Министерство образования и науки Российской Федерации

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ

ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ

ВЫСШЕГО ПРОФЕССИОНАЛЬНОГО ОБРАЗОВАНИЯ

«САРАТОВСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

ИМЕНИ Н.Г.ЧЕРНЫШЕВСКОГО»

Кафедра теоретических основ компьютерной безопасности и криптографии

**Выявление протоколов удаленного доступа**

КУРСОВАЯ РАБОТА

студента 3 курса 331 группы

специальности 10.05.01 «Компьютерная безопасность»

факультета компьютерных наук и информационных технологий

Енца Михаила Владимировича

Научный руководитель

профессор, к.ф.-м.н. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ А.В. Гортинский

подпись, дата

Зав. кафедрой

профессор, к.ф.-м.н. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ В.Н. Салий

подпись дата

Саратов 2017 г.

**ОГЛАВЛЕНИЕ**

[ВВЕДЕНИЕ 2](#_Toc493947135)

[1. Методы и средства удаленного доступа. 4](#_Toc493947136)

[1.1. Методы удаленного доступа. 4](#_Toc493947137)

[1.2 Программы удаленного администрирования 5](#_Toc493947138)

[2. Протоколы удаленного доступа 8](#_Toc493947139)

[2.1 Протокол RDP 8](#_Toc493947140)

[2.2 Протокол Telnet 9](#_Toc493947141)

[2.3 Протокол пересылки файлов FTP. 9](#_Toc493947142)

[2.4 Протокол TFTP. 10](#_Toc493947143)

[2.5 Протоколы защищенных каналов SSL, PPTP, IPSec 10](#_Toc493947144)

[2.6 Усугубление проблем безопасности при удаленном доступе. 13](#_Toc493947145)

[2.7 Протокол РАР. 18](#_Toc493947146)

[2.8 Протокол S/Key. 19](#_Toc493947147)

[3. Детектирование сеанса удаленного доступа. 22](#_Toc493947148)

[4. Практическая часть 24](#_Toc493947149)

[4.1 Перехват пакетов 24](#_Toc493947150)

[4.2. Анализ наличия попытки удаленного доступа. 26](#_Toc493947151)

[ЗАКЛЮЧЕНИЕ 29](#_Toc493947152)

[СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ 30](#_Toc493947153)

[ПРИЛОЖЕНИЕ 1 31](#_Toc493947154)

# ВВЕДЕНИЕ

В настоящее время эффективная работа, связанная с анализом и обработкой информации, невозможна без использования компьютерных сетей. Частные сети, сеть Интернет предоставляют доступ к базам данных. Кроме того, пользователь получает возможность коллективной работы и оперативных консультаций с коллегами и экспертами.

Однако физическая реализация локальной сети не всегда возможна и/или удобна. Часто складывается ситуация, когда подразделения географически удалены друг от друга или работники находятся не в пределах локальной контролируемой зоны. В этом случае важно организовать работу таким образом, чтобы, находясь в отдалении от локальной сети, они, тем не менее, могли подключиться к ней.

Для того чтобы обеспечить доступ к услугам локальной сети при отсутствии непосредственного подключения к ней, служат средства удаленного доступа к сети. Эти средства использую для связи удаленного компьютера с локальной сетью. Они обеспечивают передачу данных на любые расстояния.

В то же время не следует забывать о том, что данные в подобных системах передаются на большое расстояние и, часто, по сетям общего пользования, доступ к которым не ограничен.

Актуальность, безопасности передаваемых данных и их защиты от несанкционированного доступа в системах удаленного доступа требует особенно пристального внимания и применения специальных методик, которые давали бы достаточную конфиденциальность и достоверность данных. В противном случае, злоумышленник, подключившийся в качестве удаленного клиента или перехватывая трафик реально существующего клиента, сможет украсть данные являющееся конфиденциальными или фальсифицировать их.

Задача, решаемая данной работой связана с решением обозначенной выше проблемы и может быть сформулирована следующим образом: Проанализировать и исследовать существующие способы обеспечения безопасности работы в системах удаленного доступа.

# Методы и средства удаленного доступа.

Под средствами удаленного доступа понимаются средства, необходимые для связи небольших локальных сетей и даже удаленных отдельных компьютеров с центральной локальной сетью.

## 1.1. Методы удаленного доступа.

Метод удаленного доступа – это метод, при котором пользователь удаленного терминала с помощью специального программного обеспечения подключается по глобальной сети к другому компьютеру, как локальный узел. Удаленной управление (remote control) – это метод, который позволяет удаленному пользователю получить контроль над локальными ПК в ЛВС. Скорость проведения сеанса и его возможности зависят от характеристик управляемого ПК, так как именно на нем выполняется обработка всех сетевых команд. Коды клавиш, нажимаемых на удаленном ПК, посылаются в управляемый ПК, а все изменения на экране управляемого выводятся на экран удаленного ПК.

* Удаленное управление – используемые файлы и прикладные программы не загружаются в удаленный ПК. Недостаток метода: для выполнения одной работы задействованы два ПК.
* Метод удаленного узла (remote node) – основан на использовании сервера удаленного доступа, который служит своего рода «регулировщиком» и позволяет отдельному удаленному ПК или ЛВС связываться с центральной ЛВС. Программное обеспечение удаленного ПК, реализующее функции удаленного узла, позволяет ему функционировать как полноценному пользователю ЛВС. Как только связь установлена, телефонные линии становятся «прозрачными» и пользователь может работать со всеми ресурсами сети, как будто он сидит за ПК, непосредственно подключенным к ЛВС. Сервер удаленного доступа может быть реализован: в виде модема со встроенным специальным ПО; либо быть сервером ЛВС, на котором выполняются программы удаленного узла желательно, чтобы на удаленном ПК помимо сетевого системного ПО находилось все прикладное ПО, необходимое для сеанса связи, например, все исполняемые файлы. В противном случае их необходимо будет передавать с сетевого сервера на канал связи. Удаленный узел как каждый полноценный пользователь ЛВС, удаленный узел имеет свой сетевой адрес. Сетевая операционная система преобразует сетевые пакеты, которые нужно передать через модем, из формата протокола IP или IPX в формат, совместимый со стандартом последовательной передачи. Для программ, поддерживающих архитектуру «клиент - сервер», усиливается тенденция на программное обеспечение для удаленных узлов, так как такие программы позволяют обрабатывать большие файлы данных на серверах ЛВС, а на удаленный ПК передать только результаты обработки. [4]

## 1.2 Программы удаленного администрирования

Для реализации удаленного доступа к ЛВС существуют более десятка известного ПО. В операционных системах семейства Windows существует встроенный помощник для реализации удаленного доступа. Для его запуска нужно выполнить команду «mstsc». В появившимся окне делаем нужные настройки (параметры экрана, звука и так далее), вводите имя компьютера и имя пользователя, под которым вы должны зайти. Нажимаете кнопку "Подключить", после чего, если компьютер будет найден и подключение возможно, у вас запросят пароль пользователя.

