свои функции на программном уровне компьютера. Но именно после ряда хакерских атак на многие компании мне и пришла мысль о создании дополнительной системы безопасности.

Отсюда возникает вопрос, обеспечивают ли разработчики дистрибутивов Linux или систем на этом ядре полноценной системой безопасности, которая не позволит получить третьим лицом несанкционированный доступ к системе? Чтобы изучить этот вопрос, я использовал один из самых популярных дистрибутивов Linux, а именно Kali Linux. Этот дистрибутив станет платформой для моего будущего продукта.

Глава I Разработка системы безопасности

Глава I.1 Установка и настройка дистрибутива

Для начала стоит установить сам дистрибутив, ведь для создания продукта моего проекта, для него нужна платформа. Дистрибутив Kali Linux можно скачать с их официального сайта. **[17]**

После того как я скачал образ этого дистрибутива, я монтировал его на USB-накопитель, после чего вставил его в ноутбук, выданный школой специально для моего проекта и начал установку с использованием полного дискового пространства.

Процесс установки стандартный для дистрибутивов Linux: сначала выбирается имя пользователя, устанавливается пароль, выбирается имя домена (это поля я оставил пустым), выбирается диск и размер используемого для установки дискового пространства, после чего начинается процесс установки самой системы.

После завершения установки меня встретил слегка непонятный рабочий стол Kali Linux. Расположенная сверху панель задач аналогична панели задач Windows, за исключением того, что отсутствует панель поиска, а также находится иконка запуска терминала.

Поскольку я только установил систему, её следует ещё обновить. Для этого я открыл термин и прописал следующие команды:

sudo apt-get update

sudo apt-get upgrade

Поскольку в Linux некоторые команды, как эти, требуют специальных разрешений, перед ними я прописал sudo. Из-за того, что я это прописал, для выполнения команд потребуется дополнительно ввести пароль для подтверждения выполнения.

После процесса обновления я перезапустил систему. Возможно, из-за обновления, а возможно из-за некорректной установки, после ввода имени пользователя и пароля у

меня не запускается рабочий стол. Решение по обходу данной проблемы нашлось, на экране входа я нажал сочетание клавиш ctrl+alt+f3, после чего открылся терминал. Там я ввёл имя пользователя и пароль и у меня появился стандартный терминал на весь экран. Там я ввёл sudo su для того, чтобы никакие команды не требовали специальных разрешений, а также startx для запуска рабочего стола. После этих действий у меня загрузился уже привычный рабочий стол, но теперь уже вся система работает от лица суперпользователя, потому я могу видеть и редактировать все системные файлы, а также выполнять команды в терминале без приставки sudo и последующего ввода пароля.

В качестве языка программирования будет выступать Python. Python — это скриптовый язык программирования. Он универсален, поэтому подходит для решения разнообразных задач и многих платформ, начиная с iOS и Android и заканчивая серверными ОС **[18]**. Также этот язык является одним из встроенных в Linux, и в том же Mint Linux некоторые системные файлы также написаны именно с помощью Python. Гвоздь в коробку с сомнениями о выборе языка забил факт того, что этот язык изучается в рамках школьной программы. Также мной была изучена дополнительная литература по программированию на Python, чтобы повысить свои навыки в написании кода. **[19]**

Но для того, чтобы полноценно работать с этим языком программирования, для начала его следует обновить, поскольку в Linux установлена уже устаревшая версия. Сделал я это с помощью следующей команды:

apt install python-is-python3 **[20]**

После этого Python был обновлён до актуальной версии и теперь готов для того, чтобы я мог начать разработку продукта моего проекта.

Глава I.2 Изучение используемых системных команд

Для полноценной работы системы безопасности хватит всего двух команд, к которым с некоторый периодичностью или при определённом условии и будет

обращаться код. Этими двумя командами стали arp и tcpkill. Мануалы к командам в приложении 1 и 2 соответсвенно.

В моём случае arp будет служить для записи всех активных соединений в файл и их последующего чтения.

