**BC/NW 2013, № 2 (23):11.2)**

**РАЗРАБОТКА И ТЕСТИРОВАНИЕ МЕТОДИКИ ОЦЕНКИ НЕСАНКЦИОНИРОВАННЫХ ВТОРЖЕНИЙ**

Л.И. Абросимов

(ФГБОУ ВПО "Национальный исследовательский университет "МЭИ")

Межсетевой экран или сетевой экран — комплекс аппаратных или программных средств, осуществляющий контроль и фильтрацию проходящих через него сетевых пакетов в соответствии с заданными правилами.[1]

Основной задачей межсетевого экрана является защита компьютерных сетей или отдельных узлов от несанкционированного доступа. Также межсетевые экраны часто называют фильтрами, так как их основная задача — не пропускать (фильтровать) пакеты, не подходящие под критерии, определённые в конфигурации.

Межсетевые экраны бывают двух видов – аппаратно-реализуемые (аппаратный межсетевой экран) и программные.

Аппаратный межсетевой экран (АМЭ) представляет собой устройство, которое подключается между ЭВМ (или локальной вычислительной сетью) и точкой доступа в сеть Интернет или любую другую сеть. Его достоинство заключается в том, что, являясь отдельным устройством, оно не потребляет ресурсы ЭВМ, обеспечивает возможность защищенной работы без изменения настроек на любой ЭВМ, содержит встроенную аппаратную защиту от вирусов, служит для предотвращения и защиты от атак и выполняет функцию фильтрации спама. Недостатком АМЭ, как правило, оказывается сравнительно высокая цена для рядовых пользователей, и, как следствие, основной областью применения аппаратно-реализуемых межсетевых экранов становятся локальные вычислительные сети различных компаний.

Первые межсетевые экраны появились еще в 1980-х и представляли собой обычные маршрутизаторы, включающие фильтрацию пакетов (пересылаемые данные проверялись по заданному набору правил и не пропускались далее в случае несоответствия). В 1990-х годах появились так называемые межсетевые экраны цепного уровня. Далее по сложности и новизне - "защитники" программного уровня. Позже в основу межсетевых экранов легла динамическая фильтрация пакетов. А самая новая на сегодня архитектура программ типа межсетевой экран – фильтрация на уровне ядра операционной системы (эта архитектура имеет как программные, так и аппаратные реализации). В основном, каждое новое поколение основывается на принципе работы предыдущего. То есть то, что справедливо для первого поколения, может быть применимо для второго поколения, правда с некоторыми поправками и дополнениями.

Некоторые межсетевые экраны также позволяют осуществлять трансляцию адресов — динамическую замену внутрисетевых (серых) адресов или портов на внешние, используемые за пределами ЛВС.

Межсетевые экраны подразделяются на различные типы в зависимости от следующих характеристик:

- обеспечивает ли экран соединение между одним узлом и сетью или между двумя или более различными сетями;

- на уровне каких сетевых протоколов происходит контроль потока данных;

- отслеживаются ли состояния активных соединений или нет.

В зависимости от охвата контролируемых потоков данных АМЭ делятся на:

- традиционный сетевой (или межсетевой) экран— программа (или неотъемлемая часть операционной системы) на шлюзе (сервере, передающем трафик между сетями) или аппаратное решение, контролирующие входящие и исходящие потоки данных между подключенными сетями.

- персональный сетевой экран — программа, установленная на пользовательском компьютере и предназначенная для защиты от несанкционированного доступа только этого компьютера.

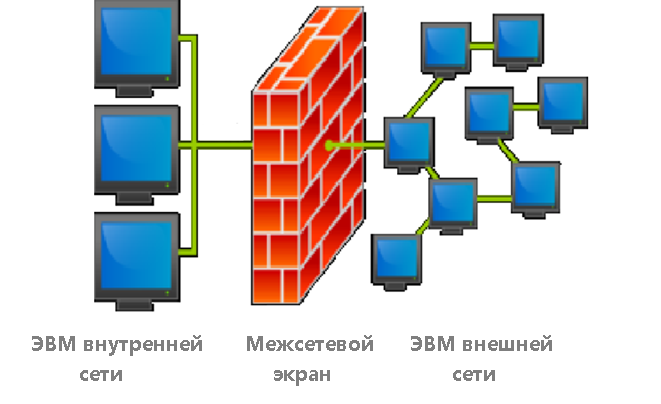


Рис.1. Пример расположения межсетевого экрана

Вырожденный случай — использование традиционного сетевого экрана сервером, для ограничения доступа к собственным ресурсам.

Один из возможных вариантов расположения межсетевого экрана представлен на рис. 1.

Известные результаты тестирования АМЭ

1) Английской компанией VNU Business Publications, проводилось тестирование АМЭ в 2004 г. [2], по результатам которого оценивалось управление, функциональность устройства, его исполнение и производительность.

Тестирование проводилось на семи устройствах: Cisco PIX 525, WatchGuard FireBox 4500, Symantec VelociRaptor 1100, CheckPoint, Nokia IP740, Webscreen 100 и SonicWall Pro 300. С точки зрения информационной безопасности тестирование велось на устойчивость к атакам типа DDoS, применялась атака «ping of death», использующая уязвимость в операционных системах Windows 95, Windows NT и в старых версиях \*nix и mac. Результаты данного тестирования, в силу его давности, уже нельзя рассматривать как актуальные, подавляющее большинство систем было исправлено, поэтому эта уязвимость ныне является исторической.

2) Редакцией журнала «Xakep» проводилось тестирование АМЭ в марте 2008 г. [3]

Аппаратные межсетевые экраны были представлены двумя устройствами: D-Link DI-604 и TRENDnet TW100-BRF114. Применялись тесты Jumper, DNS Tester и CPILSuite. Подробнее о применяемых тестах.

Jumper. Вместо прямого изменения процесса в памяти компьютера Jumper заставляет целевой процесс загрузить чужую библиотеку DLL самостоятельно. Для этого он прописывает в раздел реестра 'AppInit\_DLLs' новое значение и затем убивает процесс explorer.exe, который автоматически перезагружает Windows. Таким образом, explorer.exe автоматически загружает библиотеку DLL Jumper’а.

DNS Tester. По умолчанию на OS NT, начиная с Windows 2000, служба Windows DNS Client принимает обращения по всем запросам DNS и управляет ими. Это приложение может использоваться, чтобы передать данные на отдаленный компьютер, обрабатывая специальный запрос DNS без уведомления об этом. DNStester использует рекурсивный запрос DNS (то есть обращение службы самой к себе).