Часто требуется удаленное подключение к системам с различными операционными системами. Для протокола, который используется в мастере удаленных подключений Windows – RDP (о нем ниже), реализации существуют практически для всех версий Windows (включая Windows CE, Phone и Mobile), Linux, FreeBSD, Mac OS X, iOS, Android, Symbian.

Также можно использовать стороннее ПО, например, Radmin, TeamViewer, Ammyy admin и т.д.

Каждая из этих программ предоставляет удобный интерфейс для реализации удаленного доступа.

Рассмотрим подробнее работу программы AeroAdmin:

При запуске программы открывается главное окно (рисунок 1)

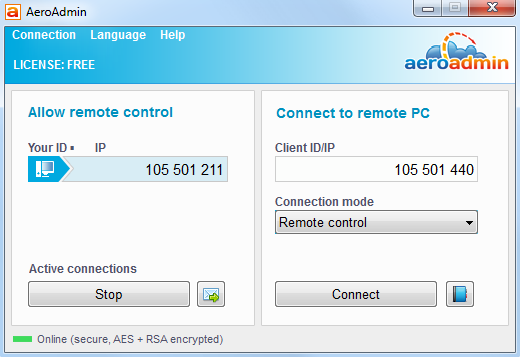


Рисунок 1 – главное окно программы AeroAdmin

Программа предоставляет интерфейс, в котором возможно удаленное управление компьютером, режим просмотра и файловый менеджер.

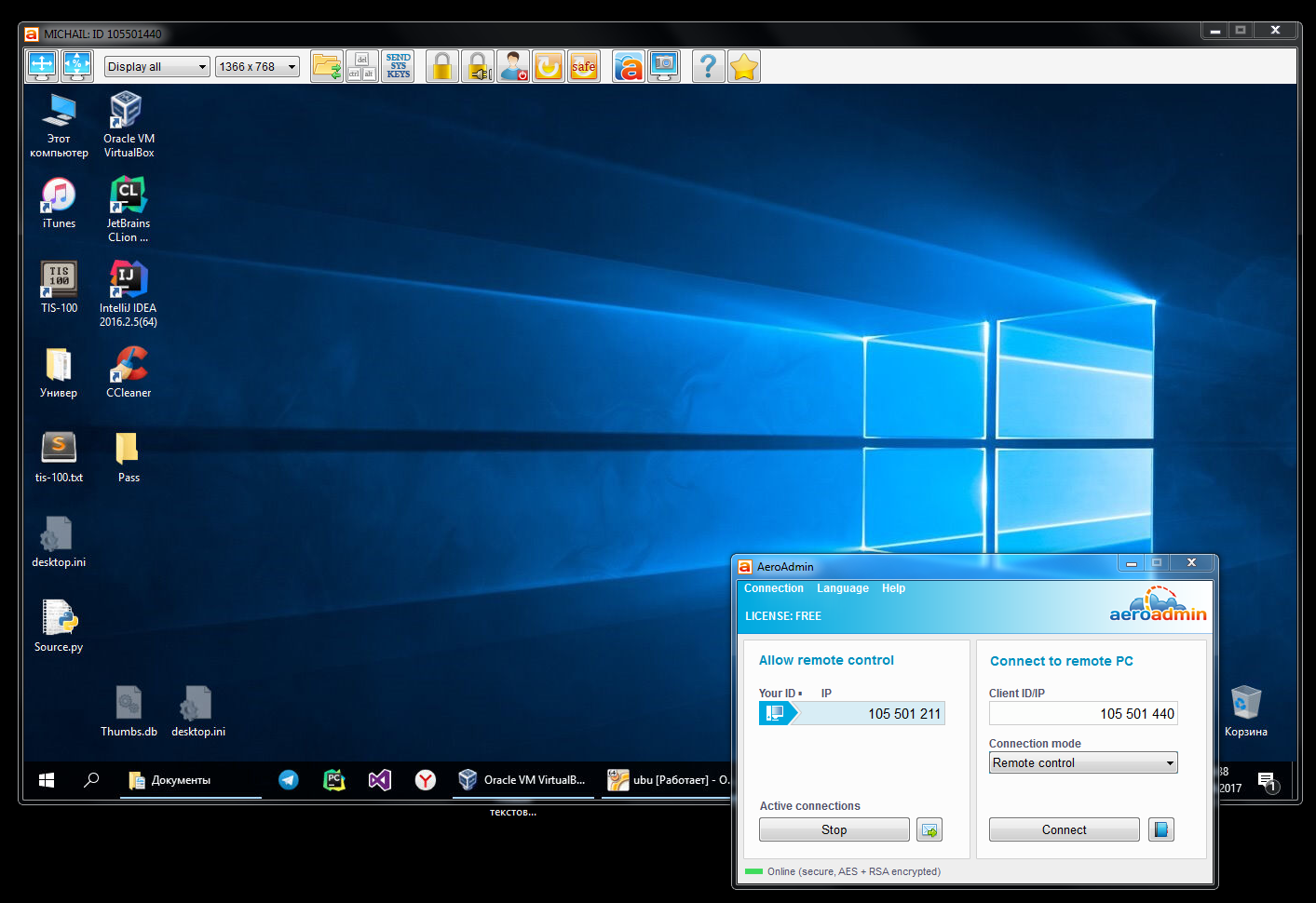


Рисунок 2 – удаленное управление компьютером

Удаленный доступ позволяет перехватывать движения мышью, клавиши клавиатуры, работу с файловой системой. Таким образом предоставляется полный доступ к системе.

# Протоколы удаленного доступа

Все программы удаленного доступа реализуют один из известных протоколов, либо же пишут свои протоколы, которые чаще всего являются коммерческой тайной.

Сетевые протоколы функционируют путем обмена порциями данных, называемыми пакетами. Пакет состоит из управляющей информации, специфичной для протокола, и собственно данных, которые должны быть переданы, их часто называют полезной нагрузкой. До тех пор, пока обмен данными происходит достаточно быстро и без ошибок, нас не волнует, какую управляющую информацию добавляет протокол для своих целей. Но она жизненно важна и должна сохраняться неизменной, если два компьютера собираются обмениваться данными, вне зависимости от среды соединения.

Здесь мы рассмотрим основные протоколы удаленного доступа и принципы их работы.

## 2.1 Протокол RDP

RDP (Remote Desktop Protocol — протокол удалённого рабочего стола) — проприетарный протокол прикладного уровня, позаимствованный Microsoft из купленной у PictureTel телекоммуникационной программы Liveshare Plus, использующийся для обеспечения удалённой работы пользователя с сервером, на котором запущен сервис терминальных подключений. По умолчанию используется порт TCP 3389. Официальное название Майкрософт для клиентского ПО — Remote Desktop Connection или Terminal Services Client (TSC), в частности, клиент в Windows 2k/XP/2003/Vista/2008/7/8/10 называется mstsc.exe.