Глава I.3 Алгоритм и написание кода

Теперь после того, как был определён и обновлён язык программирования и изучены команды, к которым будет обращаться код системы безопасности, можно приступить к алгоритму её работы, а также написанию кода и объяснению каждого его блока. Состоять система безопасности будет из двух текстовых файлов: белый список (разрешённые подключения) а также буфер, в который будет записывать вывод команды arp вторым скриптом, а также двух скриптов Python: autotcpkill (сканирует все активные соединения и заносит их белый список, после чего запускает второй скрипт) и connection_check (сканирует активные соединения, занося их в файл-буфер и проверяет каждое на предмет санкционированности). Алгоритм работы системы безопасности следующий:

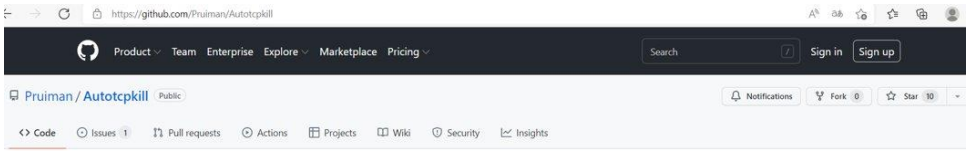
Сканирование активных соединений первым скриптом autotcpkill и занесение их в белый список (whitelist.txt)
Запуск второго скрипта connection_check
Начало цикла: сканирование активных соединений и занесение их буфер (buffer.txt)
Занесение ip-адресов и их интерфейсов подключения из буфера и обработка в список n2
Занесение ip-адресов из белого списка в список r
Проверка каждого соединения из списка n2 на предмет санкционированности путём сравнения со списком r
Если обнаружено несанкционированное соединение, запускается команда tcpkill, в которую вносятся ip-адрес

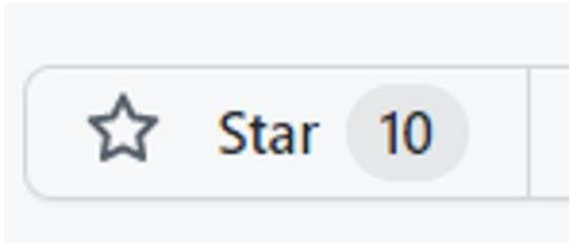
нарушителя и его интерфейс подключения
Повтор цикла через 10 секунд

Глава II Индикаторы и экономическое обоснование.

В качестве индикаторов моего проекта выступают звёзды на github, где и выложен продукт моего проекта в качестве репозитория. На момент создания этого документа и презентации на репозитории уже имеется 10 звёзд. Такое количество звёзд обусловлено тем, что спрос на такие системы безопасности невелик и на самом github не имеется таких же проектов как мой, или их настолько мало и они настолько непопулярны, что они не высвечиваются по запросу tcpkill или, как в моём случае, autotcpkill, либо никто не выкладывал такие проекты в качестве репозитория вовсе. Звёзды на github — это показатель очень высокого рейтинга, потому что просто так их никто из пользователей не ставит, их ставят только те пользователи, которые скачали этот репозиторий и не просто удостоверились эффективностью работы, но ещё и остались довольны.

Страница репозитория на github: https://github.com/Pruiman/Autotcpkill





Экономическим обоснованием моего проекта является то, что всё программное обеспечение, использованное для создания, проекта является свободным и его можно бесплатно скачать с официальных сайтов разработчиков.

Глава III Тестирование

Когда будет обнаружено несанкционированное соединение и скрипт с помощью tcpkill начнёт закрывать это соединение, у злоумышленника пропадёт всякая возможность совершения действий на машине клиента. Чтобы это доказать, мной была проведена проверка работоспособности.

На Kali Linux в терминале с правами суперпользователя был запущен второй скрипт, заранее указав ip-адрес моей Windows 10 в белом списке. На Windows 10 при помощи консоли и виртуальной локальной сети Logmein Hamachi [22] командой ssh <имя пользователя Kali Linux>@<адрес Kali Linux из Logmein Hamachi> я установил соединение с самим Kali Linux и имел полный доступ к терминалу. После того как я убрал ip-адрес Windows 10 из белого списка, скрипт посчитал это соединение несанкционированным и активировал команду tcpkill. Теперь после совершения любого действия в консоли Windows 10, подключенной к Kali Linux, я был отключен и потерял всякий доступ к Kali и возможность подключения к нему, поскольку команда tcpkill в реальном времени будет отключать несанкционированный ip-адрес пока пользователь вручную не отключит выполнение этой команды с помощью сочетания клавиш ctrl+c и цикл кода не запустится заново.

Но отсюда и вытекает главный, но не существенный с точки зрения безопасности минус: система безопасности может отключать только одно несанкционированное подключение. Но почему этот минус является несущественным? Всё по причине того, что вероятность взлома рядового пользователя 2-мя и более злоумышленниками крайне мала.

Если не брать во внимание этот минус, то эффективность работы моей системы безопасности была полностью доказана и рядовые пользователи могут абсолютно спокойно использовать её, скачав с моей страницы github, где я и выложил её в качестве репозитория.

