УДК 004.7

ВОЗМОЖНОСТЬ РЕАЛИЗАЦИИ ЗАЩИЩЕННОЙ ПЕРЕДАЧИ ДАННЫХ В ПКС СЕТЯХ С ИСПОЛЬЗОВАНИЕМ КРИПТОГРАФИЧЕСКОГО КАНАЛА

И.В. Хмелева, В.С. Ким

Представлен алгоритм поточного кодирования/декодирования данных и динамической смены ключей шифрования для передачи данных по сетям SDN.

Ключевые слова: SDN сеть; хеш-ключи; квантовый канал; алгоритм шифрования; кодер; декодер.

ABILITY TO IMPLEMENT SECURE DATA TRANSFER IN SDK NETWORKS USING CRYPTOGRAPHIC CHANNEL

I.V. Khmeleva, V.S. Kim

This paper presents an algorithm stream encoding / decoding data and dynamically change the encryption key for transmission over data networks SDN

Keywords: SDN network; the hash keys; quantum channel; encryption algorithm; encoder; decoder.

SDN сети в настоящее время приобретают все большую популярность, поэтому разработка алгоритмов защиты таких сетей является необходимой. Наиболее надежным средством защиты данных при передаче являются квантовые каналы, но из-за развития квантовых вычислений и технологий шифрования данных, это ставит под сомнение существование современной криптографии. Однако использование квантовых алгоритмов для формирования и передачи ключевой информации в симметричных криптосистемах весьма оправдано [1, 2].

Передача ключей кодирования при использовании квантового канала осуществляется на физическом уровне. Оборудование обладает информацией о компрометации канала в случае прослушивания. Скорость генерации последовательностей требует использовать динамическую смену ключа в алгоритмах кодирования и декодирования информации. Дополнительная криптостойкость обеспечивается сменой длины ключа шифрования [3, 4].

Возможным решением данной проблемы является использование алгоритмов симметричного шифрования. Недостатком таких алгоритмов является сложность обмена ключами [5]. Однако благодаря использованию криптографического канала, через который осуществляется пересылка ключей, эту проблему удаётся избежать. Достаточная скорость генерации квантовых последовательностей позволяет осуществлять поточное кодирование данных. В пакетах данных, генерируемых программой, передаётся информация о ключе (хэш-сумма) и о смещении относительно первого байта информации. Таким образом, при генерации нового ключа хэш-суммы перестают совпадать на кодирующем и декодирующем модулях, и производится перечитывание ключа.

В работе предложен алгоритм программного модуля, работа которого управляется внешними сигналами (например, от ПКС-контроллера [6]). Алгоритм осуществляет поточное кодирование или декодирование данных и динамическую смену ключа шифрования. Для решения задачи двухсторонней передачи и поддержки множества каналов (клиентов), выполняется сохранение информации о дескрипторах соединений и направлении передачи данных.

На рисунке 1 представлена блок-схема алгоритма, который реализует динамическую смену ключа шифрования данных и их отправку на конечный узел в закодированном виде, а также принимает закодированные данные и декодирует их.

Алгоритм имеет два режима работы:

Кодер:

1. При открытии соединения до TCP-порта данных сохраняет IP-адрес и TCP-порт клиента.
2. Инициализирует TCP-канал до конечного узла (декодера).

