УДК 004.7:004.428

ОСОБЕННОСТИ ПРОГРАММНОЙ РЕАЛИЗАЦИИ АЛГОРИТМОВ АНАЛИЗА СЕТЕВОГО ТРАФИКА

В. А. Баранцева, В. В. Галушка

Донской государственный технический университет, Ростов-на-Дону, Российская Федерация
vika.barantseva@mail.ru
galushkavv@yandex.ru

Рассматриваются методы и средства захвата транзитного сетевого трафика с использованием стандартных средств операционной системы для работы с сетевыми интерфейсами — сокетами и с помощью специализированных библиотек — Pcap и SharpPcap, способы декапсуляции пакетов с использованием библиотеки Packet.NET.

Ключевые слова: анализ траффика, сокеты, SharpPcap, Packet .NET, декапсуляция.

UDC 004.7:004.428

SOFTWARE IMPLEMENTATION FEATURES OF NETWORK TRAFFIC ANALYSIS ALGORITHMS

V. A. Barantseva, V. V. Galushka

Don State Technical University
Rostov-on-Don, Russian Federation

vika.barantseva@mail.ru
galushkavv@yandex.ru

The article examines the methods and means of the transit network traffic capture using standard operating system tools for network interfaces — sockets and specialized libraries — Pcap and SharpPcap, ways of packages decapsulation using the Packet.NET library.

Keywords: traffic analysis, sockets, SharpPcap, Packet.NET, decapsulation.

Введение. Под анализом сетевого трафика понимают совокупность технологий, позволяющих проводить накопление, обработку, классификацию, контроль и модификацию сетевых пакетов в зависимости от их содержания в реальном времени [1]. Анализ траффика включает два важных этапа: его перехват и извлечение полезной информации из полученных данных.

Перехват — это процесс сбора всех пакетов, проходящих через определенный сетевой интерфейс, программный доступ к которому в большинстве операционных систем осуществляется с использованием программного интерфейса сокетов.

Сокет — это объект операционной системы, через который можно передавать и принимать данные от процессов независимо от того, выполняются они на одном или разных компьютерах в сети. Сокет, как и файл, характеризуется номером дескриптора и к нему применены файловые операции. В отличие от файла, сокет существует, пока на него ссылается хотя бы один из процессов. Сокет передается по наследству и с ним могут быть связаны один или несколько процессов.

Программная реализация алгоритмов анализа сетевого траффика. Поточный сокет обеспечивает двухсторонний, последовательный, надежный и недублированный поток данных без определенных границ. Перед началом обмена данными через поточный сокет процессы должны установить канал связи. Данный тип сокета использует протокол TCP [2].

Дейтаграммный сокет не гарантирует надежную передачу и получение данных в той последовательности, в какой они передаются. Этот способ передачи более быстрый, поскольку для него не требуются сложные операции установления виртуального соединения. Данный тип сокета использует протокол UDP [2].

Программный интерфейс сокетов похож на программный интерфейс файлов. Операции с сокетом осуществляются в основном по такой же схеме: открытие, чтение/запись, закрытие. Реализация операций имеет ряд отличий:

— сокет после открытия не привязан к конкретному сетевому адресу;

— сокет не связан с другим сокетом при обмене данными.

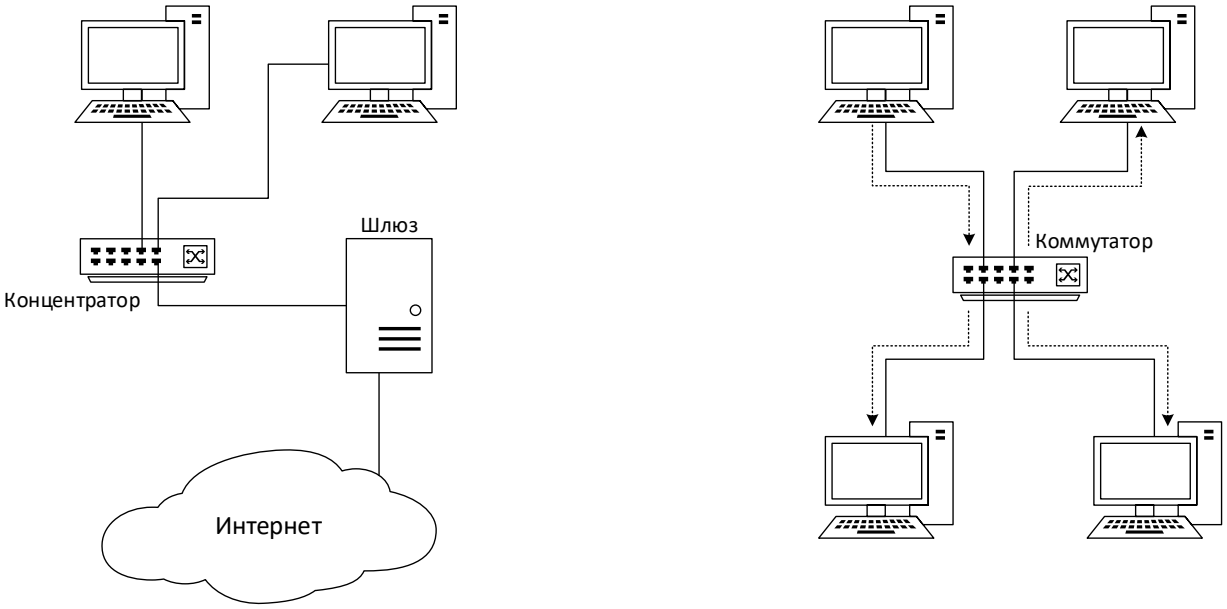
Программный интерфейс сокетов, прежде всего, расширен операциями, подготавливающими сокет к обмену данными. Одной из таких операций является привязка сокета к сетевому адресу и порту. Другие операции настраивают сокеты на соединение по виртуальному каналу связи. Полный программный интерфейс включает в себя более 20 функций.