Заключение

Таким образом и была создана дополнительная система безопасности для дистрибутивов Linux, которая действительно начала пользоваться спросом у рядовых пользователей. Минусами моего продукта является невозможность отключения от клиента двух и более нарушителей, а также необходимость специальных разрешений для запуска, что не позволяет добавить дополнительную систему безопасности в список автозагрузки. В будущем планируется добавить к моей системе безопасности шифрование белого списка разрешённых ip-адресов, а также обращение к внешнему серверу, который уже будет передавать клиенту разрешение для активного соединения или давать сигнал о том, что его стоит закрыть.

Список использованной литературы

[1] Поляков, К.Ю. Информатика. 10 класс (базовый и углубленный уровни) (в 2 частях): учебник. Ч.2 / К.Ю. Поляков, Е.А. Еремин. - 2-е изд., стереотип. - М. : БИНОМ. Лаборатория знаний, 2020. - 350, [2]с. : ил. - ISBN 978-5-9963-5454-2 (Ч. 1).

[2] Поляков, К.Ю. Информатика. 10 класс (базовый и углубленный уровни) (в 2 частях): учебник. Ч.2 / К.Ю. Поляков, Е.А. Еремин. - 2-е изд., стереотип. - М. : БИНОМ. Лаборатория знаний, 2020. - 351, [1]с. : ил. - ISBN 978-5-9963-5455-9 (Ч. 2).

[3] Материал с сайта блог.скиллфактори.ру: Linux. [Электронный ресурс] / blog.skillfactory.ru - Режим доступа: https://blog.skillfactory.ru/glossary/linux/

[4] Материал с сайта ртфм.ко.уа: Linux: Архитектура ядра Linux - общая архитектура системы. [Электронный ресурс] / rtfm.co.ua - Режим доступа: https://rtfm.co.ua/linux-arxitektura-yadra-linux-obshhaya-arxitektura-sistemy/

[5] Материал с сайта хабр.ком: Обзор Kali Linux 2021.2. [Электронный ресурс] / habr.com - Режим доступа: https://habr.com/ru/company/ruvds/blog/566164/

[6] Материал с сайта таймвэб.ком: Kali Linux для начинающих [Электронный ресурс] / timeweb.com - Режим доступа: https://timeweb.com/ru/community/articles/kali-linux-dlya-nachinayushchih

[7] Материал с сайта дэвсдэй.ру: IT-блоги • История Linux Mint | Losst [Электронный ресурс] / devsday.ru - Режим доступа: https://devsday.ru/blog/details/6723#:~:text=Linux%20Mint%20%D1%80%D0%B0%D0%B7%D1%80%D0%B0%D0%B1%D0%BE%D1%82%D0%B0%D0%BD%20%D0%B8%20%D0%B2%D1%8B%D0%BF%D1%83%D1%89%D0%B5%D0%BD,%D0%B8%D0%B7%20%D0%BE%D1%81%D0%BD%D0%BE%D0%B2%D0%BD%D1%8B%D1%85%20%D1%80%D0%B0%D0%B7%D1%80%D0%B0%D0%B1%D0%BE%D1%82%D1%87%D0%B8%D0%BA%D0%BE%D0%B2%20Linux%20Mint

[8] Материал с сайта линуксминт.ком: Образ для установки дистрибутива Linux Mint на компьютер [Электронный ресурс] / linuxmint.com - Режим доступа: https://linuxmint.com/download.php

[9] Материал с сайта ру.теастроджипэйдж.ком: Что такое сетевое подключение? - определение из техопедии - сети - 2022 [Электронный ресурс] / ru.theastrologypage.com - Режим доступа: https://ru.theastrologypage.com/network-connectivity

[10] Материал с сайта тридаблъю.айбиэм.ком: Сетевые соединения [Электронный ресурс] / www.ibm.com - Режим доступа: https://www.ibm.com/docs/ru/i/7.3?topic=networking-network-communications

[11] Материал с сайта тридаблъю.айбиэм.ком: Протокол TCP/IP [Электронный ресурс] / www.ibm.com - Режим доступа: https://www.ibm.com/docs/ru/aix/7.2?topic=management-transmission-control-protocolinternet-protocol

[12] Материал с сайта тридаблъю.айбиэм.ком: Протоколы TCP/IP [Электронный ресурс] / www.ibm.com - Режим доступа: https://www.ibm.com/docs/ru/aix/7.2?topic=protocol-tcpip-protocols