Возможность использовать 128-битовое шифрование по алгоритмам RC4, AES или 3DES с проверкой целостности хешем MD5 или SHA1, по умолчанию не используется, значение безопасности RDP в Windows 2003 по умолчанию — RC2 (56 бит) c хешем MD5 с откатом на DES (40 бит). (Из-за слабого шифрования RC2 и DES при использовании настройки по умолчанию трафик может быть расшифрован по пути). Также есть поддержка Transport Layer Security (только при использовании шифрования RC4, AES или 3DES);

Из основных особенностей можно выделить:

* Звук с удалённого ПК переадресовывается и воспроизводится на локальном компьютере;
* Позволяет подключать локальные ресурсы к удалённой машине (mapping);
* Позволяет использовать локальный или сетевой принтеры на удалённом ПК;
* Позволяет приложениям, выполняющимся в пределах текущего сеанса, обращаться к локальным последовательным и параллельным портам;
* Можно обмениваться информацией через буфер обмена. [5]

## 2.2 Протокол Telnet

Протокол Telnet (TELNET) предназначен для взаимодействия устройств и процессов терминала. TELNET часто применяется программами эмуляции терминала для входа в удаленную систему. TELNET также обеспечивает взаимодействия терминал-терминал и взаимодействия между процессами. Кроме того, TELNET применяется другими протоколами (например, FTP) для создания управляющего канала протокола.

Главным минусом этого протокола является отсутствие шифрования, что позволяет, практически беспрепятственно, перехватывать незашифрованный трафик (в том числе пароли администратора). [6]

## 2.3 Протокол пересылки файлов FTP.

Протокол пересылки файлов FTP (File Transfer Protocol) реализует удаленный доступ к файлу. Он может использоваться приложениями и пользователями для передачи файлов по сети. Для того, чтобы обеспечить надежную передачу, FTP использует в качестве транспорта протокол с установлением соединений - TCP. Однако, кроме пересылки файлов, протокол FTP предлагает и другие услуги. Так, пользователю предоставляется возможность интерактивной работы с удаленной машиной, например, он может распечатать содержимое ее каталогов. Кроме того, FTP позволяет пользователю указывать тип и формат запоминаемых данных. Наконец, FTP выполняет аутентификацию пользователей.

FTP обеспечивает защиту данных при передаче, отправляя внешнему хосту пароль пользователя и учетного файла пользователя.

## 2.4 Протокол TFTP.

Упрощенный протокол передачи файлов (TFTP) предназначен для обмена файлами с внешними хостами. Для передачи файлов TFTP применяет ненадежный протокол пользовательских дейтаграмм, поэтому обычно он работает быстрее, чем FTP. Как и FTP, TFTP поддерживает передачу файлов в формате NETASCII и в 8-разрядном двоичном формате. В отличие от FTP, TFTP не поддерживает просмотр каталогов или переход в другой каталог внешнего хоста. В нем также не предусмотрена защита с помощью пароля. Кроме того, TFTP работает только с общими каталогами. [7]

## 2.5 Протоколы защищенных каналов SSL, PPTP, IPSec

Протокол PPTP (Point-to-Point Tunneling Protocol) разработан компанией Microsoft совместно с компаниями Ascend Communications, 3Com/Primary Access, ECI-Telematics и US Robotics. Этот протокол был представлен в рабочую группу "PPP Extentions" IETF в качестве претендента на стандартный протокол создания защищенного канала при доступе удаленных пользователей через публичные сети (в первую очередь Internet) к корпоративным сетям. Этот протокол получил статус проекта стандарта Internet, однако, в качестве стандарта так и не был утвержден. Сейчас рабочая группа IETF рассматривает возможность принятия в качестве стандарта протокол L2TP (Layer 2 Tunneling Protocol), который должен объединить лучшие стороны протокола PPTP с протоколом аналогичного назначения L2F (Layer 2 Forwarding), предложенного компанией Cisco.

Протокол PPTP позволяет создавать защищенные каналы для обмена данными по различным сетевым протоколам — IP, IPX или NetBEUI. Данные этих протоколов инкапсулируются с помощью протокола PPTP в пакеты протокола IP, с помощью которого переносятся в зашифрованном виде через любую сеть TCP/IP. Инкапсулируется исходный кадр РРР, поэтому протокол PPTP можно отнести к классу протоколов инкапсуляции канального уровня в сетевой.

Многопротокольность — основное преимущество инкапсулирующих протоколов канального уровня, к которым относится протокол PPTP. Протокол SSL, например, ориентируется на один протокол сетевого уровня — IP. К тому же размещение протокола защищенного канала непосредственно под прикладным уровнем требует переписи приложений, если они хотят воспользоваться возможностями защиты. Защиты данных на канальном уровне делает средства защиты прозрачными как для протоколов прикладного уровня, так и для протоколов сетевого уровня.

Существуют также и варианты встраивания средств создания защищенного канала на сетевом уровне. Имеется несколько протоколов этого типа, использующих шифрацию и инкапсуляцию протокола сетевого уровня в сетевой. Для защиты данных в IP-сетях разработана защищенная версия протокола IP, которую чаще всего называют IPSec. Эта версия поддерживает аутентификацию на сетевом уровне, а также может выполнять шифрацию пользовательских данных. IPSec — это набор стандартов, часть из которых существует в виде проектов, а часть еще находится в стадии разработки. Протокол IPSec не определяет жестко, какие методы шифрации должны использоваться для аутентификации и создания защищенного канала, хотя для первых реализаций определен вариант IPSec, использующий дайджест-функцию MD5 для аутентификации и алгоритм шифрования DES для образования защищенного канала. Недостатком протокола IPSec является то, что он работает только в IP-сетях и не определяет способа защищенной транспортировки пакетов других протоколов. Этот недостаток устраняют такие протоколы, как PPTP или L2F.

PPTP работает путем инкапсуляции "родных" пакетов локальной сети — например, пакетов IPX — внутрь пакетов TCP/IP. Весь пакет IPX, включая его управляющую информацию, становится полезной нагрузкой для пакета TCP/IP, который затем можно передавать по Internet. Программные средства на другом конце линии связи извлекают пакет IPX и направляют его для нормальной обработки в соответствии с его собственным протоколом. Этот процесс называется туннелированием.

Туннелирование позволяет не только экономить на стоимости дальних звонков, но и повышать степень защиты данных. Поскольку туннель соединяет два совместимых протокола, операционная система может выполнять всеобъемлющие проверки надежности защиты, которые она проводит в самой локальной сети. Более того, PPTP позволяет передавать данные, зашифрованные RSA-методами RC-4 или DES. Если безопасность подключения к сети VPN является критическим фактором, администратор сервера может указать, что сервер будет принимать посредством дистанционного соединения только пакеты PPTP, но это предотвратит использование сервера в режиме открытого доступа по HTTP или FTP. Однако, если имеется более одного сервера и, если нужно обеспечить наивысшую защиту, такое решение — вполне приемлемый вариант.