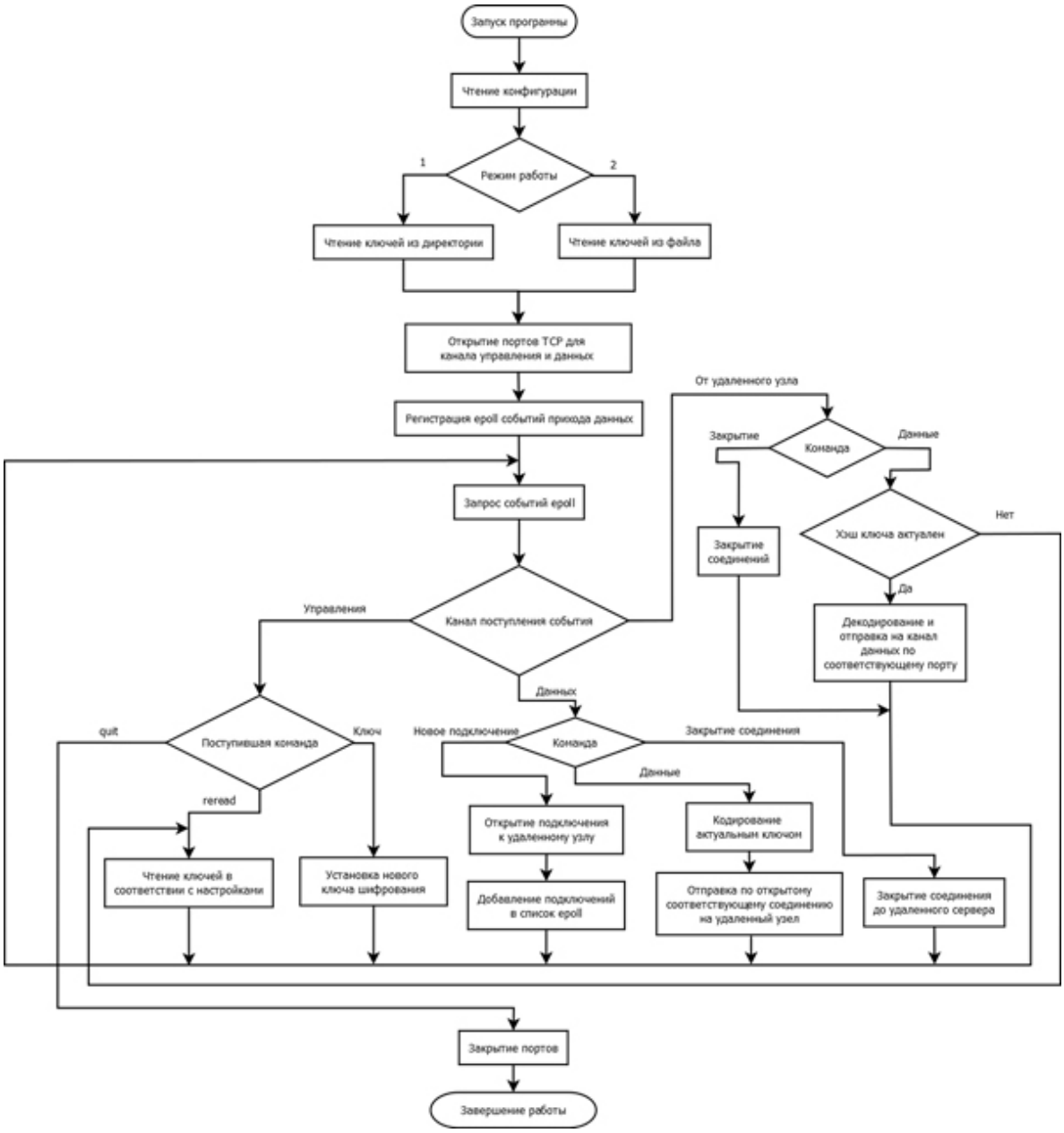

Рисунок 1 – Блок-схема алгоритма шифрования

3. Кодирует данные, приходящие по каналу данных.

4. Пересылает данные на конечный узел.

5. Принимает данные с TCP-порта, проверяет соответствие хэш- ключа, декодирует, пересылает.

Декодер:

1. При открытии соединения до TCP-порта данных сохраняет IP-адрес и TCP-порт клиента (кодер).

2. Инициализирует TCP-канал до конечного узла.

3. Проверяет соответствие хэш-ключей, декодирует данные, приходящие по каналу данных.

4. Пересылает данные на конечный узел.

5. Принимает данные с TCP-порта, пересылает.

Модуль обеспечивает 3 способа считывания ключа:

Режим 0 – чтение данных по curl-запросу:

- не требует дополнительных настроек;
- ключ считывается по каналу управления из нагрузки HTTP POST-запроса;
- размер ключа изменяется в зависимости от входных данных, что обеспечивает дополнительную степень защиты.