[13] Материал с сайта тридаблъю.айбиэм.ком: Маршрутизация TCP/IP [Электронный ресурс] / www.ibm.com - Режим доступа: https://www.ibm.com/docs/ru/aix/7.2?topic=protocol-tcpip-routing

[14] Материал с сайта тридаблъю.айбиэм.ком: Шлюзы маршрутизации TCP/IP [Электронный ресурс] / www.ibm.com - Режим доступа: https://www.ibm.com/docs/ru/aix/7.2?topic=routing-tcpip-gateways

[15] Материал с сайта тридаблъю.айбиэм.ком: Протоколы шлюза [Электронный ресурс] / www.ibm.com - Режим доступа: https://www.ibm.com/docs/ru/aix/7.2?topic=gateways-gateway-protocols

[16] Материал с сайта тридаблю.айбиэм.ком: Статическая и динамическая маршрутизация [Электронный ресурс] / www.ibm.com - Режим доступа: https://www.ibm.com/docs/ru/aix/7.2?topic=routing-static-dynamic

[17] Материал с сайта кали.орг: Образ для установки дистрибутива Kali Linux на компьютер [Электронный ресурс] / kali.org - Режим доступа: https://www.kali.org/get-kali/#kali-bare-metal

[18] Материал с сайта скиллбокс.ру: Язык программирования Python: Преимущества, недостатки и область применения [Электронный ресурс] / skillbox.ru - Режим доступа: https://skillbox.ru/media/code/dlya_chego_nuzhen_python/

[19] Бэрри Пол. Изучаем программирование на Python / Пол Бэрри ; [пер. с англ. М.А. Райтман]. - Москва: издательство «Э», 2017. - 624 с. : ил. - (Мировой компьютерный бестселлер). ISBN 978-5-699-98595-1

[20] Материал с сайта залинукс.ру: Переход на Python 3 в Debian и Kali Linux [Электронный ресурс] / zalinux.ru - Режим доступа: https://zalinux.ru/?p=5527

[21] Материал с сайта лосст.ру: Запуск Python скрипта в Linux [Электронный ресурс] / losst.ru - Режим доступа: https://losst.ru/zapusk-python-skripta-v-linux

[22] Материал с сайта тридаблъю.впн.нэт: Скачивание клиента Logmein Hamachi [Электронный ресурс] / losst.ru - Режим доступа: https://www.vpn.net/

[23] Страница репозитория продукта проекта на сайте гитхаб.ком: Autotcpkill [Электронный ресурс] / github.com - Режим доступа: https://github.com/Pruiman/Autotcpkill

Приложение 1

НАЗВАНИЕ

arp - управление системным кэшем ARP

СИНТАКСИС

arp [-evn] [-H type] [-i if] -a [имя хоста]

arp [-v] [-i if] -d имя хоста [pub]

arp [-v] [-H type] [-i if] -s имя хоста hw_addr [temp]

arp [-v] [-H type] [-i if] -s имя хоста hw_addr [сетевая маска nm] pub

arp [-v] [-H type] [-i if] -Ds имя хоста ifa [сетевая маска nm] pub

arp [-vnD] [-H type] [-i if] -f [имя файла]

ОПИСАНИЕ

Arp манипулирует кэшем ARP ядра различными способами. Основными опциями являются очистка записи сопоставления адресов и ее настройка вручную. В целях отладки программа arp также позволяет выполнять полный дамп кэша ARP.

ОПЦИИ

-v, --verbose

Расскажите пользователю, что происходит, будучи подробным.

-n, --numeric

показывает числовые адреса вместо того, чтобы пытаться определить символические имена хоста, порта или пользователя.

-H тип, -hw-type тип, -t тип

При настройке или чтении кэша ARP этот необязательный параметр сообщает arp, какой класс записей он должен проверять. Значение этого параметра по умолчанию равно ether (т.е. аппаратный код 0x01 для IEEE 802.3 10Mbps Ethernet). Другие значения могут включать сетевые технологии, такие как ARCnet (arcnet) , PROnet (pronet) , AX.25 (ax25) и NET/ROM (netrom).

-a [имя хоста], --display [имя хоста]

Показывает записи указанных хостов. Если параметр hostname не используется, будут отображены все записи. Записи будут отображаться в альтернативном стиле (BSD).

-d имя хоста, --delete имя хоста

Удалите любую запись для указанного хоста. Это может быть использовано, например, если указанный хост выведен из строя.

-D, --use-device

Используйте аппаратный адрес интерфейса ifa.

-e

Показывает записи в стиле по умолчанию (Linux).

