УДК 004.057.4 + 004.056 Вестник УрФО № 4(14)/2014. Стр. 32–35

Токарчук Н. А., Середкина Е. Д., Зюляркина Н. Д.

ПРОТОКОЛ TCP КАК СТЕГАНОГРАФИЧЕСКИЙ КОНТЕЙНЕР

Стеганография — это наука о скрытой передаче информации путём сохранения в тайне самого факта передачи. Отличительной особенностью стеганографии по сравнению с криптографией является то, что стеганография не скрывает содержимое сообщения, а скрывает сам факт передачи. При этом контейнером для сообщения может быть, например, изображение, видеофайл или аудиофайл. В данной работе рассматривается метод сетевой стеганографии, в котором в качестве контейнера для хранения стеганографического сообщения используются TCP-сегменты.

Ключевые слова: *информация, безопасность, защита информации, компьютерная безопасность, стеганография.*

Tokarchuk N. A, Seredkina E. D., Ziuliarkina N. D.

TCP PROTOCOL AS A STEGANOGRAPHIC CONTAINER

Steganography is the science of secure information transfer by means of secrecy of the fact of the transfer. In comparison with cryptography the distinctive feature of stagnography is that it does not hide the message content but hides the fact of transmission. When this happens an image, a video, or an audio file can be a container for a message. The authors dwell on the method of network steganography which uses TCP-segments as containers for steganographic messages.

Keywords: *information, security, information security, computer security, teganography.*

Основная идея

Впервые использование TCP-сегментов и механизма RTO для передачи стеганографических сообщений было предложено учеными из Варшавского университета в работе «Retransmission steganography and its detection» в 2009 году [1].

Механизм RTO (Retransmission Timeout) подразумевает повторную отправку пакетов, которые были повреждены или утеряны. Для отслеживания потери пакетов используется метод обработки ошибок ARQ (Automatic repeat request).

Когда TCP передает сегмент, содержащий данные, он помещает его копию в очередь повторной передачи и запускает таймер. При получении подтверждения для данного сегмента он удаляется из очереди. Если подтверждение не получено до окончания действия таймера, сегмент отправляется повторно [2].

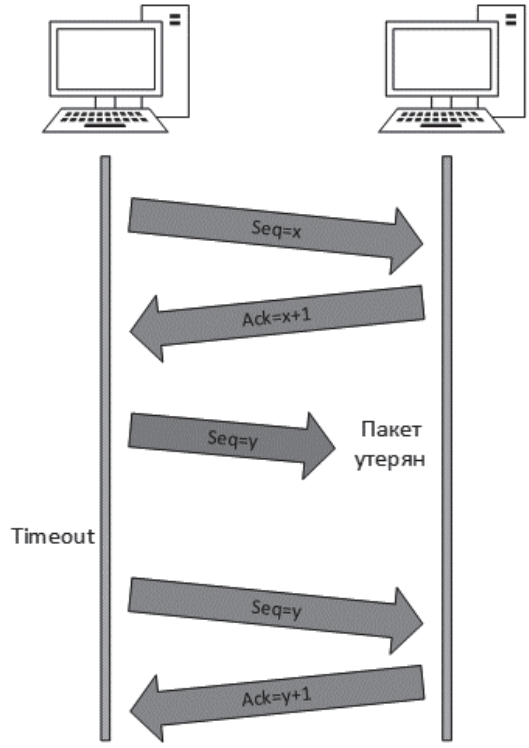


Рис. 1. Механизм RTO

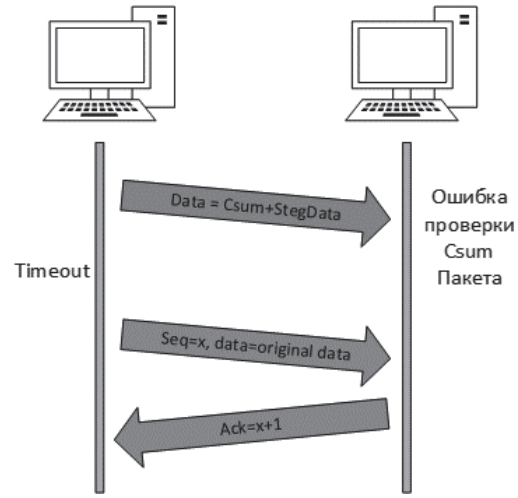


Рис. 2. Идея передачи стеганограмм

Мы реализовали свою идею использования механизма RTO для передачи стеганографических сообщений, которая заключается в паразитном использовании оригинальных сегментов с флагом PSH:

1) при отправке пользователем сегмента с некоторой вероятностью создается стеганографическая структура, состоящая из контрольной суммы стеганографического сообщения и непосредственно самого сообщения. Размер созданной стеганографической структуры должен соответствовать размеру поля данных в оригинальном сегменте;

2) получатель принимает сообщение, но из-за несовпадения контрольной суммы сегмент отбрасывается, а из данных извлекается стеганографическая структура и отправляется на проверку;

3) обработчик извлекает из стеганографической структуры контрольную сумму и сообщение, сам подсчитывает для сообщения контрольную сумму и сравнивает ее с извлеченной. При совпадении сумм сообщение считается принятым, иначе отбрасывается;

4) по истечении времени timeout отправитель передает сегмент, содержащий оригинальные данные;

5) получатель отправляет подтверждение.

Реализация

В данной работе мы взяли исходный код ядра Linux версии 3.17.1, который распространяется по лицензии GNU GPL v2 (https://www.kernel.org). Данная лицензия разрешает использовать исходный код для изучения, модифицирования и свободного распространения.

За основу мы взяли стек TCP/IP v4, реализация которого находится в файлах с префиксом «tcp_».

Механизмы отправки, приема и повторной передачи описаны в файлах tcp_input.c, tcp_output.c, tcp_ip_v4.c, tcp.c. Для реализации описанной идеи мы изменили данные файлы, как описано ниже.

Так как данные при отправке и приеме сообщений передаются от уровня к уровню (application, transport, network, link), то для исключения постоянного копирования данных и служебной информации в реализации протокола TCP ядра linux существует основная структура sk_buff (определена в заголовочном файле skbuff.h) [3].

В структуре sk_buff мы определили указатель на хранение оригинального сообщения, которое будет заменяться стеганографическими данными: unsigned char *stored_data.