Режим 1 – чтение данных посимвольно:

- в настройках указывается директория загрузки ключей;
- в настройках указывается окончание имени файла ключа;
- считывание ключей осуществляется посимвольно до заполнения буфера (либо перебора всех файлов).

Режим 2 – чтение из структурированного файла:

- в настройках указывается полный путь файла, из которого будет производиться считывание ключа;
- ключ считывается построчно с применением маски.

Для работы алгоритма требуется произвести настройку путём изменения конфигурационного файла в соответствии с режимом работы (кодер/декодер), способом считывания ключа, параметрами TCP-порта канала управления, канала данных, а также IP-адреса и TCP-порта конечного узла.

После чтения конфигурационного файла производится инициализация ключей в зависимости от заданного способа их считывания (в случае использования режима 0, ключ ожидается к поступлению до начала приёма данных).

Для обеспечения работоспособности модуля с множеством потоков данных используется API мультиплексированный ввод-вывод epoll [7]. Благодаря этому отсутствует необходимость использования семафоров, потоков, межпоточного взаимодействия, упрощается структура программы благодаря тесной интеграции с системными функциями.

При инициализации программы осуществляется инициализация структуры epoll и подписка на события прихода данных на TCP-порт данных.

Так как все события epoll равнозначны, при инициализации подписки направление обработки данных (кодирование или декодирование) требуется определить явно, для чего используется знак перед дескриптором, сам же дескриптор извлекается из события по модулю.

Алгоритм обеспечивает динамическую смену ключа без разрыва установленных соединений. При запуске программы производится инициализация TCP-сервисов, ожидающих входные данные для кодирования/декодирования, а также управляющий сервис.

Для проверки предложенной концепции использовались две виртуальные машины в облачной среде OpenStack на базе дистрибутива Scientific Linux 7.2. На одной машине модуль запускался в режиме кодера и в качестве конечного узла указывался IP-адрес второй машины и TCP-порт входа модуля, запущенного на второй машине. В качестве конечного узла модуля на второй машине указан прокси-сервер squid3. Ключи поступают с ПКС-контроллера.

При использовании системы в подобной конфигурации осуществляется защищенный доступ к Web-страницам при установке в качестве HTTPS-прокси TCP-порта входа кодера первой машины. Данные запросов шифруются кодером, передаются на декодер, расшифровываются, обрабатываются прокси-сервером и проходят обратную цепочку преобразований перед отображением в браузере клиентской машины.

Подобное решение может быть использовано для широкого спектра приложений, способных работать через прокси, например, OpenVPN.

Литература

1. *Alkim Erdem, Ducas Léo, Pöppelmann Thomas.* Post-quantum key exchange – a new hope. // URL: https://eprint.iacr.org/2015/1092.pdf
2. *Campbell Peter, Groves Michael and Shepherd Dan*. SOLILOQUY: A CAUTIONARY TALE. // URL: http://docbox.etsi.org/Workshop/2014/201410_CRYPTO/S07_Systems_and_Attacks/S07_Groves_Annex.pdf
3. *Almuhaideb A.* Beyond Fixed Key Size: Classifications Toward a Balance Between Security and Performance / A. Almuhaideb, Phu Dung Le // 24th IEEE International Conference on Advanced Information Networking and Applications, AINA 2010, Perth, Australia, April 2010.
4. *Shyamala K.C.* Variable Size Block Encryption using Dynamic-key Mechanism (VBEDM) / K.C. Bai, M.V. Shyamala Satyanarayana // IJCA-Vol. 27. No.7, August 2011.
5. *Kampanakis Panos, McGrew David*. Next Generation Encryption. URL: https://www.cisco.com/c/en/us/about/security-center/next-generation-cryptography.html
6. *Venkat R. Dasari, Travis S. Humble*. OpenFlow Arbitrated Programmable Network Channels for Managing Quantum Metadata. URL: https://arxiv.org/abs/1512.08545
7. Linux Programmer's Manual: EPOLL. URL:http://man7.org/linux/man-pages/man7/epoll.7.html