Но даже при всех подобных мерах безопасности единственным специальным программным обеспечением для клиента служит сам протокол PPTP плюс программа связи по модему, которая может соединить с сетью VPN. И даже это не является необходимым, если у поставщика услуг Internet есть средства работы с PPTP, позволяющие безопасно передавать любые данные по стандартному протоколу PPP (Point-to-Point Protocol).

В основе обмена данными по протоколу PPTP лежит управляющее соединение PPTP — последовательность управляющих сообщений, которые устанавливают и обслуживают туннель. Полное соединение PPTP состоит только из одного соединения TCP/IP, которое требует передачи эхо-команд для поддержания его открытым, пока выполняются транзакции. [8]

## 2.6 Усугубление проблем безопасности при удаленном доступе.

Обеспечение безопасности данных при удаленном доступе - проблема если и не номер один, то, по крайней мере, номер два, после проблемы обеспечения приемлемой для пользователей пропускной способности. А при активном использовании сети Internet она становится проблемой номер один.

Неотъемлемым свойством систем удаленного доступа является наличие глобальных связей. По своей природе глобальные связи, простирающиеся на много десятков и тысяч километров, не позволяют воспрепятствовать злонамеренному доступу к передаваемым по этим линиям данным. Нельзя дать никаких гарантий, что в некоторой, недоступной для контроля точке пространства, некто, используя, например, анализатор протокола, не подключится к передающей среде для захвата и последующего декодирования пакетов данных.

Однако использование общественных сетей (речь в основном идет об Internet) еще более усугубляет ситуацию, хотя бы потому, что в такой сети для доступа к конфиденциальным данным в распоряжении злоумышленника имеются более разнообразные и удобные средства, чем выход в чистое поле с анализатором протоколов. Кроме того, огромное число пользователей увеличивает вероятность попыток несанкционированного доступа.

Безопасная система - это система, которая, во-первых, надежно хранит информацию и всегда готова предоставить ее своим пользователям, а во-вторых, система, которая защищает эти данные от несанкционированного доступа.

Межсетевой экран (firewall, брандмауэр) - это устройство, как правило, представляющее собой универсальный компьютер с установленным на нем специальным программным обеспечением, который размещается между защищаемой (внутренней) сетью и внешними сетями, потенциальными источниками опасности. Межсетевой экран контролирует все информационные потоки между внутренней и внешними сетями, пропуская данные, в соответствии с заранее установленными правилами. Межсетевые экраны базируются на двух основных приемах защиты:

* Пакетной фильтрации;
* Сервисах-посредниках (proxy-services).

Эти две функции можно использовать как по отдельности, так и в комбинации.

Пакетная фильтрация. Использование маршрутизаторов в качестве firewall – фильтрация осуществляется на транспортном уровне: все проходящие через межсетевой экран пакеты или кадры данных анализируются, и те из них, которые имеют в определенных полях заданные ("неразрешенные") значения, отбрасываются.

Пропуск во внутреннюю сеть пакетов сетевого уровня или кадров канального уровня по адресам (MAC-адреса, IP-адреса, IPX-адреса) или номерам портов TCP, соответствующих приложениям. Например, для того, чтобы трафик telnet не пересекал границу внутренней сети, межсетевой экран должен отфильтровывать все пакеты, в заголовке TCP которых указан адрес порта процесса-получателя, равный 23 (этот номер зарезервирован за сервисом telnet). Сложнее отслеживать трафик FTP, который работает с большим диапазоном возможных номеров портов, что требует задания более сложных правил фильтрации.

Конечно, для фильтрации пакетов может быть использован и обычный маршрутизатор, в Internet 80% пакетных фильтров работают на базе маршрутизаторов. Однако маршрутизаторы не могут обеспечить ту степень защиты данных, которую гарантируют межсетевые экраны.

Главные преимущества фильтрации межсетевым экраном по сравнению с фильтрацией маршрутизатором состоят в следующем:

Межсетевой экран обладает гораздо более развитыми логическими способностями, поэтому он в отличие от маршрутизатора легко может, например, обнаружить обман по IP-адресу;

У межсетевого экрана большие возможности аудита всех событий, связанных с безопасностью.

Сервисы-посредники (Proxy-services) – сервисы-посредники не допускают возможности непосредственной передачи трафика между внутренней и внешней сетями. Для того, чтобы обратиться к удаленному сервису, клиент-пользователь внутренней сети устанавливает логическое соединение с сервисом-посредником, работающим на межсетевом экране. Сервис-посредник устанавливает отдельное соединение с "настоящим" сервисом, работающим на сервере внешней сети, получает от него ответ и передает по назначению клиенту - пользователю защищенной сети.

Для каждого сервиса необходима специальная программа: сервис-посредник. Обычно, защитный экран включает сервисы-посредники для FTP, HTTP, telnet. Многие защитные экраны имеют средства для создания программ-посредников для других сервисов. Некоторые реализации сервисов-посредников требуют наличия на клиенте специального программного обеспечения. Пример: Sock - широко применяемый набор инструментальных средств для создания программ-посредников.

Сервисы-посредники не только пересылают запросы на услуги, например, разработанный CERN сервис-посредник, работающий по протоколу HTTP, может накапливать данные в кэше межсетевого экрана, так что пользователи внутренней сети могут получать данные с гораздо меньшим временем доступа.

Журналы событий, поддерживаемые сервисами-посредниками, могут помочь предупредить вторжение на основании записей о регулярных неудачных попытках. Еще одним важным свойством сервисов-посредников, положительно сказывающимся на безопасности системы, является то, что при отказе межсетевого экрана защищаемый посредником сервис-оригинал остается недоступным.

Трансляция сетевых адресов – один из видов сервиса-посредника (NAT). Трансляторы адресов заменяют "внешние" IP-адреса серверов своих сетей на "внутренние". При таком подходе топология внутренней сети скрыта от внешних пользователей, вся сеть может быть представлена для них одним-единственным IP-адресом. Такая непрозрачность сети усложняет задачу несанкционированного доступа. Кроме этого, трансляция адресов дает еще одно преимущество - позволяет иметь внутри сети собственную систему адресации, не согласованную с Internet, что снимает проблему дефицита IP-адресов.

Сервисы-посредники намного надежнее фильтров, однако они снижают производительность обмена данными между внутренней и внешней сетями, они также не обладают той степенью прозрачности для приложений и конечных пользователей, которая характерна для фильтров.

Технология защищенного канала призвана обеспечивать безопасность передачи данных по открытой транспортной сети, например, по сети Internet. Защищенный канал включает в себя выполнение трех основных функций:

Взаимная аутентификация абонентов;

Защита передаваемых по каналу сообщений от несанкционированного доступа;

Подтверждение целостности поступающих по каналу сообщений.