CPILSuite – набор из трех тестов от компании Comodo, которая сама производит межсетевые экраны.

Таким образом, из трёх тестов, два, скорее, относятся к тестированию антивирусных систем, и один является продуктом компании, которая может быть заинтересована в определённых результатах тестирования, поэтому в целом тестирование нельзя считать объективным.

На сегодняшний день на рынке представлено порядка 10 компаний, занимающихся разработкой аппаратно-реализуемых межсетевых экранов. Несмотря на то, что аппаратно-реализуемые межсетевые экраны заняли свое место на рынке в качестве важного компонента стратегии безопасности, существующие методы оценки качества устройств данного класса весьма примитивны. Вследствие этого приходится полагаться на субъективные оценки, как то: не сертифицированные тесты, маркетинговые ходы и т.п. По этой причине представляются актуальными формулирование подхода к решению задачи экспериментальной оценки качества информационной защиты, обеспечиваемой аппаратным межсетевым экраном, и разработка алгоритма и программного средства для их тестирования. Таким образом, станет возможным выявление реакции устройств рассматриваемого класса на действие определенных типов атак.

Постановка задачи

В настоящей статье рассматривается разработка методики и программы для тестирования межсетевого экрана на возможность предотвращения угроз, создаваемых сообщениями, содержащими аномалии в заголовках сетевых и транспортных протоколов.

В первую очередь, для решения поставленной задачи необходимо выделить перечень сообщений с аномалиями в заголовках сетевых и транспортных протоколов, для этого рассматриваются возможные угрозы сетевой безопасности и спецификации сетевых и транспортных протоколов.

Далее необходимо составить алгоритм тестирования аппаратно-реализуемого межсетевого экрана, алгоритм должен позволять:

протестировать межсетевой экран на большом количестве угроз;

легко моделировать угрозы при помощи разрабатываемого программного средства;

сделать процесс тестирования безопасным, то есть не подвергать компьютеры стенда действию вредоносных пакетов;

использовать для проведения тестирования малое количество оборудования.

На основе этого алгоритма разрабатывается программное средство, осуществляющее тестирование аппаратно-реализуемого межсетевого экрана.

Далее требуется разработать алгоритм управления аппаратно-реализуемым межсетевым экраном, следуя которому производится настройка экрана. Для взаимодействия программного средства с пользователем создается графический интерфейс пользователя.

Для проведения тестирования потребуется составить план испытаний на испытательном стенде, предназначенном для генерации эмулированных угроз, отслеживания перемещения вредоносных пакетов и выявления реакции межсетевого экрана на них.

Однако вследствие того, что разработка выполняется для АМЭ, больший интерес представляет классификация угроз. Классифицировать их можно двумя способами. Классификация, представленная в табл. 1, основывается на способах фильтрации угроз.

Таблица 1

Классификация угроз по способам фильтрации угроз

|  |  |  |
| --- | --- | --- |
| Способы фильтрации | Используются | Категории угроз |
| По  сигнатурам | фиксированные шаблоны  регулярные выражения | известные эксплойты (exploits)  вирусы (virus)  программы-шпионы (spyware)  Peer-to-peer приложения (Р2Р)  неразрешенные Instant Messenger приложения (IM) |
| По аномалиям протоколов | соответствие RFC  декодирование протоколов  нормализация протоколов | зондирование (reconnaissance)  угроза безопасности (security policy)  захват пропускной способности (network equipment)  трафик с аномальным поведением (traffic normalization)  неизвестные эксплойты (exploits) |
| По  уязвимостям | декодирование протоколов  регулярные выражения  разбор сообщений приложений | угроза безопасности (security policy)  неизвестные эксплойты (exploits) |

В настоящем исследовании рассматриваются атаки «отказ в обслуживании».

Атаки «отказ в обслуживании» (DoS – Denial of Service) – это разновидность сетевых атак на вычислительную систему, цель которых – довести систему до отказа, то есть создать такие условия, при которых легитимные (правомерные) пользователи системы не могут получить доступ к предоставляемым системой ресурсам, либо этот доступ затруднён. DoS-атаки блокируют сетевые сервисы лавинными соединениями; повреждением серверов или программ, выполняющихся на серверах; полной загрузкой серверных ресурсов или иными способами, препятствующими доступу легитимных пользователей к сетевым сервисам.

В общем случае DoS-атаки можно разделить на:

атаки, основанные на посылке одиночных пакетов, которые выводят из строя серверы (далее именно они будут подробно рассмотрены);

атаки, основанные на скоординированной массовой посылке пакетов, полностью загружающей ресурсы сервера и/или каналы связи.

В арсенале злоумышленников имеется семь следующих способов совершения DoS- и DDoS-атак [19]:

использование уязвимостей;

привлечение «зомби»;

инструментальные средства создания атак;

атаки на производительность;

SYN-флуд;

лавинное установление соединения;

флуд «соединений в секунду».

Концепция методики

Методика – это систематизированная совокупность шагов, действий, кото­рые необходимо предпринять, чтобы решить определенную задачу или достичь определенной цели.

Рассматривая поставленную задачу разработки методики для тестирования аппаратных систем предотвращения вторжений, выделим две обязательных составляющих ее решения.

Одна из составляющих – это необходимость хранения множества вредоносных пакетов (или правил их генерации), образующих трафик, результат прохождения которого сквозь систему предотвращения вторжений и определит ее способность защищать приложения, операционные системы и инфраструктуру.

Второй составляющей является программное обеспечение, представляющее собой генератор вредоносного трафика в частности, и средство управления тестирующей системой в целом.

Таким образом, разрабатываемая методика должна представлять собой совокупность библиотек систематизированного множества форматов пакетов, имитирующих вредоносный и нежелательный трафик, и программного обеспечения, необходимого для генерации трафика и проведения исследований и тестирования.

Употребляя термин «библиотека», следует сразу оговориться: в номенклатуре вычислительных средств речь, безусловно, идет о базе данных, то есть особым образом организованной структуре для хранения информации.

Рассмотрим несколько вариантов реализации хранения. Идея одного из них заключается в том, что бы создать для каждой угрозы свой исполняемый файл. Запуск этих исполняемых файлов будет приводить к генерации требуемых угроз. При кажущейся простоте использования данного способа он имеет существенные недостатки: отсутствие систематизации, универсальности (для каждой угрозы необходимо создавать свою отдельную программу), гибкости (плохая модифицируемость базы угроз); сложность контролирования большого количества программ и их запуска.