-i If, -device If

Выберите интерфейс. При сбросе кэша ARP будут напечатаны только записи, соответствующие указанному интерфейсу. При установке постоянной или временной записи ARP этот интерфейс будет связан с записью; если эта опция не используется, ядро будет угадывать на основе таблицы маршрутизации. Для записей pub указанный интерфейс — это интерфейс, на который будут отвечать на запросы ARP.

ПРИМЕЧАНИЕ: это должно отличаться от интерфейса, на который будут перенаправляться IP-дейтаграммы.

-s имя хоста hw_addr, --set hostname

Вручную создайте запись сопоставления адресов ARP для имени хоста с аппаратным адресом, установленным в класс hw_addr, но для большинства классов можно предположить, что можно использовать обычное представление. Для класса Ethernet это 6 байт в шестнадцатеричном формате, разделенных двоеточиями. При добавлении записей прокси-сервера arp (то есть записей с установленным флагом публикации) для прокси-сервера arp для целых подсетей может быть указана маска сети. Это не очень хорошая практика, но поддерживается старыми ядрами, потому что это может быть полезно. Если флаг temp не указан, записи будут постоянно храниться в кэше ARP.

ПРИМЕЧАНИЕ: Начиная с версии ядра 2.2.0, больше невозможно установить запись ARP для всей подсети. Linux вместо этого выполняет автоматический прокси-сервер arp, когда маршрут существует и он пересылается.

-f имя файла, --file имя файла

Аналогично опции -s, только на этот раз информация об адресе берется из заданного имени файла. Имя файла данных очень часто /etc/ethers, но это не является официальным. Если имя файла не указано, по умолчанию используется /etc/ethers.

Формат файла прост; он содержит только текстовые строки ASCII с аппаратным адресом и именем хоста, разделенными пробелами. Дополнительно можно использовать флаги pub, temp и netmask.

Во всех местах, где ожидается имя хоста, можно также ввести IP-адрес в десятичной системе счисления с точками. В качестве особого случая для обеспечения совместимости порядок имени хоста и аппаратного адреса может быть изменен.

Каждая полная запись в кэше ARP будет отмечена флагом C. Постоянные записи помечены символом M, а опубликованные записи имеют флаг P.

Приложение 2

НАЗВАНИЕ

tcpkill

-

отключение TCP-соединения в сети

СИНТАКСИС

tcpkill [-i интерфейс] [-1...9] выражение

ОПИСАНИЕ

tcpkill отключает указанные текущие TCP-соединения (полезно для приложений на основе libnids, которым требуется полный TCP 3-whs для создания TCB).

ОПЦИИ

-i интерфейс

Укажите интерфейс для прослушивания.

-1...9

Укажите количество грубой силы, которое необходимо использовать для прерывания соединения. Для быстрых подключений может потребоваться большее число, чтобы получить первое соединение в окне переключения соединений. Значение по умолчанию равно 3.

Приложение 3

```
#!/usr/bin/python3.8

import os, sys

n = []

c = []

b = []

bes = []


for i1 in range(10):
    for i2 in range(10):
        for i3 in range(10):
            c.append(str(i1)+str(i2)+str(i3))


command = "arp>buffer.txt"

result = os.system(command)

with open('buffer.txt', 'r') as f:
    result = f.read()
    print(result)

with open(r"buffer.txt", 'r') as file:
    for line in file:
        for i in range(len(c)):
            if (c[i] in line):
```

```
            n.append(line)
for j1 in range(len(n)):
    for j2 in range(len(n)):
        if (j1==j2):
            continue
        if (n[j1]==n[j2]):
            del n[j2]
            n.append(" ")
while " " in n:
    n.remove(" ")
for j in range(len(n)):
    b = n[j].split()
    del n[j]
    bes.append(b[0])
    n = bes + n
    b = []
    bes = []

with open(r"whitelist.txt", 'w') as file:
    print(*n, file=file, sep="\n")

os.system("./connection_check")
```