Взаимная аутентификация обеих сторон при установлении соединения может быть выполнена, например, путем обмена сертификатами.

Секретность может быть обеспечена каким-либо методом шифрования, например, передаваемые сообщения шифруются с использованием симметричных сессионных ключей, которыми стороны обмениваются при установлении соединения. Сессионные ключи передаются также в зашифрованном виде, при этом они шифруются с помощью открытых ключей. Использование для защиты сообщений симметричных ключей связано с тем, что скорость процессов шифрации и дешифрации на основе симметричного ключа существенно выше, чем при использовании несимметричных ключей.

Целостность передаваемых сообщений достигается за счет того, что к сообщению (еще до его шифрации сессионным ключом) добавляется дайджест, полученный в результате применения односторонней функции к тексту сообщения.

**Протоколы аутентификации удаленных пользователей.**

Контроль доступа пользователей к ресурсам частной сети должен осуществляться в соответствии с политикой безопасности организации, которой принадлежит данная сеть. Эффективное разграничение доступа к сетевым ресурсам может быть обеспечено только при надежной аутентификации пользователей. Требования к надежности аутентификации удаленных пользователей должны быть особенно высокими, так как при взаимодействии с физически удаленными пользователями значительно сложнее обеспечить доступ к сетевым ресурсам. В отличие от локальных пользователей удаленные пользователи не проходят процедуру физического контроля при допуске на территорию контролируемой зоны.

При удаленном взаимодействии важна аутентификация не только пользователей, но и оборудования, поскольку подмена пользователя или маршрутизатора приводит к одним и тем же последствиям — данные из частной сети передаются не тем лицам, которым они предназначены.

Для обеспечения надежной аутентификации удаленных пользователей необходимо выполнение следующих требований:

• проведение аутентификации обеих взаимодействующих сторон — как удаленного пользователя, так и сервера удаленного доступа — для исключения маскировки злоумышленников;

• оперативное согласование используемых протоколов аутентификации;

• осуществление динамической аутентификации взаимодействующих сторон в процессе работы удаленного соединения;

• применение криптозащиты передаваемых секретных паролей либо механизма одноразовых паролей для исключения перехвата и несанкционированного использования аутентифицирующей информации.

Протокол РРР имеет встроенные средства, которые могут быть использованы для организации аутентификации при удаленном взаимодействии. В стандарте RFC 1334 определены два протокола аутентификации:

• по паролю — PAP (Password Authentication Protocol);

• по рукопожатию — CHAP (Challenge Handshake Authentication Protocol).

В процессе установления удаленного соединения каждая из взаимодействующих сторон может предложить для применения один из стандартных протоколов аутентификации — РАР или CHAP.

Иногда компании создают собственные протоколы аутентификации удаленного доступа, работающие вместе с протоколом РРР. Эти фирменные протоколы обычно являются модификациями протоколов РАР и CHAP.

Широкое применение для аутентификации по одноразовым паролям получил протокол S/Key. В программных продуктах, обеспечивающих связь по протоколу РРР, протоколы РАР и CHAP, как правило, поддерживаются в первую очередь. [9]

## 2.7 Протокол РАР.

Суть работы протокола РАР довольно проста. В процессе аутентификации участвуют две стороны — проверяемая и проверяющая. Протокол РАР использует для аутентификации передачу проверяемой стороной идентификатора и пароля в виде открытого текста. Если проверяющая сторона обнаруживает совпадение идентификатора и пароля с записью, имеющейся у него в БД легальных пользователей, то процесс аутентификации считается успешно завершенным, после чего проверяемой стороне посылается соответствующее сообщение. В качестве стороны, чья подлинность проверяется, как правило, выступает удаленный пользователь, а в качестве проверяющей стороны — сервер удаленного доступа.

Для инициализации процесса аутентификации на базе протокола РАР сервер удаленного доступа после установления сеанса связи высылает удаленному компьютеру пакет LCP (Link Control Protocol) — протокол управления каналом, указывающий на необходимость применения протокола РАР. Далее осуществляется обмен пакетами РАР. Удаленный компьютер передает по каналу связи проверяющей стороне идентификатор и пароль, введенные удаленным пользователем. Сервер удаленного доступа по полученному идентификатору пользователя выбирает эталонный пароль из БД системы защиты и сравнивает его с полученным паролем. Если они совпадают, то аутентификация считается успешной, что сообщается удаленному пользователю.

Следует особо отметить, что протокол аутентификации РАР, согласно которому идентификаторы и пароли передаются по линии связи в незашифрованном виде, целесообразно применять только совместно с протоколом, ориентированным на аутентификацию по одноразовым паролям, например, совместно с протоколом S/Key. В противном случае пароль, передаваемый по каналу связи, может быть перехвачен злоумышленником и использован повторно в целях маскировки под санкционированного удаленного пользователя. [10]

## 2.8 Протокол S/Key.

Одним из наиболее распространенных протоколов аутентификации на основе одноразовых паролей является стандартизованный в Интернете протокол S/Key (RFC 1760). Этот протокол реализован во многих системах, требующих проверки подлинности удаленных пользователей, в частности в системе TACACS+ компании Cisco.

Перехват одноразового пароля, передаваемого по сети в процессе аутентификации, не предоставляет злоумышленнику возможности повторно использовать этот пароль, так как при следующей проверке подлинности необходимо предъявлять уже другой пароль. Поэтому схема аутентификации на основе одноразовых паролей, в частности S/Key, позволяет передавать по сети одноразовый пароль в открытом виде и, таким образом, компенсирует основной недостаток протокола аутентификации РАР.

Однако следует отметить, что протокол S/Key не исключает необходимость задания секретного пароля для каждого пользователя. Этот секретный пароль используется только для генерации одноразовых паролей. Для того чтобы злоумышленник не смог по перехваченному одноразовому паролю вычислить секретный исходный пароль, генерация одноразовых паролей выполняется с помощью односторонней, т. е. необратимой, функции. В качестве такой односторонней функции в спецификации протокола S/Key определен алгоритм хэширования MD4 (Message Digest Algorithm 4). Некоторые реализации протокола S/Key в качестве односторонней функции используют алгоритм хэширования MD5 (Message Digest Algorithm 5).

Иногда желательно, чтобы пользователь имел возможность сам назначать секретный постоянный пароль. Для осуществления такой возможности спецификация S/Key предусматривает режим вычисления одноразовых паролей не только на основе секретного пароля, но и на основе генерируемого проверяющей стороной случайного числа. Таким образом, в соответствии с протоколом S/Key за каждым пользователем закрепляется идентификатор и секретный постоянный пароль.

Перед тем как проходить аутентификацию, каждый пользователь должен сначала пройти процедуру инициализации очередного списка одноразовых паролей, т. е. фазу парольной инициализации. Данная фаза выполняется по запросу пользователя на сервере удаленного доступа.