Избежать этих недостатков позволяет другой вариант реализации хранения моделируемых угроз. Он заключается в генерации трафика, представляющего собой требуемые угрозы, на основании хранящихся в базе данных значений всех полей пакетов. Однако и у этого варианта есть один недостаток: вследствие того, что на каждый пакет приходится большой объем данных (значительная часть которых одинакова для многих пакетов), база данных будет занимать немалый объем дискового пространства.

Наиболее целесообразным является хранение данных для генерации вредоносного трафика, а не уже созданные угрозы.

Таким образом, для каждой угрозы устанавливаются ее характерные особенности (например, определенные значения определенных полей пакета; минимально необходимое для создания атаки количество пакетов и т.д.).

Выделенные особенности формализуются и агрегируются, на основании чего становится возможным определить для каждой угрозы характеризующий ее вектор параметров.

Вектор параметров является правилом генерации моделируемой угрозы. В связи с тем, что в базе данных будут храниться лишь значения полей пакетов, «превращающие» обыкновенный пакет во вредоносный, необходимо для каждого протокола создать пакет, поля которого не будут нести в себе вредоносных значений. Называть такие пакеты будем «шаблонными».

Таким образом, для формирования пакетов моделируемых угроз вредоносные значения, хранящиеся в базе данных, будут замещать значения полей шаблонного пакета.

Совокупность всех параметров является универсальной для угроз, основывающихся на уязвимостях и/или аномалиях протоколов 3-го (IP, ICMP, IGMP) и 4-го (TCP, UDP) уровней модели OSI. Таким образом, при появлении новых угроз рассматриваемых типов для включения их в библиотеку необходимо лишь установить их характерные особенности, то есть конкретные значения описанных параметров.

Разработка базы данных угроз

Основными этапами проектирования базы данных являются [15]:

выбор модели и средств управления базой данных;

сбор и анализ входных данных;

создание логической модели базы данных;

создание физической модели базы данных.

Назначение проектируемой базы данных – хранение моделируемых угроз.

Создание логической модели базы данных

Продолжением функциональной декомпозиции процесса проектирования реляционной базы данных является создание ее логической модели, результатом которого будет нормализованная схема отношений базы данных.

Протокол IP (Internet Protocol) предназначен для использования в соединенных между собой компьютерных сетях обмена данными на основе коммутации пакетов. Протокол обеспечивает передачу блоков данных, называемых дейтаграммами между отправителем и получателем, узлы которых идентифицируются адресами фиксированной длины. Протокол также обеспечивает фрагментацию и сборку для дейтаграмм большого размера, если сеть не позволяет передать дейтаграмму целиком [17].

Угрозы моделируются по разным протоколам (IP, ICMP, IGMP, OSPF, TCP, UDP), однако согласно механизму инкапсуляции стека протоколов TCP/IP всех их объединяет наличие IP-заголовка [18]. Поэтому необходимо создать отношение, задающее векторы значений полей IP-заголовков для каждой моделируемой угрозы. Формат заголовка IP-пакета представлен на рис. 2.



Рис. 2. Заголовок IP-пакета

Для однозначной идентификации угроз во всех отношениях базы данных необходимо включить атрибут, устанавливающий соответствие между именем угрозы и персональным номером, ее идентифицирующим. Таким образом, в дальнейшем во всех отношениях каждая угроза будет представлена своим уникальным номером.

В случае угроз, основанных только на IP-пакетах, для их генерации требуются дополнительные параметры, то есть параметры, не связанные с IP-заголовком. Таким образом, атрибутами данного отношения будут являться (атрибуты, сформированные из полей IP-заголовка обозначены как «IP: имя поля»):

id угрозы;

id категории угрозы;

наименование угрозы;

IP: протокол (указывает, какому протоколу верхнего уровня принадлежит пакет);

IP: длина IP-заголовка (IHL);

IP: длина IP-пакета;

IP: идентификатор (используется для распознавания пакетов, образовавшихся путем фрагментации исходного пакета);

IP: флаг MF (показывает, не является ли этот пакет последним в цепочке пакетов);

IP: смещение фрагмента (указывает смещение поля данных данного пакета от начала общего поля данных исходного пакета, подвергнутого фрагментации);

IP: время жизни пакета (TTL);

IP: контрольная сумма заголовка;

IP: IP-адрес отправителя;

IP: IP-адрес получателя;

IP: параметры;

реальная длина IP-пакета;

требуемое количество пакетов (для посылки).

Время от времени шлюзу или узлу назначения необходимо связаться с узлом-источником, например, чтобы уведомить об ошибке при обработке дейтаграммы. Для этих целей используется ICMP-протокол (Internet Control Message Protocol). ICMP использует базовый сервис протокола IP так, будто является протоколом вышестоящего уровня, в действительности же ICMP является составной частью IP. Первый октет поля данных дейтаграммы является полем, указывающим на тип ICMP, – значение этого поля определяет формат остальной части дейтаграммы [19]. На рис. 3. приведен формат ICMP-пакета в общем виде.



Рис. 3. Формат ICMP-пакета в общем виде

Наиболее часто используемый тип ICMP-сообщений – «echo request» и ассоциированные с ними «echo reply» пакеты, также известные как «ping». Формат этих пакетов представлен на рис. 4.

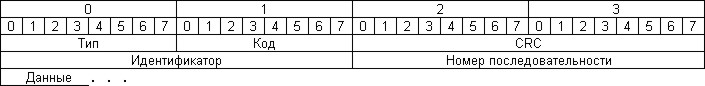


Рис. 4. Формат ICMP-пакета с типом echo request или echo reply

Для осуществления возможности одновременной выборки, то есть одним запросом, данных из двух таблиц необходимо включить атрибут, связывающий эти таблицы. Таким образом, каждая угроза, основывающаяся на значениях полей ICMP-пакета, будет иметь два идентификатора: «сквозной», являющийся составляющей домена атрибута «id угрозы», и icmp-идентификатор, определенный исключительно для отношения ICMP.

Правила для генерации угроз, основанных на значениях полей ICMP-пакета, описаны в отношении со следующими атрибутами:

id\_icmp угрозы;

id угрозы;

ICMP: тип;

ICMP: код;

ICMP: данные;

ICMP.EchoRequest: идентификатор;

ICMP: длина заголовка.