Код второго скрипта:

```
#!/usr/bin/python3.8

import sys, os, time



while True:
    n = []
    c = []
    b = []
    r = []
    n1 = []
    n2 = []
    t = ''


    for i1 in range(10):
        for i2 in range(10):
            for i3 in range(10):
                i11 = str(i1)
                i22 = str(i2)
                i33 = str(i3)
                c.append(i11+i22+i33)
```

```
command = "arp>buffer.txt"
result = os.system(command)
with open('buffer.txt', 'r') as f:
    result = f.read()
    print(result)
with open(r"buffer.txt", 'r') as file:
    for line in file:
        for i in range(len(c)):
            if ("_gateway" in line):
                continue
            if (c[i] in line):
                n.append(line)


for j1 in range(len(n)):
    for j2 in range(len(n)):
        if (j1==j2):
            continue
        if (n[j1]==n[j2]):
            del n[j2]
            n.append(" ")
while " " in n:
    n.remove(" ")
```

```
for jack in range(len(n)):
    n1.append(n[jack].split())


for jack2 in range(len(n1)):
    best1 = n1[jack2]
    n2.append(best1[0])
    n2.append("")
    n2.append(best1[-1])
    n2.append("")
del n2[-1]


for j in range(len(n)):
    b = []
    b = n[j].split()
    del n[j]
    n.append(b[0])
    n.append(n[j])
    del n[j]
with open(r"whitelist.txt", 'r') as file1:
    for line in file1:
        r.append(line)
for l1 in range(len(n)):
    for l2 in range(len(r)):
```

```
        k = 0
        if (n[l1] in r[l2]):
            k += 1
            break
    if (k == 0):
        for ste in range(len(n2)):
            if (n2[ste]==n[l1]):
                t = n2[ste+2]
        com = "tcpkill -i " + t + " host " + n[l1]
        os.system(com)
time.sleep(10)
```

Приложение 4

Сетевые соединения

Сетевое соединение описывает обширный процесс соединения различных частей сети друг с другом, например, посредством использования маршрутизаторов, коммутаторов и шлюзов.

Существует множество различных сетевых топологий, включая концентраторы, линейные, древовидные и звездообразные конструкции, каждая из которых настроена по-своему для облегчения соединения между компьютерами или устройствами. У каждого есть свои плюсы и минусы в плане сетевого подключения.

В системе могут существовать несколько различных сетевых технологий. К числу поддерживаемых протоколов относятся TCP/IP, APPC, APPN, HPR, удаленные рабочие станции, асинхронные и бинарные синхронные средства связи.

APPC, APPN, и HPR

Системная сетевая архитектура (SNA) определяет многоуровневую логическую структуру, форматы, протоколы и последовательности операций, которые обеспечивают передачу блоков информации по сети. В SNA включены, например, протоколы APPC, APPN и HPR.

Ethernet

Ethernet на платформе System i поддерживает протокол TCP/IP, архитектуры Расширенное равноправное сетевое взаимодействие (APPN) и Расширенные средства межпрограммной связи (APPC), коммуникации розничной торговли, финансовые коммуникации, хосты и удаленные рабочие станции.

OptiConnect

OptiConnect — это системная сеть (SNA) IBM System i, которая обеспечивает высокоскоростное соединение между множеством систем локальной среды с применением технологий глобальной сети (WAN) и локальной сети (LAN). [10]

Одним из самых популярных протоколов подключения является TCP/IP, потому именно с этим протоколом и будет работать моя система безопасности, поэтому теперь разберёмся с самим протоколом.

Протоколы TCP/IP

Протоколом называется набор правил, задающих форматы сообщений и процедуры, которые позволяют компьютерам и прикладным программам обмениваться информацией. Эти правила соблюдаются каждым компьютером в сети, в результате чего любой хост-получатель может понять отправленное ему сообщение. Набор протоколов TCP/IP можно рассматривать как многоуровневую структуру. Протоколы (или наборы правил) применяются для упорядочивания обмена данными между компьютерами. TCP/IP — это набор протоколов, который задает стандарты связи между компьютерами и содержит подробные соглашения о маршрутизации и межсетевом взаимодействии. TCP/IP широко применяется в Internet, поэтому с его помощью могут общаться пользователи из исследовательских институтов, школ, университетов, правительственных учреждений и промышленных предприятий. Основные функции семейства протоколов TCP/IP: электронная почта, передача файлов между компьютерами и удаленный вход в систему.

TCP/IP обеспечивает связь подключенных к сети компьютеров, обычно называемых хостами. Любую сеть можно подключить к другой сети и организовать связь с ее хостами. Несмотря на то, что существуют различные сетевые технологии, многие из которых основаны на коммутации пакетов и потоковом режиме передачи, набор протокол TCP/IP обладает одним важным преимуществом: он обеспечивает аппаратную независимость.