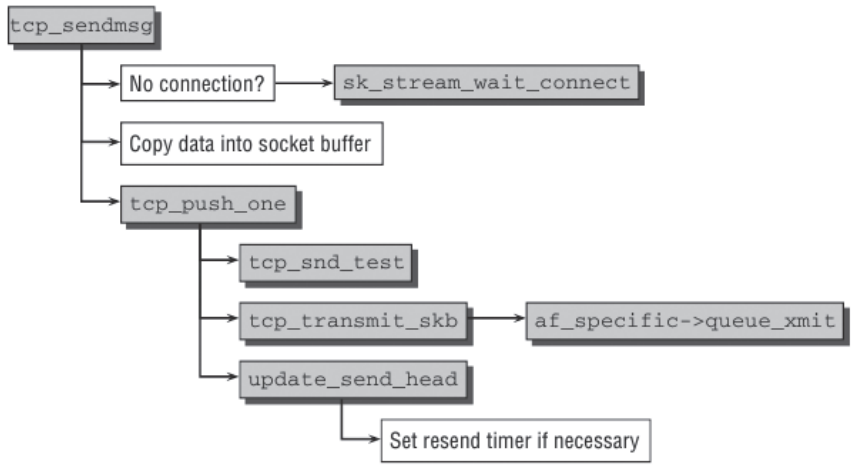


Рис. 3. Стек вызовов функций отправки сообщения

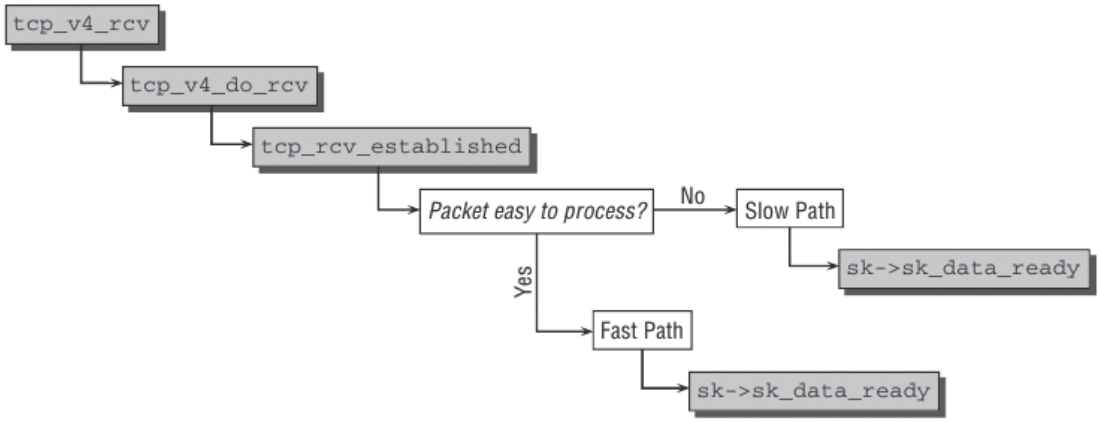


Рис. 4. Стек вызовов функций приема сообщения

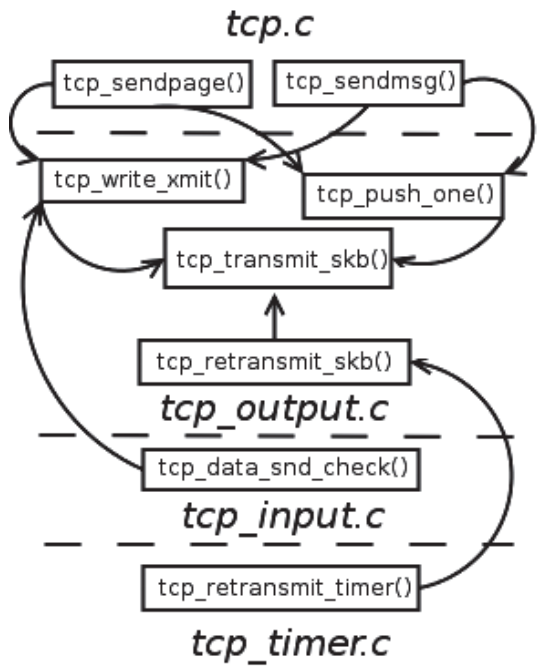


Рис. 5. Диаграмма работы механизма отправки сообщений протокола TCP

Для того чтобы модифицировать функции приема и передачи сообщений, рассмотрим, как они работают (рис. 3, рис. 4).

Так как протокол TCP является протоколом с гарантированной передачей данных, механизм отправки данного протокола является достаточно сложным. Чтобы правильно выбрать функцию отправки для модификации, рассмотрим диаграмму работы механизма отправки (рис. 5) [4].

Как видно из рис. 5, основной функцией для модификации отправки является функция tcp_transmit_skb(). В данной функции мы реализовали алгоритм отправки стеганографических сообщений.

Основной функцией приема является функция tcp_v4_rcv(). Мы модифицировали ее, добавив обработчик стеганографических сообщений.

Так как механизм RTO использует для повторной отправки функцию tcp_transmit_skb(), дополнительных изменений делать не нужно.

При модификации протокола важно сохранять его функциональность, чтобы не происходило потери пакетов. Потерей стеганографических пакетов на начальной стадии можно пренебречь.

Заключение

Описанная в данной работе реализация позволяет скрывать факт передачи сообщений, маскируясь под поврежденные сегменты, тем самым не изменяя логику работы TCP протокола. Однако данный метод не может полностью защитить от перехвата и подмены пакетов. Повысить защищенность передаваемых данных можно добиться использованием стеганографии в совокупности с криптографическими методами. В дальнейшем предполагается работа в данном направлении.

References

1. Mazurczyk W., Smolarczyk M., Szczypiorski K. Retransmission Steganography Applied, Poland: Warsaw University of Technology, 2010.

2. Transmission control protocol. Darpa internet program. Protocol specification. Defense Advanced Research Projects Agency, 1981.

3. Mauerer, W. Professional Linux® Kernel Architecture, Wiley Publishing, Inc, 2008.

4. How the Linux TCP output engine works // David S. Miller's Linux Networking Homepage. URL: http://vger.kernel.org/~davem/tcp_output.html (Date of access: 16.11.2014)

Токарчук Никита Александрович, студент ЮУрГУ. E-mail: nikita@tokarch.uk

Середкина Евгения Дмитриевна, студент ЮУрГУ. E-mail: vremendm@gmail.com

Зюляркина Наталья Дмитриевна, к. ф-.м. н., доцент ЮУрГУ. E-mail: toddeath@yandex.ru

Nikita Aleksandrovich Tokarchuk, student of South Ural State University. E-mail: nikita@tokarch.uk

Evgeniya Dmitrievna Seredkina, student of South Ural State University. E-mail: vremendm@gmail.com

Natalia Dmitrievna Ziuliarkina, Cand. Sc. Physics and Mathematics, associate professor of South Ural State University. E-mail: toddeath@yandex.ru