Для ускорения процедуры аутентификации определенное число одноразовых паролей, например, несколько десятков, может быть вычислено заранее и храниться на удаленном компьютере в зашифрованном виде.

Протокол аутентификации на основе одноразовых паролей S/Key применяют, в частности, для улучшения характеристик протоколов централизованного контроля доступа к сети удаленных пользователей TACACS и RADIUS. [11]

# Детектирование сеанса удаленного доступа.

Теоретически способы, позволяющие пользователям обнаружить сеанс удаленного управления, это:

* Сканирование открытых портов позволяет гарантированно детектировать только те средства удаленного управления, которые используют жестко заданные порты. Таким способом можно детектировать Remote Desktop Connection – стандартный удаленный рабочий стол Windows (порт 3389). Radmin, хоть и имеет стандартный порт, используемый по умолчанию (4899), предоставляет возможность настройки порта в диапазоне от 1 до 65535. Таким образом, если используется один из широко известных портов, то обнаружить Radmin будет сложнее. VNC не имеет порта по умолчанию. TeamViewer использует 80 и 443 порты HTTP. Ammyy Admin использует HTTPs прокси, так что обнаружить его сканированием портов невозможно;
* Сканирование реестра. Если программа удаленного управления сохраняет свои настройки в реестре, то её можно обнаружить. Так можно обнаружить, например, стандартный RDP Windows, TeamViewer, Ammyy Admin и Radmin. Но для того чтобы обнаружить средство удаленного управления необходимо знать, что именно и в каких ветках реестра нужно искать. А VNC таким образом не обнаружить, т.к. они не хранят свои настройки в реестре;
* Сканирование сетевого трафика. На первый взгляд подобным способом можно будет детектировать все средства удаленного управления, однако это не так. Дело в том, что многие средства удаленного управления шифруют свой трафик, и обнаружить удаленное управление не удастся. Но детектировать, например, Radmin, RDP Windows всё же возможно, т.к. они используют известные протоколы удаленного управления без шифрования. А TeamViewer и Ammyy Admin, шифруют свой трафик. VNC использует SSL/SSH/VPN, т.е. протоколы удаленного доступа будут скрыты от Вас;
* Сигнатурный анализ. В случае, если сигнатуры программ удаленного управления известны, то обнаружить их исполняемые файлы не составит труда, но если будет использоваться стандартный удаленный стол Windows, то такое сканирование бесполезно, т.к. он присутствует во всех версиях Windows выше NT 4.0.

Самым доступным способом защиты от несанкционированного удаленного доступа являются антивирусы.

Антивирусы, при их относительной сложности, хоть и обнаруживают наличие средств удаленного управления, сам факт установления сеанса удаленного управления они не детектируют.

Есть возможность использования средства клавиатурного мониторинга. Они привлекательны тем, что их назначение - аутентификация оператора ПК, это позволит в случае необходимости доказать, что противоправные действия совершались не оператором ПК, а удаленным пользователем (злоумышленником). [3]

# 4. Практическая часть

В качестве практической части было предложено написать программу, моделирующую процесс перехвата пакетов в виртуальной сети. Программа была написана на языке Python 2.7 в программной среде PyCharm 2017.1.1. Полный код программы представлен в приложении 1, но мы рассмотрим её основные фрагменты и ход работы.

## 4.1 Перехват пакетов

Программа написана с использованием библиотеки Python Pcapy [1], именно с помощью реализованных функций в этой библиотеки мы будет перехватывать трафик, поступающий на компьютер под управлением операционной системы Linux Ubuntu 16.04 LTS.

Краткий алгоритм работы программы:

* Выбираем устройство, которое будем слушать.
* Настраиваем его на прослушивание.
* Далее с помощь метода *pcapy.next()* в цикле *while(1)* будем получать пакеты по одному*.*
* Разбираем пакет на составляющие.
* Смотрим в поля портов и на информацию в данных пакета.
* При наличии подозрительных данных выводим оповещение об этом.

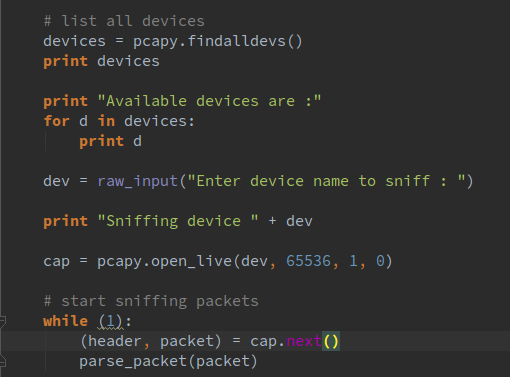


Рисунок 3 – поиск доступных устройств и настройка устройства для снифинга.

На рисунке 3 показан листинг кода, где происходит определение доступных устройств в сети и их настройка при помощи функции *pcapy.open\_live()*, которая принимает аргументы *pcap.open\_live(device, snaplen, promisc, to\_ms)*, где *device* – выбранное устройство, *snaplet* – максимальное число байтов, которое может захватить Pcapy, *promisc –* установка неразборчивого режима устройства, *to\_ms* – время чтения в миллисекундах.

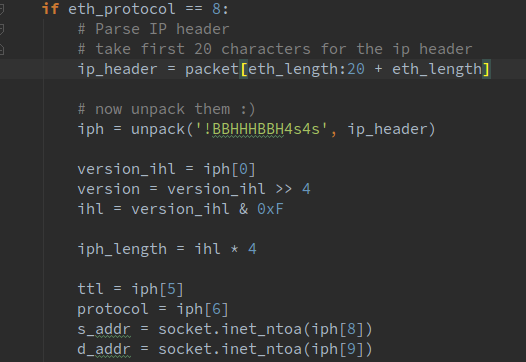


Рисунок 4 – разбор заголовка IP пакета

На рисунке 4 показан листинг кода для разбора IP-пакета, для последующего анализа его данных. Разбор пакета происходит, опираясь на модель построения IP пакета [2].

Далее происходит разбор протоколов транспортного уровня, где можно выделить TCP, UPD, ICMP протоколы и другие.

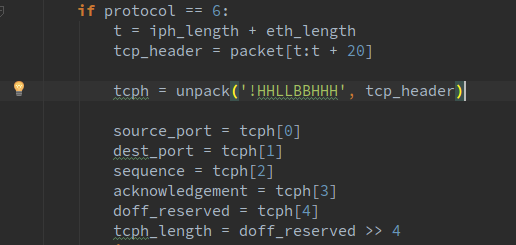


Рисунок 5 – разбор TCP-протокола

## 4.2. Анализ наличия попытки удаленного доступа.

Проверять наличие удаленного доступа будем, прослушивая популярные порты, которые используются программами удаленного доступа, а также анализом данных, находящихся в пакете.