Протокол TCP (Transmission Control Protocol) предназначен для организации надежного обмена данными между процессами в разнородных сетевых средах. Процесс передает данные, вызывая функции TCP с передачей им буферов данных в качестве аргументов. TCP упаковывает данные из таких буферов в сегменты и вызывает модуль IP для передачи каждого сегмента адресату TCP. На приемной стороне TCP помещает данные из сегмента в приемный пользовательский буфер и уведомляет заинтересованного пользователя. TCP включает в сегменты управляющую информацию, которая служит для обеспечения гарантированной доставки с сохранением порядка сегментов [20]. Формат заголовка TCP представлен на рис. 5.



Рис. 5. Формат заголовка TCP

Схема отношения, описывающего значения полей TCP-заголовков для генерации моделируемых угроз, представляет собой:

id\_tcp угрозы (аналогично по сути атрибуту «id\_icmp угрозы» в отношении ICMP);

id угрозы;

TCP: порт назначения;

TCP: смещение данных;

TCP: флаг URG (указывает, что поле «Указатель важности» значимо);

TCP: флаг ACK (указывает, что поле «Номер подтверждения» значимо);

TCP: флаг PSH (инструктирует получателя протолкнуть данные, накопившиеся в приемном буфере, в приложение пользователя);

TCP: флаг RST (инструктирует оборвать соединение, очистить буфер);

TCP: флаг SYN (синхронизация номеров последовательности);

TCP: флаг FIN (флаг, будучи установлен, указывает на завершение соединения);

реальная длина TCP-сегмента.

Последним отношением, необходимым для целостности базы данных, является отношение, задающее векторы значений полей UDP-сегментов.

Протокол передачи пользовательских дейтаграмм UDP (User Datagram Protocol) предназначен для поддержки режима обмена дейтаграммами на основе коммутации пакетов в среде связанных между собой компьютерных сетей. Этот протокол предполагает использование IP в качестве протокола низлежащего уровня.

Протокол UDP обеспечивает прикладным программам процедуры для передачи сообщений другим приложениям с минимальным сервисом. Протокол ориентирован на транзакции, не гарантирует доставки сообщений и не предотвращает появление дубликатов [21]. Формат заголовка UDP представлен на рис. 6.

|  |  |  |
| --- | --- | --- |
| 0 | Порт источника | Порт назначения |
| 32 | Длина пакета | CRC |
| 64 | Данные | |

Рис.6. Формат заголовка UDP

Правила для генерации угроз, основанных на значениях полей UDP-сегмента, описаны в отношении со следующими атрибутами:

id\_udp угрозы (аналогично атрибутам «id\_icmp угрозы» и «id\_tcp угрозы» в отношениях ICMP и TCP соответственно);

id угрозы;

UDP: порт источника;

UDP: порт назначения;

UDP: длина пакета.

Собрав воедино все спроектированные и описанные ранее отношения, а также создав с помощью механизма первичных и внешних ключей связи между ними, получим концептуальную модель данных, представленную на рис. 7



Рис. 7. Концептуальная модель данных

Алгоритм работы с базой данных представлен на рис. 8.

Рис. 8. Схема алгоритма работы с базой данных

Создание физической модели базы данных

Задача создания физической модели реляционной базы данных заключается в преобразовании логической модели в последовательность команд SQL для создания объектов реляционной базы данных. Другими словами, отношения логической модели отображаются в таблицы и индексы реляционной базы данных.

Основным пунктом в решении этой задачи является создание базовых таблиц, представляющих собой основные блоки хранения данных. Таблицы выводятся из сущностей логической модели данных. Из рис. 7 видно, что сущностей в разработанной модели 4, таким образом, и таблиц в базе данных также должно быть 4.

Рассмотрим каждую из таблиц более подробно.

Сущность «IP» будет преобразована в одноименную таблицу. Колонки таблицы выводятся из атрибутов сущности логической модели данных.

Далее необходимо определить типы данных для каждой колонки. В табл. 2 приведено соответствие имен атрибутов сущности именам колонок и их типы данных.

Таблица 2

Соответствие имен атрибутов сущности именам колонок и их типы данных в таблице «IP»

|  |  |  |
| --- | --- | --- |
| Имя атрибута сущности | Имя колонки | Тип данных |
| id угрозы | mal\_id | tinyint unsigned |
| наименование угрозы | mal\_name | varchar |
| id категории угрозы | categ\_id | tinyint unsigned |
| протокол | protocol | smallint unsigned |
| IP-адрес отправителя | source\_addr | varchar |
| IP-адрес получателя | dest\_addr | varchar |
| длина IP-заголовка | ihl | tinyint unsigned |
| длина IP-пакета | pack\_len | smallint unsigned |
| реальная длина пакета | act\_len | smallint unsigned |
| TTL | ttl | smallint unsigned |
| идентификатор | id | smallint unsigned |
| флаг MF | mf | bool |
| смещение фрагмента | offset | tinyint unsigned |
| контрольная сумма | crc | bool |
| параметры | options | tinyint unsigned |
| требуемое количество пакетов | pack\_amount | smallint unsigned |

Сущность «ICMP» будет преобразована в таблицу с таким же именем. В табл. 3 приведено соответствие имен атрибутов сущности именам колонок и их типы данных.

Таблица 3

Соответствие имен атрибутов сущности именам колонок и их типы данных в таблице «ICMP»

|  |  |  |
| --- | --- | --- |
| Имя атрибута сущности | Имя колонки | Тип данных |
| id\_icmp угрозы | mal\_id\_icmp | tinyint unsigned |
| id угрозы | mal\_id | tinyint unsigned |
| тип | type | tinyint unsigned |
| код | code | tinyint unsigned |
| данные | data | varchar |
| echoreq.идентификатор | echoreq\_id | smallint unsigned |
| длина заголовка | header\_len | tinyint unsigned |

Сущность «TCP» будет преобразована в одноименную таблицу. В табл. 4 приведено соответствие имен атрибутов сущности именам колонок и их типы данных.

Таблица 4.

Соответствие имен атрибутов сущности именам колонок и их типы данных в таблице «TCP»

|  |  |  |
| --- | --- | --- |
| Имя атрибута сущности | Имя колонки | Тип данных |
| id\_tcp угрозы | mal\_id\_tcp | tinyint unsigned |
| id угрозы | mal\_id | tinyint unsigned |
| порт назначения | dest\_port | smallint unsigned |
| флаг URG | urg | tinyint unsigned |
| флаг ACK | ack | tinyint unsigned |
| флаг PSH | psh | tinyint unsigned |
| флаг RST | rst | tinyint unsigned |
| флаг SYN | syn | tinyint unsigned |
| флаг FIN | fin | tinyint unsigned |
| смещение данных | offset | smallint unsigned |
| реальная длина сегмента | act\_length | smallint unsigned |

Сущность «UDP», как и все предыдущие, будет преобразована в одноименную таблицу. В табл. 5 приведено соответствие имен атрибутов сущности именам колонок и их типы данных.