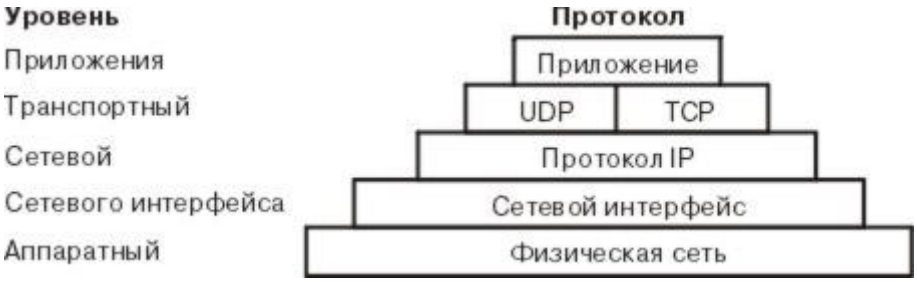
TCP/IP обеспечивает средства, позволяющие вашему компьютеру выступать в роли хоста Internet, который может подключиться к сети и установить соединение с любым другим хостом Internet. В TCP/IP предусмотрены команды и средства, которые позволяют выполнять следующие действия:

- Передавать файлы в другую систему
- Входить в удаленную систему

- Выполнять команды в удаленной системе
- Печатать файлы в удаленной системе
- Отправлять электронные сообщения удаленным пользователям
- Вести интерактивный диалог с удаленными пользователями

Управлять сетью **[11]**

На этом рисунке показан стек протоколов TCP/IP.



В протоколе TCP/IP строго зафиксированы правила передачи информации от отправителя к получателю. Сообщение или поток данных приложения отправляется протоколу Internet транспортного уровня, то есть Протоколу пользовательских дейтаграмм (UDP) или Протоколу управления передачей (TCP). Получив данные от приложения, эти протоколы разделяют всю информацию на небольшие блоки, которые называются пакетами. К каждому пакету добавляется адрес назначения, а затем пакет передается на следующий уровень протоколов Internet, то есть сетевой уровень.**[12]**

Приложение 5

Что такое Linux и где его применяют

Linux — это семейство операционных систем, работающих на основе одноимённого ядра. Операционной системы Linux не существует, потому под этой ОС подразумевают конкретные дистрибутивы. Дистрибутивами можно назвать модификации ОС, созданные для своих целей разными компаниями.

Ядро Linux разработал Линус Торвальдс в 1991 году. Через некоторое время этот проект получил широкое распространение в сообществе программистов, так как им стала интересна идя свободного программного обеспечения, и в работе над ним стали помогать специалисты со всего мира. Логотипом стал пингвин, по причине того, что они нравились Линусу Торвальдсу.

Но отсюда так или иначе возникает вопрос, почему в качестве платформы для своего проекта я выбрал именно Linux? Одной из главных причин является то, что, как было сказано выше, это платформа для, в первую очередь, свободного ПО. Также он получил широкое распространение по всему миру в самых разных устройствах:

Веб-серверы. Согласно рейтингу аналитического агентства W3Techs, на Linux-серверах развёрнуты 75,1% сайтов.

Суперкомпьютеры. Они уникальны, потому для каждого требуется операционная система, способная решать конкретные задачи. Благодаря открытому исходному коду Linux разработчики способны видоизменять его так, как требуют поставленные задачи.

Игровые консоли. На базе ядра Linux создана относительно популярная операционная система SteamOS, использующаяся в недавно вышедшем Steam Deck.

Умная техника. Компания Samsung разработала операционную систему Tize, LG – WebOS, Panasonic и Philips используют FirefoxOS.

Транспорт. Tesla используют ядро в бортовых компьютерах своих электромобилей. **[3]**

Также ещё одной веской причиной, почему я выбрал именно Linux стало повышение спроса на отечественные ОС со стороны госкомпаний и органов власти в шесть раз, о чём сообщили представители “Ред Софт” и “Мой офис”.

Строение Linux, плюсы и минусы данной ОС

Теперь после того, как мы разобрались что такое Linux, где его применяют и почему я его выбрал в качестве платформы для своего проекта, давайте разберёмся, из чего состоит сам Linux и какими плюсами и минусами обладает данная ОС.

Ядро. Оно представляет собой своего рода виртуальную машину для процессов. Процессы работают без всякой информации про оборудование компьютера – ядро абстрагирует весь уровень оборудования в единый совместимый виртуальный интерфейс. Кроме того, ядро реализует многозадачность прозрачно для всех процессов – каждый процесс “думает”, что он является единственным процессом в системе, и имеет полные и эксклюзивные права на память и другие ресурсы оборудования компьютера. Фактически же – ядро выполняет несколько процессов одновременно, и оно ответственно за распределение ресурсов оборудования таким образом, чтобы каждый процесс получил достаточный доступ к этим ресурсам.