1. Запустим программу из терминала командой: *python sniffer.py -c.*
2. На удаленном компьютере запустим программу Teamviewer и попробуем подключиться к локальной машине по сети интернет.

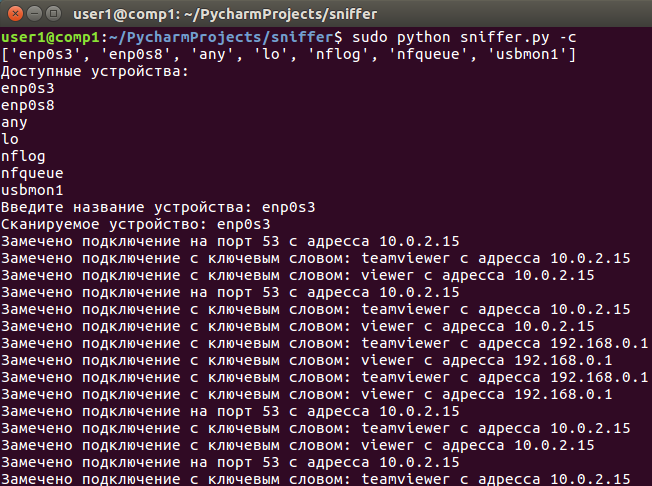


Рисунок 6 – реакция программы на попытку подключения через программу Teamviewer.

На рисунке 6 видно, что программа детектирует попытку подключения через программу Teamviewer, как по порту 53 (стандартный порт Teamviewer), так и по данным в составе пакета.

1. Запустим программу из терминала командой: *python sniffer.py -c.*
2. На удаленном компьютере запустим программу Ammyy Admin и попробуем подключиться к локальной машине по сети интернет

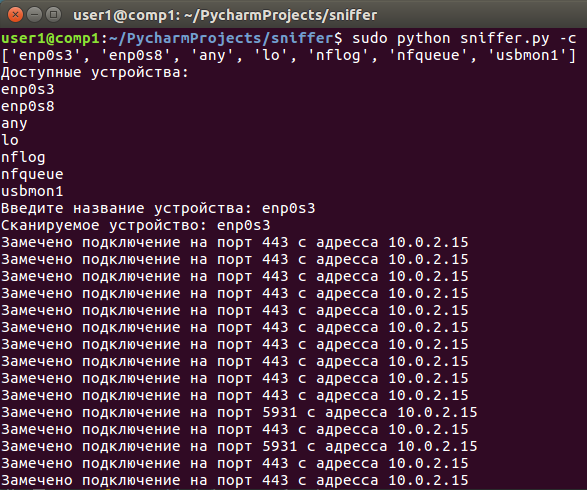


Рисунок 7 – реакция программы на попытку подключения через программу Ammyy Admin

На рисунке 7 видно, что программа распознает активность на порт 443 (стандартный порт Ammyy Admin). Но извлечь информацию из данных пакета не удается, так как в программе Ammyy Admin весь трафик шифруется и получить данные не удастся.

Теперь проверим работу на сервере. Запустим программу в локальной сети из 3 компьтеров. На компьютере под управлением Linux 16.04 запустим сам скрипт и на двух других (под управлением Windows XP) установим удаленное соединение через стандартный протокол RDP.

1. Запустим программу из терминала командой python sniffer.py -c -n

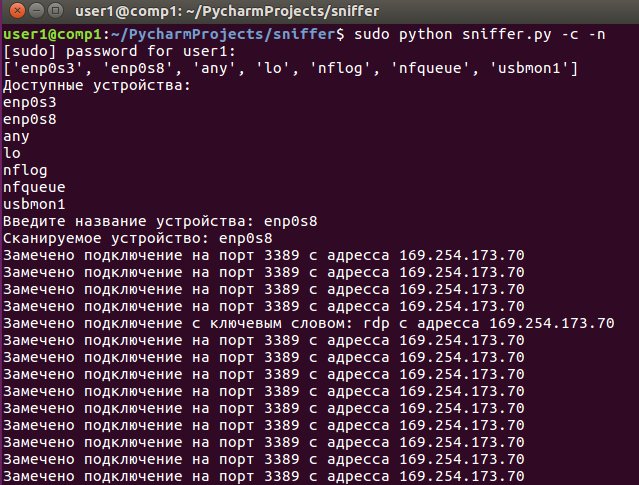


Рисунок 8 – реакция сервера на попытку установления соединения в локальной сети

На рисунке 8 видно, что программа распознает активность на двух компьютерах в локальной сети на порту 3389 и по ключевому слову «rdp».

# ЗАКЛЮЧЕНИЕ

Перехват пакетов является достаточно простой задачей, в то время как анализ заголовков пакетов является достаточно серьезной задачей. Распознавание протоколов удаленного доступа через перехват пакетов позволяет сделать некие суждения о несанкционированном доступе к системе, но не дает полной гарантии. Так как трафик многих программ шифруется, а порты, через которые можно получить доступ к системе могут постоянно меняться. Поэтому анализ пакетов не является достаточным условием для обеспечения безопасности системы. Для приемлемой защиты необходимо также использовать принцип минимизации привилегий, а также использовать сканирование реестра на наличие следов программ удаленного доступа.

# СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Pcapy 0.11.1 Python pcap extension [Электронный ресурс]. URL: https://pypi.python.org/pypi/pcapy (Дата обращения 10.04.2017).

2. Формат IP-пакета [Электронный ресурс]. URL: http://iptcp.net/format-ip-paketa.html (Дата обращения 4.06.2017).

3. Детектирование удаленного доступа [Электронный ресурс]. URL: http://www.frodex.ru/article/radkl2015 (Дата обращения 1.05.2017).

4. Многообразие видов и схем удаленного доступа [Электронный ресурс]. URL: http://citforum.ru/nets/spsmp/spsmpred\_10.shtml (Дата обращения 5.06.2017).

5. Remote Desktop Protocol – Википедия [Электронный ресурс]. URL: https://ru.wikipedia.org/w/index.php?oldid=83162997 (Дата обращения 5.06.2017).

6. Telnet – Википедия [Электронный ресурс]. URL: https://ru.wikipedia.org/w/index.php?oldid=86470453 (Дата обращения 5.06.2017).

7. FTP – Википедия [Электронный ресурс]. URL: https://ru.wikipedia.org/w/index.php?oldid=86291145 (Дата обращения 5.06.2017).

8. Протоколы защищенных каналов [Электронный ресурс]. URL: http://kunegin.com/ref5/ipsec/ (Дата обращения 5.06.2017).

9. Усугубление проблем безопасности при удаленном доступе [Электронный ресурс]. URL: http://5fan.ru/wievjob.php?id=44562 (Дата обращения 6.06.2017).

10. PAP – Википедия [Электронный ресурс]. URL: https://ru.wikipedia.org/w/index.php?oldid=76529884 (Дата обращения 6.06.2017).