Таблица 5

Соответствие имен атрибутов сущности именам колонок и их типы данных в таблице «UDP»

|  |  |  |
| --- | --- | --- |
| Имя атрибута сущности | Имя колонки | Тип данных |
| id\_udp угрозы | mal\_id\_udp | tinyint unsigned |
| id угрозы | mal\_id | tinyint unsigned |
| порт источника | source\_port | smallint unsigned |
| порт назначения | dest\_port | smallint unsigned |
| длина пакета | header\_length | tinyint unsigned |

Тестирование методики

На основе предложенного механизма тестирования АМЭ разработано программное средство, осуществляющее тестирование АМЭ.

Для проведения тестирования потребуется составить план испытаний на испытательном стенде, предназначенном для генерации эмулированных угроз, отслеживания перемещения вредоносных пакетов и выявления реакции межсетевого экрана на них.

План проведения испытаний содержит следующие пункты:

Тестирование разработанного программного средства и проверка корректности его работы;

конфигурирование тестового стенда, представляющего физическую конфигурацию фрагментов сети и настройку аппаратно-реализуемого межсетевого экрана Cisco 2611XM;

тестирование межсетевого экрана, то есть эмуляция угроз выбранных категорий и фиксирование реакций на угрозы с целью проверки работоспособности алгоритма и тестируемого межсетевого экрана;

оценка результатов тестирования.

Чтобы убедиться, что разработанное программное средство выполняет свои задачи и делает это корректно, требуется произвести его тестирование. Проверяться будет поток исходящего сетевого трафика на машине с запущенным программным средством. Для таких задач используется программа-анализатор сетевого трафика, наиболее подходящим вариантом такой программы является Wireshark. Функциональность, которую предоставляет Wireshark, аналогична возможностям программы tcpdump, однако Wireshark имеет графический пользовательский интерфейс и гораздо больше возможностей по сортировке и фильтрации информации. Программа позволяет пользователю просматривать весь проходящий по сети трафик в режиме реального времени, переводя сетевую карту в неразборчивый режим. Программа распространяется под свободной лицензией GNU GPL и использует для формирования графического интерфейса кроссплатформенную библиотеку GTK+. Wireshark — это приложение, которое «знает» структуру самых различных сетевых протоколов, и поэтому позволяет разобрать сетевой пакет, отображая значение каждого поля протокола любого уровня. На рис. 9 представлено окно программы Wireshark во время тестирования программного средства. Тестирование произвелось путем отметки для генерации всех 13 угроз. В результате как показано на рис. 9.на сетевом интерфейсе были обнаружены корректно сгенерированными все 13 угроз.

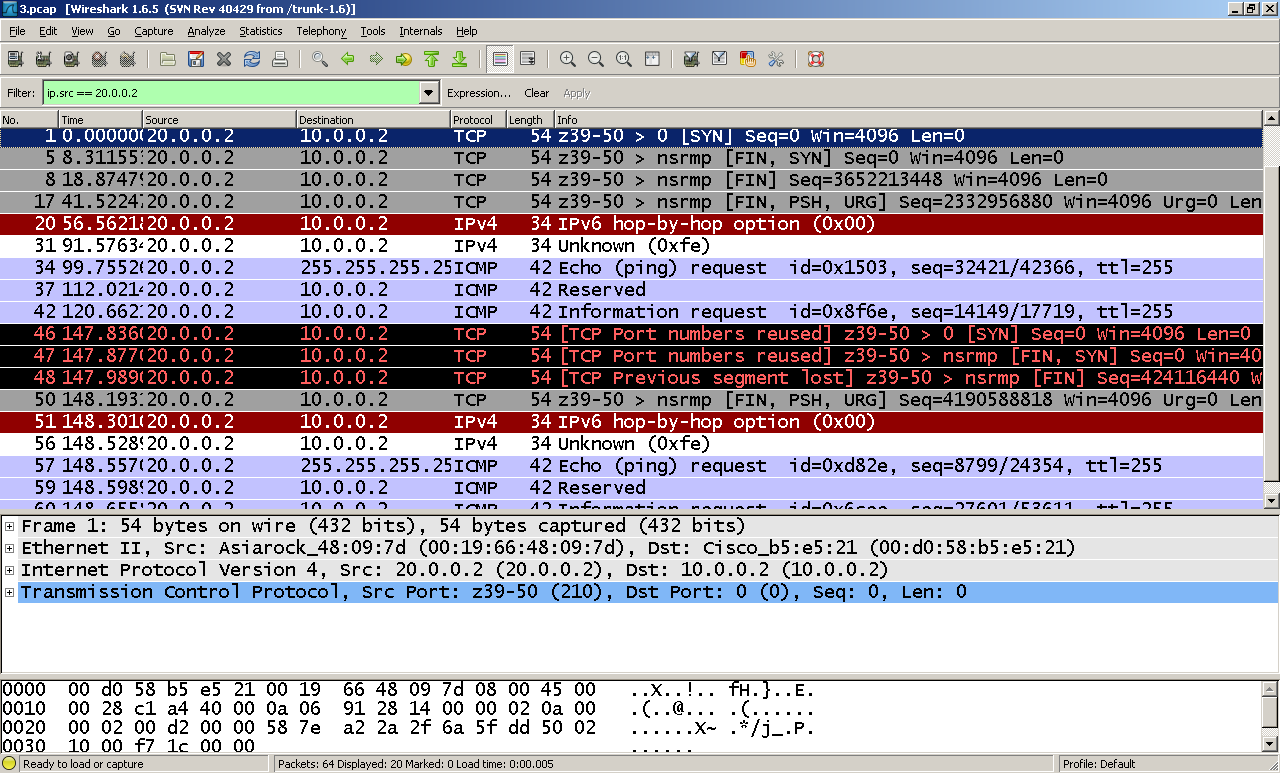


Рис. 9– Окно программы Wireshark

Окно программы Wireshark (рис. 9)состоит из трех областей, в первой отображаются все захваченные с сетевого интерфейса кадры, область поделена на столбцы, в которых указаны детали кадра: «No» - номер кадра, «Time» - время с начала захвата, «Source» - источник, «Destination» - получатель, «Protocol» - протокол верхнего уровня, «Length» - длина кадра в байтах, «Info» - информация содержащаяся в кадре. Во второй области содержится детальная информация кадра, разобранная по протоколам, в третье области кадр в шестнадцатиричном формате.