Планировщик процессов. Он отвечает за контроль над доступом процессов к процессору. Планировщик обеспечивает такое поведения ядра, при котором все процессы имеют справедливый доступ к ресурсам центрального процессора.

Менеджер памяти. Он обеспечивает различным процессам безопасный доступ к основной памяти системы. Кроме того, обеспечивает работу виртуальной памяти, которая позволяет процессам использовать больше памяти, чем реально доступно в системе. Выделенная, но неиспользуемая память вытесняется на файловую систему, и при необходимости – возвращается из неё обратно в память.

Виртуальная файловая система. Она создаёт абстрактный слой, скрывая детали оборудования, предоставляя общий файловый интерфейс для всех устройств. Кроме

того, поддерживает несколько форматов файловых систем, которые совместимы с другими операционными системами.

Сетевые интерфейсы. Они обеспечивает работу с различными сетевыми стандартами и сетевым оборудованием. Именно они позже и будут интересовать нас больше всего.

Межпроцессная подсистема. Она поддерживает несколько механизмов для process-to-process связей в единой Linux-системе. **[4]**

Системные утилиты. Утилиты — вспомогательные компьютерные программы в составе общего ПО. Они нужны для выполнения типовых задач, связанных с работой оборудования и ОС. У Linux есть набор простых утилит. Они позволяют, например, редактировать данные, изменять расположение файлов.

Системные библиотеки. Системные библиотеки — это специальные программы, дающие доступ к функциям ядра. Для выполнения какой-либо задачи ядро вначале получает системный вызов, исходящий от приложений. Но у каждого ядра свой набор системных вызовов, и они должны понимать формат выполнения задачи. Поэтому программисты разработали стандартную библиотеку процедур, описывающую набор системных вызовов для конкретной ОС.

Утилиты разработки ПО. При помощи трех вышеперечисленных компонентов операционная система сможет запускаться и функционировать. Но для обновления и создания новых программ нужно иметь дополнительные библиотеки и инструменты — toolchain. Этот набор программ, инструментов и утилит поможет создавать рабочее приложение из исходных кодов.

Пользовательские программы. Они не считаются обязательными компонентами ОС. Нередко их пишут сами пользователи, как в случае с моим проектом. Программы помогают задать конкретную работу. К таким утилитам относятся браузеры, офисные пакеты, инструменты графического дизайна, плееры и пр.

Плюсы Linux:

Бесплатное использование. Использование большинства ОС Linux и большей части программ, основанных на ней, абсолютно бесплатно.

Открытый исходный код. Благодаря доступу к исходному коду можно изучать его, изменять, распространять, а также публиковать изменения в соответствии с лицензией.

Актуальность и производительность. По сравнению с Windows Linux не устаревает со временем. То есть, чтобы восстановить первоначальную производительность, не нужно будет регулярно чистить или переустанавливать ОС, запускать дефрагментацию.

Универсальность. Linux поддерживает практически все популярные языки программирования: Java, C/C++, Python, Ruby, C# и другие. Менеджер пакетов поможет установить и обновить целые и отдельные части компонентов ПО. Благодаря поддержке SSH можно быстро управлять серверами.

Дистрибутивы. Многие организации модифицировали ОС Linux, выпустив собственные дистрибутивы. Наиболее популярные из них: Debian, Ubuntu, Linux Mint, Arch Linux, MX Linux, Fedora, Manjaro, CentOS, Kali Linux. Linux Mint и Ubuntu подойдут начинающим пользователям, а Arch Linux, Fedora и Debian — опытным разработчикам. Дистрибутив можно собрать и самостоятельно.

Установка ПО из централизованного места - репозитория. Это место, где хранятся данные. Благодаря этому можно установить несколько программ одним щелчком мыши. Можно забыть о поиске кряков, серийных ключей и программ в интернете — с Linux это точно не понадобится.

Но не всё так прекрасно у этой ОС, как оно могло показаться, присутствуют и свои довольно существенные для рядового пользователя Windows минусы:

Сложности с освоением. Интерфейс большей части версий Linux значительно отличается от привычных Windows и MacOS.

Управление операционной системой через терминал. Консоль позволяет управлять операционной системой через ввод текста. Там же высвечиваются ответы ОС. Терминал — это современный аналог консоли, он отображается в виде окна на фоне рабочего стола.

Требование прав доступа к файлам для работы части программ. Для пользователя это значит, что придется вводить пароль как минимум по несколько раз за рабочий сеанс.