11. S/Key – Википедия [Электронный ресурс]. URL: https://ru.wikipedia.org/w/index.php?oldid=83040262 (Дата обращения 6.06.2017).

# ПРИЛОЖЕНИЕ 1

# coding=utf-8

import getopt

import socket

from struct import \*

import datetime

import pcapy

import sys

from impacket.ImpactDecoder import EthDecoder, LinuxSLLDecoder

CATCH\_MODE = False

PROMISC\_MODE = False

def main(argv):

try:

cmd\_opts = "pchn" # packets, catch, help, promisc

opts, args = getopt.getopt(argv[1:], cmd\_opts)

for opt in opts:

global CATCH\_MODE

global PROMISC\_MODE

if opt[0] == '-p':

CATCH\_MODE = True

if opt[0] == '-c':

CATCH\_MODE = False

if opt[0] == '-n':

PROMISC\_MODE = True

if opt[0] == '-h':

print "-p - перехват пакетов \n" \

"-c - распознавание протоколов RPD \n" \

"-n - неразборчивый режим"

except getopt.GetoptError:

pass

# list all devices

devices = pcapy.findalldevs()

print devices

print "Доступные устройства:"

for d in devices:

print d

dev = raw\_input("Введите название устройства: ")

print "Сканируемое устройство: " + dev

cap = pcapy.open\_live(dev, 65536 \* 8, PROMISC\_MODE, 0)

# start sniffing packets

while (1):

(header, packet) = cap.next()

parse\_packet(packet)

# Convert a string of 6 characters of ethernet address into a dash separated hex string

def eth\_addr(a):

b = "%.2x:%.2x:%.2x:%.2x:%.2x:%.2x" % (ord(a[0]), ord(a[1]), ord(a[2]), ord(a[3]), ord(a[4]), ord(a[5]))

return b

# function to parse a packet

def parse\_packet(packet):

# init:

# telnet 23, STD RDP 3389, Radmin 4899, Teamviewer 80 443 53, ammyy 443 1255 5931

bad\_ports = [23, 3389, 4899, 80, 443, 53, 1255, 5931]

bad\_words\_data = ['teamviewer', 'rdp', 'RDP', 'viewer', 'TEAMVIEWER', 'radmin',

'ammyyadmin', 'ammyy', 'telnet']

# parse ethernet header

eth\_length = 14

eth\_header = packet[:eth\_length]

eth = unpack('!6s6sH', eth\_header)

# print 'UNPACKING RAW ETH\_HEADER: ' + str(eth) # unpacking

eth\_protocol = socket.ntohs(eth[2])

# print 'Destination MAC : ' + eth\_addr(packet[0:6]) + ' Source MAC : ' + eth\_addr(

# packet[6:12]) + ' Protocol : ' + str(eth\_protocol)

# Parse IP packets, IP Protocol number = 8

if eth\_protocol == 8:

# Parse IP header

# take first 20 characters for the ip header

ip\_header = packet[eth\_length:20 + eth\_length]

# now unpack them :)

iph = unpack('!BBHHHBBH4s4s', ip\_header)

version\_ihl = iph[0]

version = version\_ihl >> 4

ihl = version\_ihl & 0xF

iph\_length = ihl \* 4

ttl = iph[5]

protocol = iph[6]

s\_addr = socket.inet\_ntoa(iph[8])

d\_addr = socket.inet\_ntoa(iph[9])

if CATCH\_MODE:

print 'Версия : ' + str(version) + ' Длинна IP заголовка : ' + str(ihl) + \

' TTL : ' + str(ttl) + ' Протокол : ' + str(protocol) + ' Адресс отправения : ' + \

str(s\_addr) + ' Адресс доставки : ' + str(d\_addr)

# TCP protocol

if protocol == 6:

t = iph\_length + eth\_length

tcp\_header = packet[t:t + 20]

tcph = unpack('!HHLLBBHHH', tcp\_header)

source\_port = tcph[0]

dest\_port = tcph[1]

sequence = tcph[2]

acknowledgement = tcph[3]

doff\_reserved = tcph[4]

tcph\_length = doff\_reserved >> 4

if CATCH\_MODE:

print 'Протокол: TCP ' + 'Исходный порт : ' + str(source\_port) + ' Порт назначения : ' + str(

dest\_port) + ' Порядковый номер : ' + str(

sequence) + ' Подтверждение : ' + str(acknowledgement) + ' Длина TCP заголовка : ' + str(

tcph\_length)

else:

if dest\_port in bad\_ports:

print 'Замечено подключение на порт ' + str(dest\_port) + ' с адресса ' + str(s\_addr)

h\_size = eth\_length + iph\_length + tcph\_length \* 4

data\_size = len(packet) - h\_size

# get data from the packet

data = packet[h\_size:]

decode\_data = EthDecoder().decode(data).get\_data\_as\_string()

if CATCH\_MODE:

print decode\_data

else:

for item in bad\_words\_data:

if item in decode\_data:

print 'Замечено подключение с ключевым словом: ' + item + ' с адресса ' + str(s\_addr)

# ICMP Packets

elif protocol == 1:

u = iph\_length + eth\_length

icmph\_length = 4

icmp\_header = packet[u:u + 4]

# now unpack them :)

icmph = unpack('!BBH', icmp\_header)

icmp\_type = icmph[0]

code = icmph[1]

checksum = icmph[2]

print 'Протокол: ICMP ' + 'Тип : ' + str(icmp\_type) + \

' Код : ' + str(code) + ' Checksum : ' + str(checksum)

h\_size = eth\_length + iph\_length + icmph\_length

data\_size = len(packet) - h\_size

# get data from the packet

data = packet[h\_size:]

print 'Данные пакета : ' + data

# UDP packets

elif protocol == 17:

u = iph\_length + eth\_length

udph\_length = 8

udp\_header = packet[u:u + 8]

# now unpack them :)

udph = unpack('!HHHH', udp\_header)

source\_port = udph[0]

dest\_port = udph[1]

length = udph[2]

checksum = udph[3]

if CATCH\_MODE:

print 'Протокол: UDP ' + 'Исходный порт : ' + \

str(source\_port) + ' Порт назначения : ' + str(dest\_port) + ' Длинна : ' + str(

length) + ' Checksum : ' + str(checksum)

else:

if dest\_port in bad\_ports:

print 'Замечено подключение на порт ' + str(dest\_port) + ' с адресса ' + str(s\_addr)

h\_size = eth\_length + iph\_length + udph\_length

data\_size = len(packet) - h\_size

data = packet[h\_size:]

if CATCH\_MODE:

print 'Данные пакета : ' + data

else:

for item in bad\_words\_data:

if item in data:

print 'Замечено подключение с ключевым словом: ' + item + ' с адресса ' + str(s\_addr)

if \_\_name\_\_ == "\_\_main\_\_":

main(sys.argv)