Тестирование межсетевого проводится на специально созданном стенде, который представляет собой совокупность аппаратно-реализуемого межсетевого экрана и двух фрагментов сетей. ЭВМ из одного фрагмента сети c запущенным на ней программным средством осуществляет тестирование аппаратно-реализованного межсетевого экрана, подключенным к порту Ethernet 0/1, вторая ЭВМ выполняет функции атакуемого объекта, подключенная к порту Ethernet 0/0.

Схема стенда для тестирования представлена на рис10

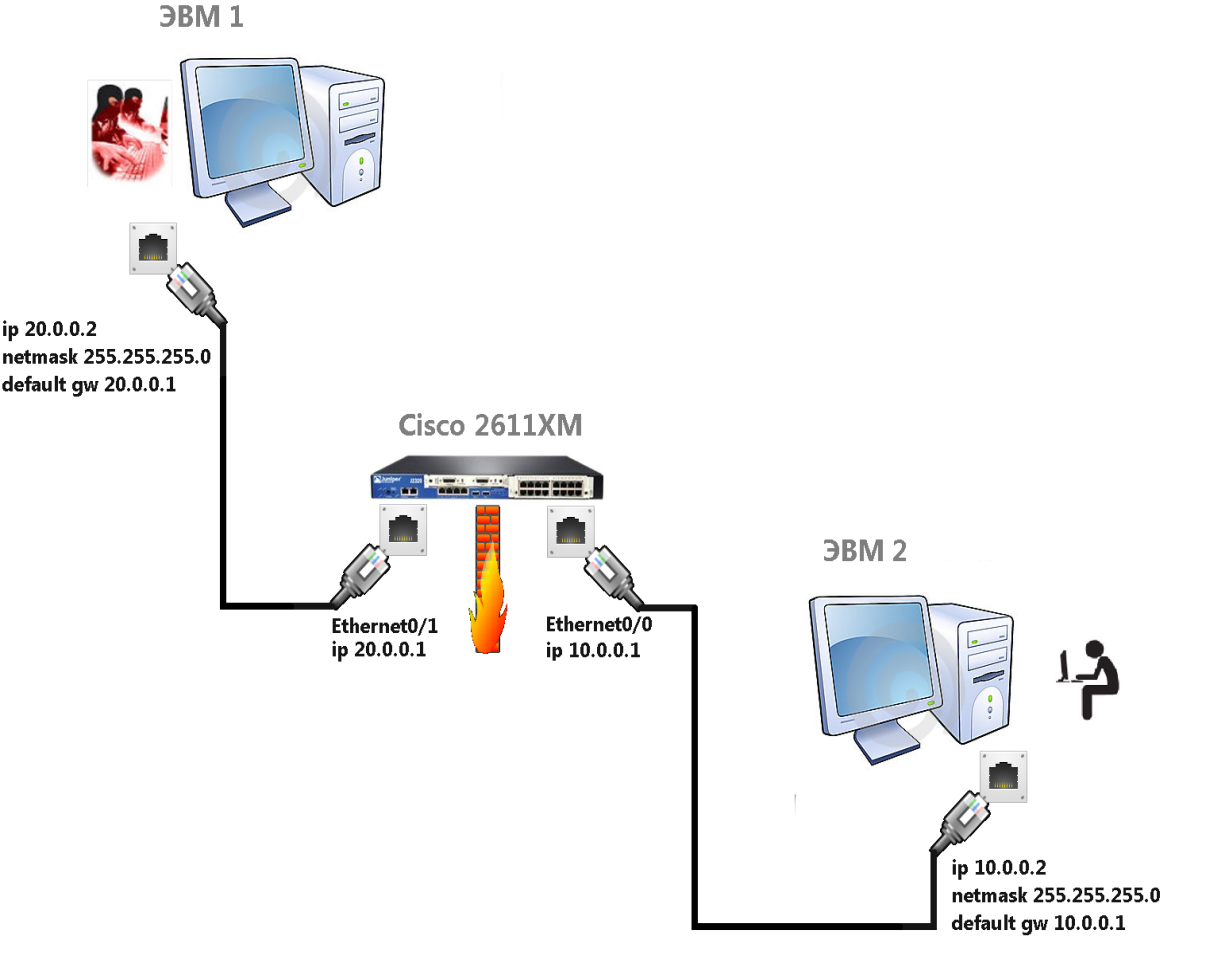


Рис. 11 Схема фрагментов сетей для тестирования

Cisco 2600XM работает в режиме моста, соединяющего две сети. Сеть 10.0.0.0 с маской сети 255.255.255.0 выступает как внешняя которая генерирует угрозы , и сеть 20.0.0.0 с маской сети 255.255.255.0 является защищаемой . Между сетями действует протокол маршрутизации RIP-1, запущен механизм контроля сессий (CBAC) и созданы списки доступа (access-list).

У ЭВМ из первой сети назначен ip-адрес 20.0.0.2, и в качестве шлюза по умолчанию значится ip-адрес интерфейса Ethernet 0/1 20.0.0.1.

Вторая ЭВМ рассматривается как представитель защищаемой сети имеет ip-адрес 10.0.0.2, и в качестве шлюза по умолчанию значится ip-адрес интерфейса Ethernet 0/0 10.0.0.1.

Задачей тестирования является выявление реакции аппаратно-реализуемого межсетевого экрана на выбранные угрозы и проверка работоспособности спроектированной системы. Тестирование заключается в посылке генератором атак атакуемому объекту 13-ти сообщений по каждой из 13-х угроз: угрозы эмулируются программным обеспечением автоматически при выборе категории, к которой они относятся. Категории выбираются в произвольном порядке.

Ожидаемые результаты:

работоспособность разработанной системы будет доказана в случае конъюнкции следующих исходов тестирования:

на сетевой карте атакующего объекта будут зафиксированы все сгенерированные сообщения;

сгенерированные сообщения будут корректны, то есть значения полей сгенерированных пакетов, отслеженных на атакующем объекте, будут идентичны ожидаемым.

Корректность работы аппаратно-реализуемого межсетевого экрана подтверждается непоступлением на сетевую карту атакуемого объекта ни одного сгенерированного пакета.

Трафик на атакующем и атакуемом объектах отслеживался с помощью программного обеспечения – анализатора трафика Wireshark 1.6.5.