Работа с сокетами включает несколько этапов: сокет создается, настраивается на заданный режим работы, применяется для организации обмена и ликвидируется. С точки зрения задачи перехвата сетевых пакетов наиболее важным является этап настройки режима работы сокета.

В нормальном режиме на Ethernet-интерфейсе используется фильтрация пакетов канального уровня. Если MAC-адрес в заголовке назначения принятого пакета не совпадает с MAC-адресом текущего сетевого интерфейса и не является широковещательным, то пакет отбрасывается. Очевидно, что в таком режиме работы перехват и анализ траффика невозможны.

Для решения данной проблемы следует использовать так называемый неразборчивый режим (promiscuous mode) работы сетевой платы, который позволяет ей принимать все пакеты независимо от того, кому они адресованы. Обеспечить попадание пакетов на интерфейс сетевой платы возможно нескольким способами. Все пакеты, отправляемые узлом, проходят через интерфейс сетевой платы. Анализ такого траффика может представлять лишь ограниченный интерес, связанный с анализом работы самого узла и не дающего общего представления о функционировании сети.

Практическую ценность имеют варианты, представленные на рисунке 1, когда на интерфейс сетевой платы узла (шлюза на рис. 1а и локального компьютера на рис. 1.б) попадают данные, передаваемые другими компьютерами. Это возможно при использовании в качестве интернет-шлюза отдельного компьютера или в случае, когда сеть построена на основе концентраторов, повторяющих сигналы, пришедшие на один порт, на всех остальных портах. Также, независимо от топологии и используемых устройств, каждый интерфейс в сети получает сообщения широковещательной и многоадресной рассылки.



а) б)

Рис. 1 . Примеры сетей для неразборчивого режима

Для разных языков программирования включение неразборчивого режима сетевого адаптера при создании сокета различается, также, как и принципы работы с самими сокетами. Для унификации этой процедуры и упрощения стандартных, часто встречающихся операций, используются различные библиотеки подпрограмм, наиболее известной из которых является Pcap (от англ. Packet Capture). В её состав входят NDIS драйверы для большинства современных операционных систем и низкоуровневые библиотеки для взаимодействия с драйверами сетевых интерфейсов. Библиотека Pcap предназначена для использования совместно с языками C/C++. Для работы с библиотекой Pcap на других языках, таких как Java, C#, используют обёртки [3]. В частности, для языков, совместимых с .NET существует библиотека SharpPcap, основными возможностями которой являются:

— кроссплатформенность. Библиотеку можно использовать на операционных системах Windows и Linux 32 и 64-битных версий;
— производительность — захват пакетов со скоростью до 3MB/c;
— удалённый захват пакетов;
— инъекции пакетов;
— поддержка AirPcap для работы с беспроводными сетями;
— использование Packet.Net для разбора пакетов.

Разбор пакетов является важной задачей процесса анализа сетевого трафика, что связано со сложной внутренней структурой самих пакетов, которая включает в себя несколько заголовков и полей данных, причём заголовок более высокого уровня является частью поля данных пакета более низкого уровня и попадает туда в результате инкапсуляции.

Инкапсуляция (в сетевых технологиях) — это процесс передачи данных с верхнего уровня приложений вниз (по стеку протоколов) к физическому уровню. При продвижении пакета данных по уровням сверху вниз каждый новый уровень добавляет к пакету свою служебную информацию.

Для анализа пакета необходимо произвести обратный процесс, который в различных источниках называется декапсуляцией или деинкапсуляцией [4], то есть разбором и извлечением

заголовков всех протоколов из исходного пакета, полученного на канальном уровне. Причём извлекаются именно заголовки, а не поля данных, так как они содержат всю полезную и статистически значимую информацию. Упомянутая ранее библиотека Packet.Net содержит в себе классы для представления пакетов наиболее распространённых сетевых протоколов стека TCP/IP, а также методы и свойства для проведения декапсуляции и лёгкого доступа к содержимому пакета.

Заключение. Для современного этапа развития сетевых технологий характерен как количественный (в связи с ростами объёмов трафика и ширины каналов связи), так и качественный рост (в связи с новыми прикладными задачами) потребностей в инструментах анализа трафика. Владение современными средствами для захвата и разбора пакетов позволит существенно повысить эффективность как самих алгоритмов анализа, так и процедуры их программной реализации.

Библиографический список.

1. Дуглас Камер Э. Сети TCP/IP Принципы, протоколы, структура. / Э. Дуглас Камер. — Москва : Вильямс, 2013. — 851 с.

2. Таненбаум Э. Компьютерные сети.. / Э. Таненбаум, Д. Уэзеролл. — 5-е изд. — Санкт-Петербург : Питер, 2012. — 960 с.

3. Перехват сетевых пакетов используя C# [Электронный ресурс]. — Режим доступа : https://habrahabr.ru/sandbox/30450 / (дата обращения: 18.04.2016).

4. Сетевые модели. Часть 3. [Электронный ресурс] — Режим доступа : http://infocisco.ru/network_model_encapsulation_pdu.html (дата обращения: 20.04.2016).