Результаты тестирования на возможность предотвращения угроз типа «Зондирование сети»

1. Целью исследуемого режима является реакция межсетевого экрана на атаку «IP: Сокращенное время жизни (1) » – «IP: Short Time to Live (1) ». Для шаблонной структуры IP-пакета меняется параметр «Время жизни» (TTL), его значение устанавливается в 1. На сетевой карте атакующей ЭВМ фиксируется соответствующий ip-пакет рис. 12. На сетевой карте атакуемой машины нет фиксации этого пакета. Вывод: межсетевой экран распознал этот вид угрозы.

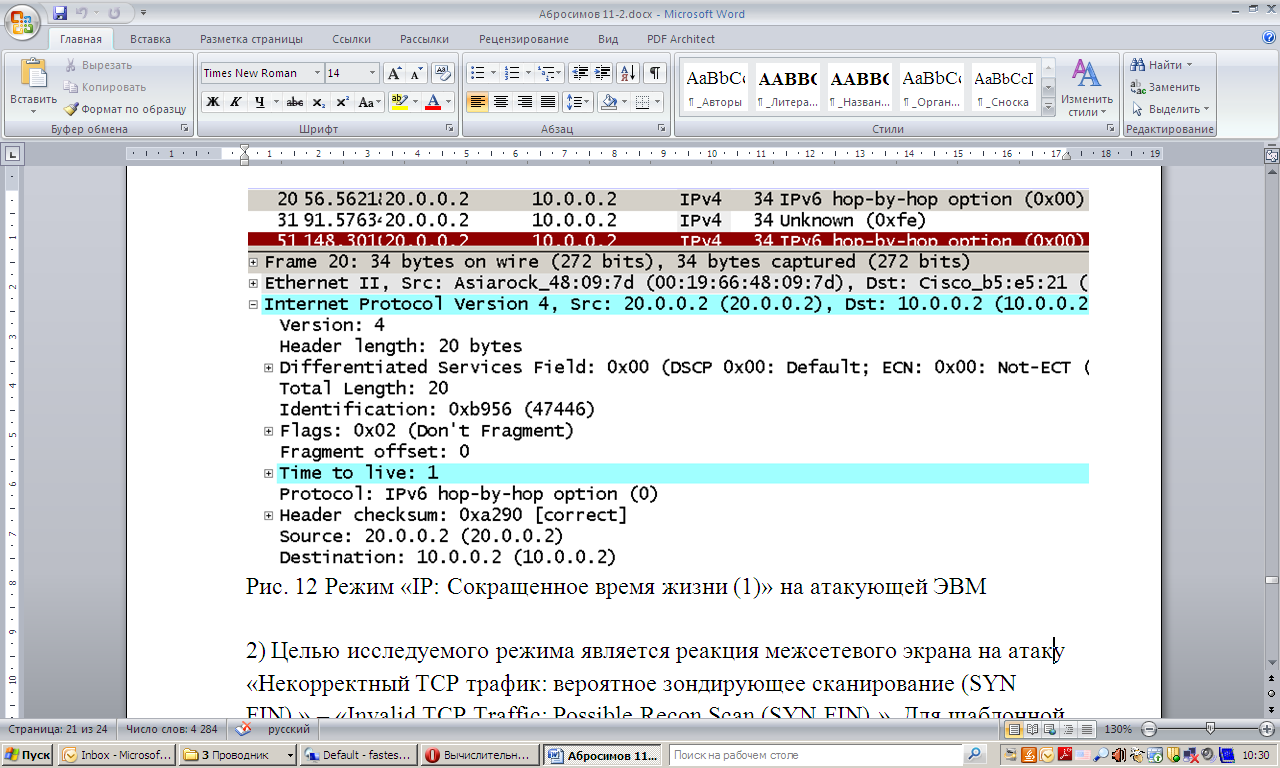


Рис. 12 Режим «IP: Сокращенное время жизни (1)» на атакующей ЭВМ

2) Целью исследуемого режима является реакция межсетевого экрана на атаку «Некорректный TCP трафик: вероятное зондирующее сканирование (SYN FIN) » – «Invalid TCP Traffic: Possible Recon Scan (SYN FIN) ». Для шаблонной структуры TCP-пакета меняются параметры «установлен флаг SYN» и «установлен флаг FIN». На сетевой карте атакующей ЭВМ фиксируется соответствующий ip-пакет рис. 13. На сетевой карте атакуемой машины так же есть фиксация этого пакета рис. 14. Вывод: межсетевой экран не распознал этот вид угрозы.

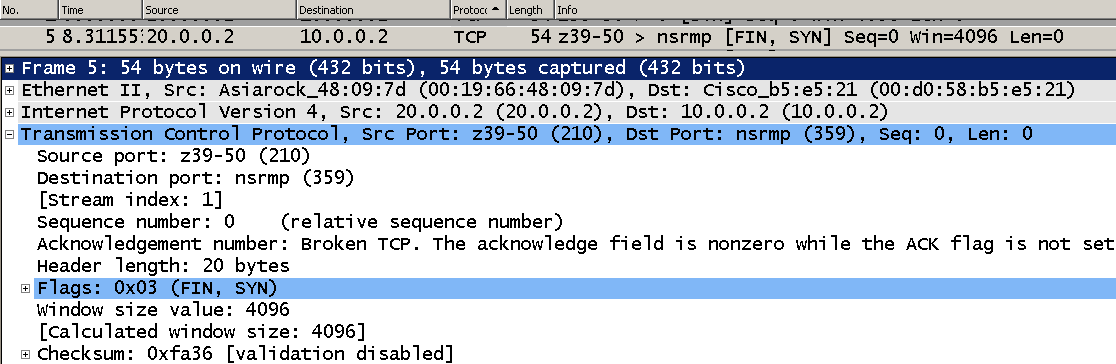


Рис. 13 Режим «Некорректный TCP трафик: вероятное зондирующее сканирование (SYN FIN) » на атакующей ЭВМ

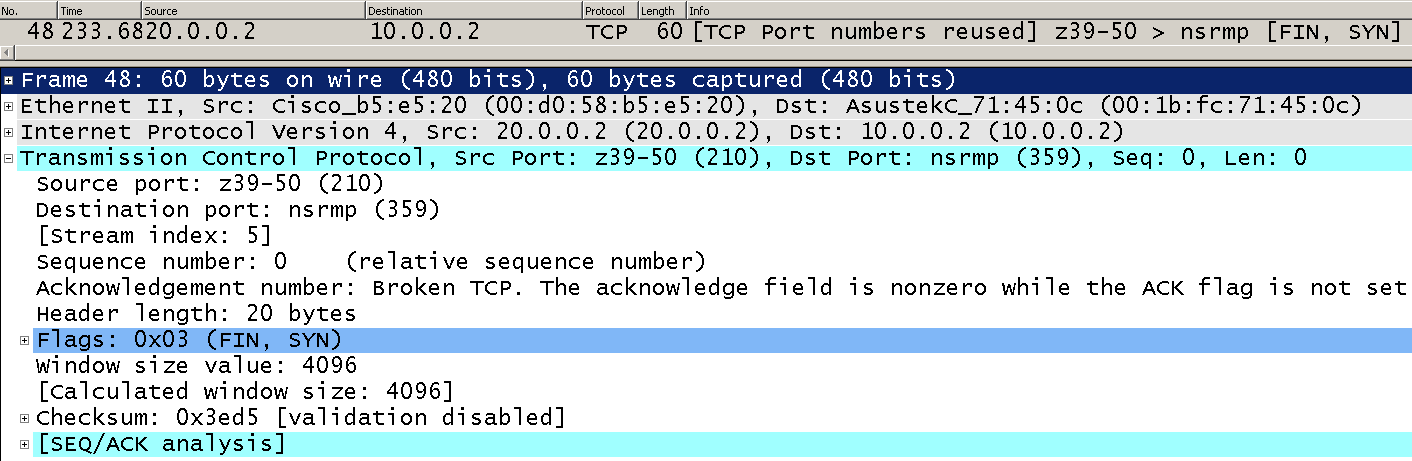


Рис. 14 Режим «Некорректный TCP трафик: вероятное зондирующее сканирование (SYN FIN) » на атакуемой ЭВМ

Некоторые оценки результатов тестирования межсетевого экрана

Результаты тестирования аппаратно-реализуемого межсетевого экрана приведены в табл. 3.

Таблица 3

| Угрозы | Зафиксировано  на атакующей  ЭВМ | Зафиксировано  на атакуемой  ЭВМ |
| --- | --- | --- |
| 1. IP: Short Time To Live (1) | 10 | 0 |
| 2. Invalid TCP Traffic: Possible Recon  Scan (SYN FIN) | 10 | 10 |
| 3. Invalid TCP Traffic: Possible nmap  Scan (FIN no ACK) | 10 | 10 |
| 4. Invalid TCP Traffic: Possible nmap  Scan (No Flags) | 10 | 10 |
| 5. Invalid TCP Traffic: Possible nmap  Scan (XMAS (FIN PSH URG)) | 10 | 10 |
| 6. Invalid TCP Traffic: Destination Port 0 | 10 | 10 |
| 7. ICMP: Broadcast Ping Request | 10 | 0 |
| 8. ICMP: Information Request (Type 15) | 10 | 10 |
| 9. ICMP: Unassigned Type (Type 1) | 10 | 10 |
| 10. IP: Source IP Address Spoofed  (Impossible Packet) | 10 | 10 |
| 11. IP: Source IP Address Spoofed  (Loopback) | 10 | 10 |
| 12. IP: Source IP Address Spoofed  (Multicast) | 10 | 10 |
| 13. Invalid IP Traffic: Unknown IP Protocol | 10 | 10 |

Столбец «Зафиксировано на атакующей ЭВМ» отображает количество пакетов со встроенной угрозой, отправленных в сеть, а столбец «Зафиксировано на атакуемой ЭВМ» - количество пакетов со встроенной угрозой, преодолевших аппаратно-реализуемый межсетевой экран и доставленных на защищаемый объект.

Анализируя данные, представленные в табл. 3, можно сделать следующие выводы:

Разработанное программное средство для тестирования аппаратно-реализуемого межсетевого экрана работоспособно и выполняет поставленные перед ним задачи;

Аппаратно-реализуемый межсетевой экран Cisco 2600XM распознал 15% эмулированных угроз;

Литература

Wikipedia.org. <http://ru.wikipedia.org/>

Статья «Групповой тест аппаратных брандмауэров.» <http://firebox.ru/ibc_2.asp>

Статья «Софтостены VS. Хардмауэры: с чем безопаснее?» <http://www.xakep.ru/post/42593/>

Лаборатория Касперского. Сравнительное тестирование систем предотвращения вторжений, октябрь 2006. <http://www.kaspersky.ru/downloads/word/idscomparison.doc>

3Com Corporation. 3Com X Family Local Security Manager User’s Guide, version 2.5.2 // TECHD-176. – July 2007. – 333 p.

Вихорев С.В. Классификация угроз информационной безопасности // годовой обзор «Сетевые атаки и системы информационной безопасности». – 2001. – 11 с.

Open Source Vulnerability Database [OSVDB]: Vulnerability Standards. <http://osvdb.org/vuln_standards>

Cohen F. Computer Viruses: Theory and Experiments // Computers and Security – Elsevier Advanced Technology Publications, Oxford, UK. – February 1987. – Vol. 6, Issue 1. P. 22-35.

SearchSecurity.com – Информационный ресурс по безопасности для IT-профессионалов. <http://searchsecurity.techtarget.com/sDefinition/0,,sid14_gci214518,00.html>

RootkitAnalytics [RkA]. <http://www.rootkitanalytics.com/>

2Spyware.com – проект, посвященный борьбе со шпионским и вредоносным программным обеспечением. <http://www.2-spyware.com/backdoors-removal>

Web Application Security Consortium [WASC] – международная группа экспертов, занимающаяся разработкой, утверждением и пропагандой стандартов безопасности Интернет приложений. <http://www.webappsec.org/projects/threat/>

RFC 791 – Internet Protocol. <http://www.ietf.org/rfc/rfc791.txt>

Таненбаум Э. Компьютерные сети. 4-е изд. – СПб.: Питер, 2006. – 992 с.: ил.

RFC 792 – Internet Control Message Protocol. <http://www.ietf.org/rfc/rfc792.txt>

RFC 793 – Transmission Control Protocol. <http://www.ietf.org/rfc/rfc793.txt>

RFC 768 – User Datagram Protocol. <http://www.ietf.org/rfc/rfc793.txt>

Qt – A cross-platform application and UI framework. <http://www.qtsoftware.com/products>

Фленов М. Сетевая библиотека Winsock // Xakep. – 2006. – № 41.

Microsoft TechNet – Resources for IT Proffesionals. <http://technet.microsoft.com/en-us/library/cc723706.aspx>